User Evaluation Framework for Model Finding Research

by

Ryan Danas

A Thesis

Submitted to the Faculty of

WORCESTER POLYTECHNIC INSTITUTE

In partial fulfillment of the requirements for the

Degree of Master of Science

in

Computer Science

by

September 2016

APPROVED:

Professor Dan Dougherty, Major Thesis Advisor

Professor Lane Harrison, Thesis Reader

Professor Shriram Krishnamurthi, External Advisor

Dr. Tim Nelson, External Advisor

Professor Craig Wills, Head of Department

Abstract

We report the results of a series of crowd-sourced user studies in the formal-methods domain. Specifically, we explore the efficacy of the notion of "minimal counterexample" — or more colloquially, "minimal bug report" — when reasoning about logical specifications. Our results here suggest that minimal counterexamples are beneficial some specific cases, and harmful in others. Furthermore, our analysis leads to refined hypotheses about the role of minimal counterexamples that can be further evaluated in future studies. User-based evaluation has little precedent in the formal methods community. Therefore, as a further contribution, we discuss and analyze our research methodology, and offer guidelines for future user studies in formal methods research.

Acknowledgments

I would like to thank my reader, Professor Lane Harrison, for his excellent and detailed feedback on this paper, as well as his collaboration on this research. His experience played a key role, and is one of many reasons for the bright future in this research direction. It is always a pleasure collaborating with him, and I look forward to our future projects.

I would also like to thank my external advisors. Professor Shriram Krishnamurthi formulated and assigned me the general research question during my summer research assistantship at Brown University. He greatly assisted me in navigating this very new area, and is why I am looking at such a promising dissertation topic. I am very excited to be a PhD student under his advisement. Dr. Tim Nelson worked closely with me during my assistantship. He has been the best research partner I have ever worked with; a great colleague and a great friend.

Last and foremost, I would like to express my immense gratitude to my advisor, Professor Dan Dougherty, for his unwavering support during the beginning of my graduate studies. I set out on one of the most important paradigm shifts in my life during his advisement. I experienced an unbelievable amount of hardship during my transition. He was an always available and helped me when I needed it most. He may not have understood exactly what I was going through, but he did not let that hinder his support. He was asked more than anyone could ever expect a MS advisor to do. I am forever grateful for what he has done for me and my career.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

When users are trying to understand specifications, such as protocols, policies, and software designs, seeing concrete examples of their spec can be quite useful. Model-finders are tools that produce these concrete examples (models). These tools are useful for many applications. Suppose a software engineer is curious about their design, and would like to see implementations possible under their type hierarchy. Or, a security officer is concerned about holes in their access policy, and wants to see situations in which unauthorized persons gain entry. Maybe a network admin notices some erroneous packet forwarding behavior, and hopes to isolate the policy issue by looking at different instances of the general issue. Model finding enables users to better understand their respective specifications. An expanded discussion of this can be found in section 2.1.

Model-finding research focuses on improving the performance, informativeness, and usability of these tools. Generating models is quite difficult, as it is undecidable in general. Input interaction and output presentation are important to make these tools usable. We are not concerned with computational efficiency in this work. While we do tackle usability issues, they are auxiliary to our contributions. Our

research focuses on the informativeness of the output of these tools.

Specifications have a large, often infinite number of models. Considering the number is too large for any person to exhaustively explore, some obvious questions arise. Which subset of these models should be presented? What order should this subset be shown in? How can users explore this possibly infinite space of models? How do users best understand the information presented in these models? Current model-finders typically do not address these questions. Two exceptions are our previous research contributions and the associated tools, Aluminum and Razor (see section 2.2.4). In this work, we are concerned with exploring the first question: which models should be presented.

Mathematical considerations and intuition about what may be helpful to users directed the development of our previous work on Aluminum and Razor, and model-finding research in general. Formal methods researchers often conduct case studies consisting of hand picked examples tested with a small set of users, sometimes only consisting of the researchers themselves. This method of evaluation leaves the burden of calculating bias on the consumers of this research. While full user evaluations that report biases are possible to conduct, the expert population is small. Acquiring a sample size large enough to yield a fruitful statistical analysis is quite unlikely. Expanding the population outside the realm of expertise could make the analysis feasible. However, presenting such formal logic to non-experts is a non-trivial task. Our contributions are discussed in detail below, and are evaluated in chapter 7.

1.1 Contributions

1.1.1 User Model Preference and Performance

Models can be classified in any number of ways. Since we are concerned with the informativeness of models, we classify models by the amount of information they contain. Informally, one can think of this by the size of the models. Minimal models are the smallest, or contain the least information. That is, they only contain the information necessary to answer the question the model-finder was asked. A formal discussion of models, information, and minimality can be found in section 2.2. We find a significant split preference among users for minimal, and relatively larger models. Minimal models appeals to those concerned with only necessary entities, while larger models provide a sense of confidence. User model performance is a more objective judgment. While our results are not conclusive, our partial evaluation suggests that minimal models do benefit users performing specification refinement tasks, particularly for more difficult tasks.

1.1.2 Sources of Variance in User Studies

The formality of and matching syntax between the specification and model being presented each directly affect user ability. As each of these increase, the user performance increases. We report how these sources impact the ability to abstract knowledge, answer questions, and perform refinement tasks for both expert and non-expert users. The phrasing of the first order variables and relations has an especially drastic effect on non-experts. They bring predetermined knowledge to these experiments, which may relate to the naming conventions chosen. This can cause users to jump to possibly false conclusions without even absorbing the first-order information presented. This effect is the worst in simple tasks, as the solution is easy to figure out.

Chapter 2

Preliminaries

In this chapter we cover the logical background necessary to understand the concepts of model finding, minimality, and model-finder interactions discussed in this work.

2.1 Model Finding

Suppose a user writes a specification $\mathcal{T}$, which defines a software artifact, such as a product design, network protocol description, or access-control policy. The user is curious whether the logical consequences of $\mathcal{T}$ match their expectations. To explore this, the user provides their specification along with a sentence, $\mathcal{T} \cup \sigma$, to a model finder. The tool produces models, which illustrate the logical consequences of $\mathcal{T}$ and σ. For example, take this simple first-order specification and query.

$$\exists a.\ Object(a); \tag{2.1}$$

$$\forall o.\ Object(o) \ \Rightarrow \ File(o) \vee Folder(o); \tag{2.2}$$

$$\forall d.\ Folder(d) \ \Rightarrow \ \exists c.\ Object(c) \wedge InFolder(c, d); \tag{2.3}$$

$$\forall o.\ File(o) \wedge Folder(o) \ \Rightarrow \ Falsehood; \tag{2.4}$$

$$\sigma \equiv \exists f1, f2.\ Folder(f1) \wedge Folder(f2) \wedge InFolder(f1, f2);$$

This specification defines a simple computer file-system. The first sentence reads "there exists an Object a." The second reads "for all o, if o is an Object, then o is a File or a Folder." The third sentence reads "for all d, if d is a Folder, then there exists an Object c, and c is InFolder d." The fourth reads "for all o, if o is a File and a Folder, that is not allowed (Falsehood)." Suppose the user provided the following query along with this specification.

```
Object(a)    Folder(a)
Object(b)    Folder(b)    InFolder(b, a)
Object(c)    Folder(c)    InFolder(c, b)
Object(d)    File(d)      InFolder(d, c)
```

This model consists of four elements and eleven facts. The four left facts specify that a, b, c, and d are file system objects. The middle four facts say that there are three folders (a, b, c) and one file (d). The three right facts describe the file system structure. The file d happens to be in folder c, which is in folder b, which is in folder a.

These facts are by no means arbitrary. A model finder chooses them to satisfy the given first-order specification $\mathcal{T}$ and sentence σ. To elaborate, the first line of the spec requires $Object(a)$ to be true. Because of this, either $File(a)$ or $Folder(a)$ needs to be true. In this case, the tool chose $Folder(a)$. This causes yet another logical consequence. The folder needs a new $Object(b)$ to be inside of $Folder(a)$. The same process happens for b and c. For d, the tool chooses $File(d)$, which produces no further logical consequences. The query happens to already be satisfied, so no further action is taken.

In general, model finders generate these satisfying instances using well founded algorithms [36]. If a model $\mathbb{M}$ satisfies a specification $\mathcal{T}$, we write $\mathbb{M} \models \mathcal{T}$. We define the class of models of $\mathcal{T}$ as $C(\mathcal{T})$. In general, model finding is undecidable. For instance, the previous specification could be satisfied by an infinite chain of folders. The algorithm in that case would always choose the new *Object* to be a *Folder* to satisfy line 3 of the theory. To terminate, most model finders perform bounded searches.

2.2 Minimality

Specifications have a large, often infinite number of satisfying models, each of which conveys a particular amount of information. The size of a model defines the amount of information it contains. Minimal models are the models smallest in size. The formal definitions of this notion are discussed below. The number of minimal models is often infinite, so a further question is which minimal models do we present first to maximize coverage. This is future work.

2.2.1 Homomorphisms

We can define the relationship between information in models using homomorphisms. An arbitrary homomorphism $h : |\mathbb{M}| \rightarrow |\mathbb{N}|$ is a map from the domain of $\mathbb{M}$ to $\mathbb{N}$, such that for every relation R and tuple $\langle e_1...e_n \rangle$ of elements of $|\mathbb{M}|$, if $\mathbb{M} \models R[e_1...e_n]$, then $\mathbb{N} \models R[h(e_1)...h(e_n)]$. An injective homomorphism is exactly an abitrary homomorphism where h is a one-to-one function. That is, for every element e_i in the domain of $\mathbb{N}$, there is exactly one element $e_{\mathbb{M}}$, such that $h(e_{\mathbb{M}}) = e_i$.

[@@ Each of these notions of homomorphism] defines a pre-order on models:

$\mathbb{M} \ll \mathbb{N}$, if there is exists some homomorphism from $|\mathbb{N}|$ to $|\mathbb{M}|$. [@@ We write] $\mathbb{M} \approx \mathbb{N}$ to mean $\mathbb{M} \ll \mathbb{N}$ and $\mathbb{N} \ll \mathbb{M}$.

2.2.2 Minimal models

A model $\mathbb{M}$ is minimal for a class $C(\mathcal{T})$ of models if for every model $\mathbb{N}$ in $C(\mathcal{T})$, $\mathbb{M} \ll \mathbb{N}$ implies $\mathbb{M} \approx \mathbb{N}$.

2.2.3 Model Coverage

The cone of a model, $Cone(\mathbb{M})$, is the set of all models $\mathbb{N}$, such that $\mathbb{M} \ll \mathbb{N}$. Observe that the cones of the minimal models cover the space of all models. That is, the minimal models act as a set of support for all other models. With this set of support, we can now explore all possible models by starting at a minimal one, and augmenting them with additional facts.

2.2.4 Minimal Model Finders

Aluminum [29] is a tool built on top of Alloy [18] that produces minimal models under injective homomorphisms. Razor [34] produces minimal models under standard homomorphisms. Both tools enable exploration through the minimal models and set of support to any other model.

2.3 Refinement Tasks

Quite often users have specific concerns about a specification $\mathcal{T}$ they have written. A user states their concern as an assertion σ. The negation of assertion is provided to a model finder. Given $\mathcal{T} \cup \neg\sigma$, the tool attempts to find a model in which the assertion does not hold true under the specification. Any model found acts as a

counterexample to the failing assertion. If the tool finds no models, the assertion is true for models up to the set bound. Although, it is possible a model exists past these bounds. Setting sufficiently large bounds addresses this; however, the bounds is a burden placed on the user. Luckily, Jackson's [18] small scope hypothesis states that if a counterexample exists, there most likely is a one with small size. To the extent that this, *empirical,* assertion holds, the user gains confidence in their assertion if the tool doesn't find (bounded) counterexamples.

In general, a refinement task consists of the following steps:

Input: User provides $\mathcal{T}$ and σ

Feedback: Tool produces model $\mathbb{M} \models \mathcal{T} \cup \neg\sigma$, if one exists (in the case that σ is NOT entailed by $\mathcal{T}$)

Trace: User conjectures which part of $\mathcal{T}$ requires refinement.

Edit: User edits $\mathcal{T}$ to try to nullify the counterexample $\mathbb{M}$.

2.3.1 Alloy Example

Let us run through a refinement task for the following Alloy class grading policy. The specification defines two types of people (Students and Professors), classes, assignments. Classes have a set of student assistants and a single instructor. Assignments have an associated class and a set of assigned students. A person can grade an assignment if they are a student assistant, or the instructor. The assertion states that none of the assigned students should be able to grade their own assignment.

Alloy produces a counterexample with several classes, professors, students, and assignments. Among all these facts, a student happens to be assigned to an assignment for a class they are an assistant for! That set of facts should have directed you

```
abstract sig Person {}                                              (2.1)
sig Student extends Person {}                                       (2.2)
sig Professor extends Person {}                                     (2.3)
sig Class {                                                         (2.4)
  assistant_for : set Student,                                      (2.5)
  instructor_of : one Professor                                     (2.6)
}                                                                   (2.7)
sig Assignment {                                                    (2.8)
  associated_with : one Class,                                      (2.9)
  assigned_to : some Student                                        (2.10)
}                                                                   (2.11)
pred CanGrade(s : Person, a : Assignment) {                         (2.12)
  s in a.associated_with.assistant_for                              (2.13)
  or s in a.associated_with.instructor_of                           (2.14)
}                                                                   (2.15)
assert NoOneCanGradeTheirOwnAssignment {                            (2.16)
  all s : Person | all a : Assignment |                             (2.17)
  CanGrade[s, a] implies not s in a.assigned_to                     (2.18)
}                                                                   (2.19)
checkNoOneCanGradeTheirOwnAssignment                                (2.20)
```

Figure 2.1: Gradebook Specification

towards line 13, which should have **and not** s **in** $a.assigned\_to$ added to it. Alloy fails to produce further counterexamples after this refinement is made.

Chapter 3

Background

The following chapter details the motivation of our contributions discussed in the introduction, challenges reaching those findings, our method of achieving those results, and any relevant, non-mathematical definitions.

3.1 Motivation

The formal methods community in general does not focus on user evaluations. Hahnle, in his note in the Automated Reasoning Newsletter [15], remarks "Our community currently focuses on theoretical analyses, on empirical evaluations, and on systems' competitions. But hardly any attention is given to user evaluations." The statement aligns with our motivations.

User evaluation is a well established scientific method; however, it has not gained wide acceptance in the formal methods community. In his note, Hahnle remarks about his own paper being rejected from a top formal methods conference on the grounds of not being technical enough. A particular reviewer questioned whether "Is it really necessary to know the exact hypothesis of the evaluation, or which hypothesis can be rejected using which kind of test?" Hahnle replies, "Why, yes of

course! Or else you are just claiming nonsense! You could just as well ask 'Is it really necessary to prove soundness using all this complex mathematical stuff?"' This evaluation method is a well-respected staple in other scientific disciplines such as human computer interaction, visualization, life sciences, social sciences, psychology, and medicine. It is time for user studies to find their place in the formal methods community.

3.2 Challenges

Formal methods researchers usually conduct case studies consisting of hand picked examples tested with a small set of experts. Conducting larger scale case studies with more examples, i.e. user studies, in a formal setting poses several difficulties. In most contexts, the small number of available experts will not be a large enough sample to measure effects confidently. In the worst case, the scarcity of data could obfuscate trends that lead to interesting hypotheses to be tested.

A straightforward way to subvert this issue is to not survey just experts. However, most of the research and tools to be evaluated are not intelligible to laymen in their original form. To present formal methods tasks to non-experts is a non-trivial task. Furthermore, making this information presentable to everyday people has its own side effects. Without the underlying logical structure, non-experts can interpret the information however they please. Both of these major challenges are discussed below.

3.2.1 First-Order Information Translation

Translating a first-order specification and model, then presenting it to a person completely unfamiliar with logic is not easy. While first-order information can be read

in English, it is challenging to fully mechanize such a syntactic manipulation. One could rely on the first-order logic grammar to guide the translation. However, the result usually will not read like natural language. A logician could overcome this syntactic awkwardness, but a non-expert lacks the knowledge to fill in the missing context. Thus, a human translation step is required. This handwork adds the missing context. Unfortunately, it also adds bias. Addressing this bias is compounded by the chosen natural language phrasing.

3.2.2 Presented Translation Phrasing

The hand-picked context and chosen phrasing adds bias to non-expert experiments. We suggest that non-experts process first-order information differently than logicians. They are most likely unaware of the structure that once existed, and the semantics behind it. To them, the model-finding task looks like a regular word problem. The particular phrasing of the first-order elements, relations, and so-forth also influence the study. We believe non-experts reading first-order information like word problems. They most likely have the natural language semantics in mind instead of the first order ones, due to their lack of experience writing the specification.

Choosing an arbitrary context influences the study. This can be solved by evaluating several context choices, but that will increase cost and vulnerability of the user evaluation. Instead, a context that accurately reflects what a trained logician understands during their model-finding task should be found. While this addresses one side-effect of the translation, the real-world phrasing of the problem still poses problems. This can be solved similarly to the first problem. Except, the more desirable solution, to use a single justifiable phrasing, is no longer viable. Intuitively, the original phrasing should be the proper phrasing. However, the original phrasing does not have the same effect on non-experts as it does on logicians. Unfortunately,

multiple trials with phrasing variations must account for this bias.

3.3 Approach

We set out to evaluate our minimal model definitions and address these challenges in tandem. Through iterative refinement, we measured the effect of the above biases in the form of variance. By identifying sources of variance in each of our experiment trials, we garnered an understanding of how non-experts process various translations and phrasings we presented them. We surveyed non-expert participants on Amazon's Mechanical Turk, a popular crowd sourcing tool. For further details on our reasons for surveying Mechanical Turkers, see section 8.2.

3.3.1 Minimality User Preference and Performance

We presented non-experts our translated and phrased refinement tasks, paying careful attention to the effects of the translation and phrasing. Given a particular refinement task, we constructed a translated and phrased version. This non-expert version was then split into minimal and non-minimal variants. To avoid bias, the non-minimal models were the default output from the tool originating the refinement task. The set of minimal models was always static from the previous definitions.

Applicability of Refinement Tasks

Table 3.1 shows the results of a multiple choice question answered by 18 experts in one of our preliminary studies. The study is discussed in greater detail in the later chapters. The table shows the frequency how experts used models in their respective tools. A chi-squared test found a significant difference in the frequencies for the third task with a p-value of 0.047. The third task, using a model to suggest

additional constraints required in a specification, was exactly what a refinement task is defined as. While we cannot claim the universal applicability of this task, it is at least popular among experts in the model finding community.

	Often	Sometimes	Rarely
Help find the source of a behavior or bug in a specification	5	6	3
Suggest additional properties to check of a specification	5	7	2
Suggest additional constraints required in a specification	10	2	2
Identify unexpected relationships between parts of a specification	5	5	4
Foster human confidence in the behavior of a specification	7	2	5

Table 3.1: Expert Preference Survey - Question 4 Results

3.3.2 Mitigating Challenges

By collecting data at each step of the non-expert task, we reconstructed how the user interacted, parsed, comprehended, and acted upon the first-order information we presented. From this data, we searched for trends that may exemplify effects caused by the above challenges. If any effects were discovered, they are dealt with to remove bias in the next iteration. This process was repeated until no such bias was found; that is, the non-expert results reflected the expert results.

3.4 Definitions

The following definitions are paraphrased from Intuitive Statistics [26].

3.4.1 Studies

Subject and Treatment

A *treatment* is an experiment with a particular configuration of independent and dependent variables. A *subject*, also known as a respondent, is a person being surveyed who completes a certain treatment.

Independent and Dependent Variables

Variables are the types of data recorded from respondents during an experiment. An *independent variable* is the part of a study changed for each treatment. The values of *Dependent variables* are expected to change depending on the treatment.

Within and Between Subjects Experiments

There are two common ways to run experiments. A *between-subjects* study consists of treatments with disjoint groups of respondents. That is, each respondent only sees one treatment. A *within-subjects* study consists of respondents given a set of treatments. That is, each respondent sees multiple treatments with adjusted independent variables.

3.4.2 Results

Parametric and Non-Parametric Data

Parametric data fits a normal distribution: the data matches the shape of a standard bell curve. Parametric, or normal data is preferred, because it enables a greater range of statistical testing. *Non-parametric data* does not fit a bell curve, and thus more sophisticated tests much be used to analyze the data. These tests usually have very specific conditions and assumptions. They also make it more difficult to achieve large median and mean differences.

3.4.3 Analysis

Shapiro Test

This test can be used to analyze whether data truly fits a normal distribution.

Friedman Test

This test is performed on matched groups of ordinal data. Ordinal data can be categorized into consecutively numbered bins. For example, preferential ranks where respondents vote for their first, second, third, etc. choice is ordinal data. The test confirms whether there is a significant difference in the ordinal data. Following the previous example, a positive test would suggest a particular option is preferred over the others.

Chi-Squared Test

The chi-squared test compares data for independence. That is, if one or more groups differs from the other groups. This test has many applications because most data

can be manipulated to be proper input for this test. It is also often used to test whether experimental data defers from the expected result.

Mann-Whitney U Test

A Mann-Whitney U test is used to test whether two groups of between-subjects non-parametric data differ. This test is a non-parametric version of the well-known t-test for unpaired data. Since it compares medians instead of means as the t-test does, it usually has less of a chance of achieving significance.

Chapter 4

Results: User Model Preference

Both expert and non-expert users of model-finders fail to agree on which models should be presented. However, there is significant split preference for minimal, and relatively large models. Minimal models appeals to those concerned with only necessary entities, while larger models provide a sense of confidence. Moreover, the same exact bi-modal preference is exhibited in both expert and non-expert users.

4.1 Expert Preference Study

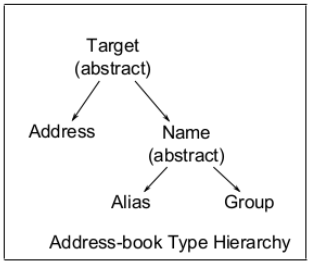
We surveyed 18 experts from the Alloy-B-and-Z (ABZ) 2012 conference and Brown University. The survey consisted of a few general model-finder usage questions, and three analysis tasks. Each analysis task consisted of three specifications, each with a set of five models. The users reviewed a specification, chose which models they would prefer to see first, and explained their model choice. The specifications were presented by type hierarchy, relations, and factual constraints for brevity and quick comprehension. Due to small sample size, the results reported in this section lack confidence. However, since these are venerable model-finding users, their opinions still hold weight. Also note, their hypotheses are later confirmed in the crowd-

sourced study discussed later in this chapter.

4.1.1 Addressbook

Figure 4.1 specifies the types of an address book that maps names to targets, which can be an address, an alias, or a group of targets. Constraints are also provided to restrict 1. cyclic lookups, 2. aliases to a one-to-one mapping, and 3. empty address books.

Consider the following specification for an address book that supports groups of addresses and aliases for single addresses. The hierarchy of types (aka, sorts, signatures, or domains) is on the left. The relations and constraints are on the right.



Relations

addr: Names × Targets

Constraints

1) addr is acyclic
2) addr maps each alias to
   no more than one name
3) there is at least one Address

Figure 4.1: Addressbook Specification

The models in figure 4.2 can be categorized as follows. The first and fifth models are minimal. The third model is medium sized, while the second and fourth models are large. Medium and large are informal and relative terms.

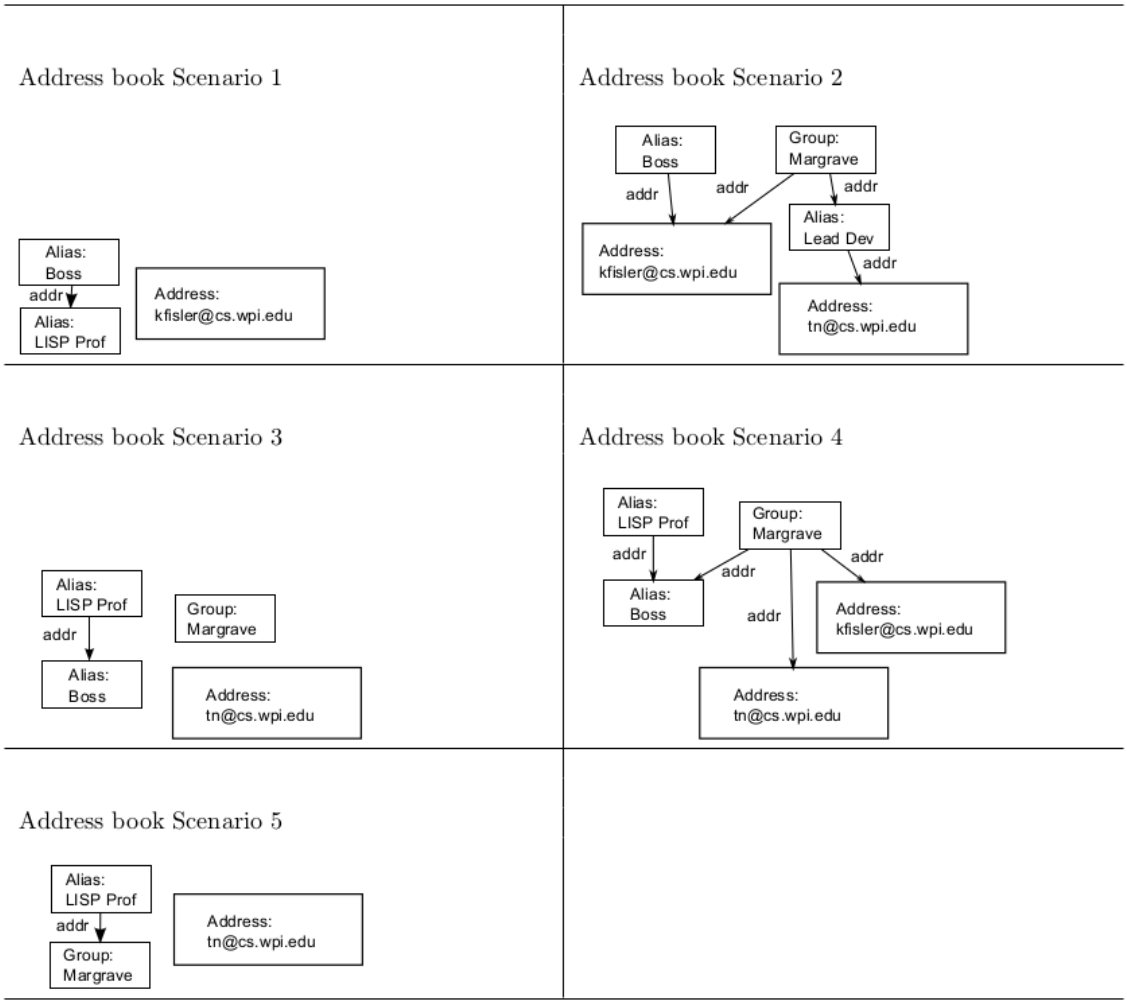


Figure 4.2: Addressbook Models

For this specification, there is no discernible preference for a particular model. It is worth noting that these are exploratory models; no assertion was provided along with the spec. This note affected our crowd-sourced results, and we suggest the same conclusion: without a specific task given, respondents do not decide preference under the same criteria.

Model #	Model Type	Voted to See First
1	Minimal	57.14%
2	Large	37.5%
3	Medium	42.86%
4	Large	42.86%
5	Minimal	57.14%

Table 4.1: Addressbook Preference Results

4.1.2 Gradebook

Figure 4.3 specifies a grading policy that consists of students, professors, classes, and assignments. Particular students are assigned as TAs. Only TAs and the Professor can grade assignments.

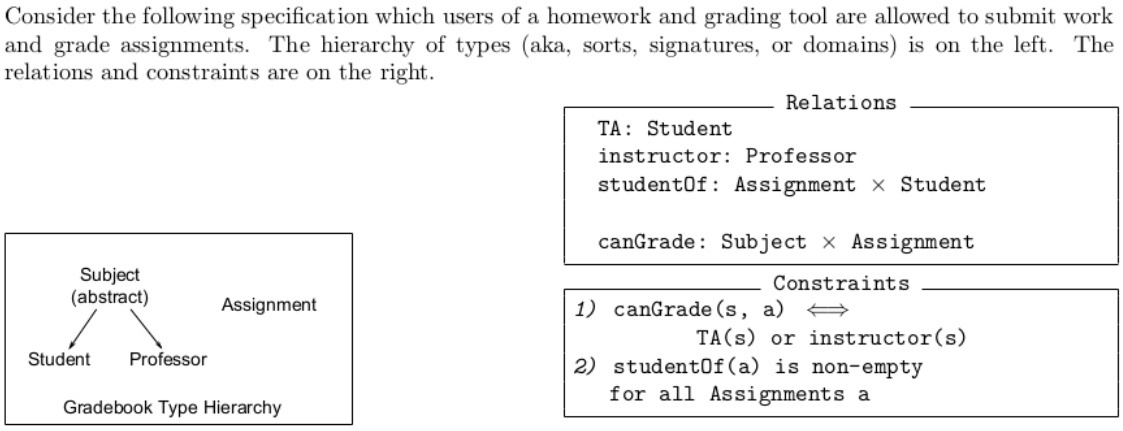


Figure 4.3: Gradebook Specification

The models in figure 4.4 can be categorized as follows. Models 1, 2, and 3 are minimal, while models 4, and 5 are large. Additionally, models 1, 3, and 5 exhibit an interesting problem. In these models, students submitting for the assignment can

grade it. While no assertion is given, this problem violates grading policy intuition, or common sense.

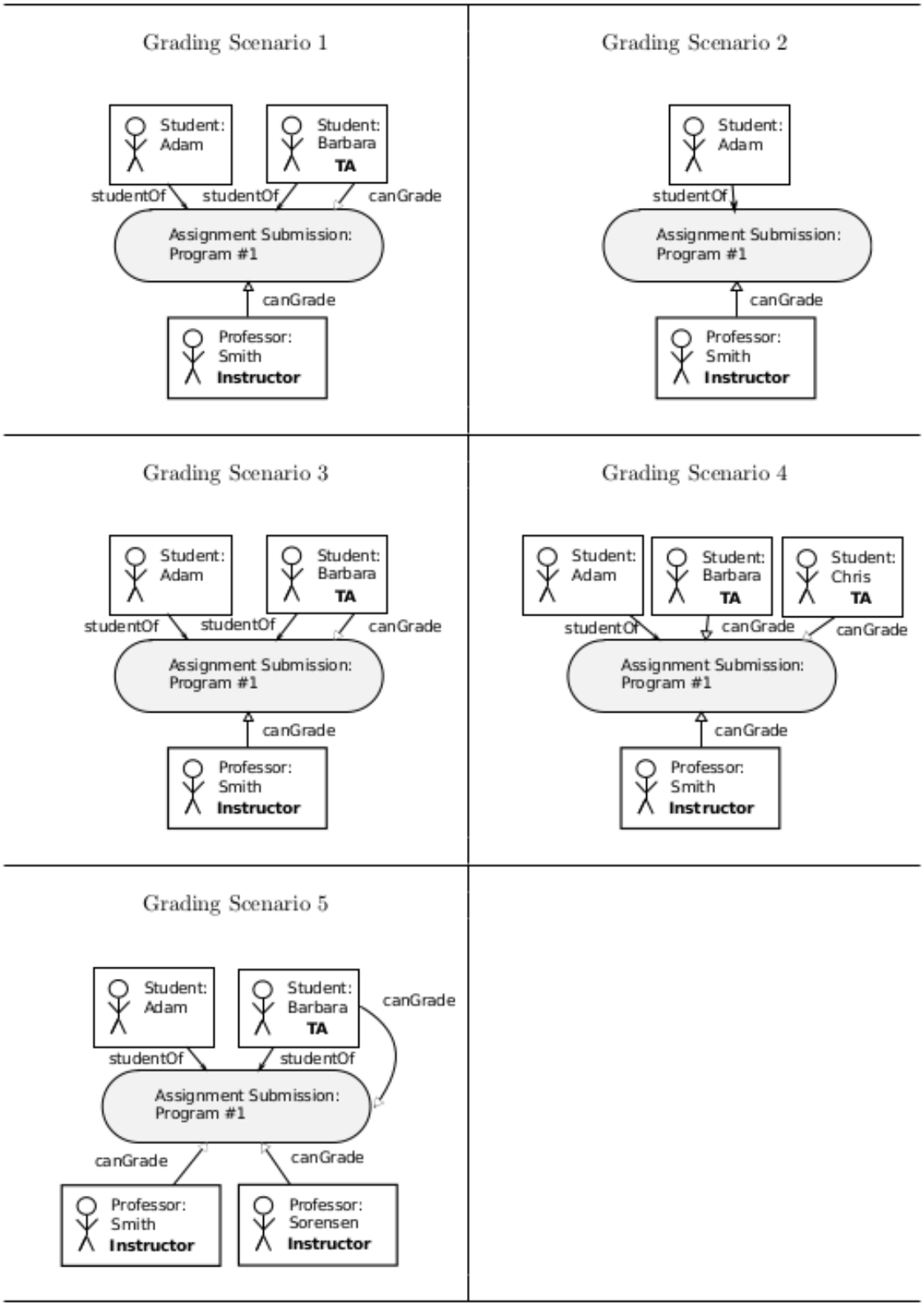


Figure 4.4: Gradebook Models

Table 4.1.2 denotes the percentage of experts who marked each model as being useful to see first. The models are also marked with "Problem" if it exhibits a Teaching Assistant able to grade their own assignment. This was not provided as an assertion, but it is intuitively considered a problem by the researchers, and most participants.

Model #	Model Type	Voted to See First
1	Minimal Problem	74.13%
2	Minimal	50%
3	Minimal Problem	50%
4	Large	28.57%
5	Large Problem	66.67%

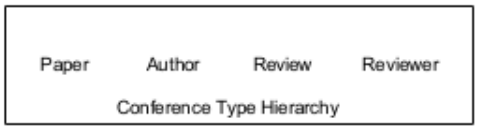
Table 4.2: Gradebook Preference Results

For this specification, the problem models are explicitly preferred. However, there is no clear preference between minimal and non-minimal problem models. The larger, non-problem models are seen as the least useful. As with the last task, there was no assertion provided to expose these particular models as a problem. This notion was merely intuitive for both the researchers and participants.

4.1.3 Conference Manager

Figure 4.5 specifies a paper review system. The specification for this system includes the following axioms: *Papers have a non-empty set of authors. Reviews are for a non-empty set papers. A review is written by one reviewer.*

Consider the following specification of the relationships between reviews, reviewers, papers, and authors in a conference manager. The hierarchy of types (aka, sorts, signatures, or domains) is on the left. The relations and constraints are on the right.



Relations

hasAuthor: Paper × Author
reviewFor: Review × Paper
writtenBy: Review → Reviewer

Constraints

1) There is at least one paper, one author, one review, and one reviewer
2) The relations hasAuthor and reviewFor are total (i.e. every paper has at least one author and every review is about at least one paper)

Figure 4.5: Conference Manager Specification

The models in figure 4.6 can be categorized relative to the first minimal model. The fifth model has only a few extra facts. Models 2 and 4 have a slightly more extra facts. The third model has many more facts than the first model.

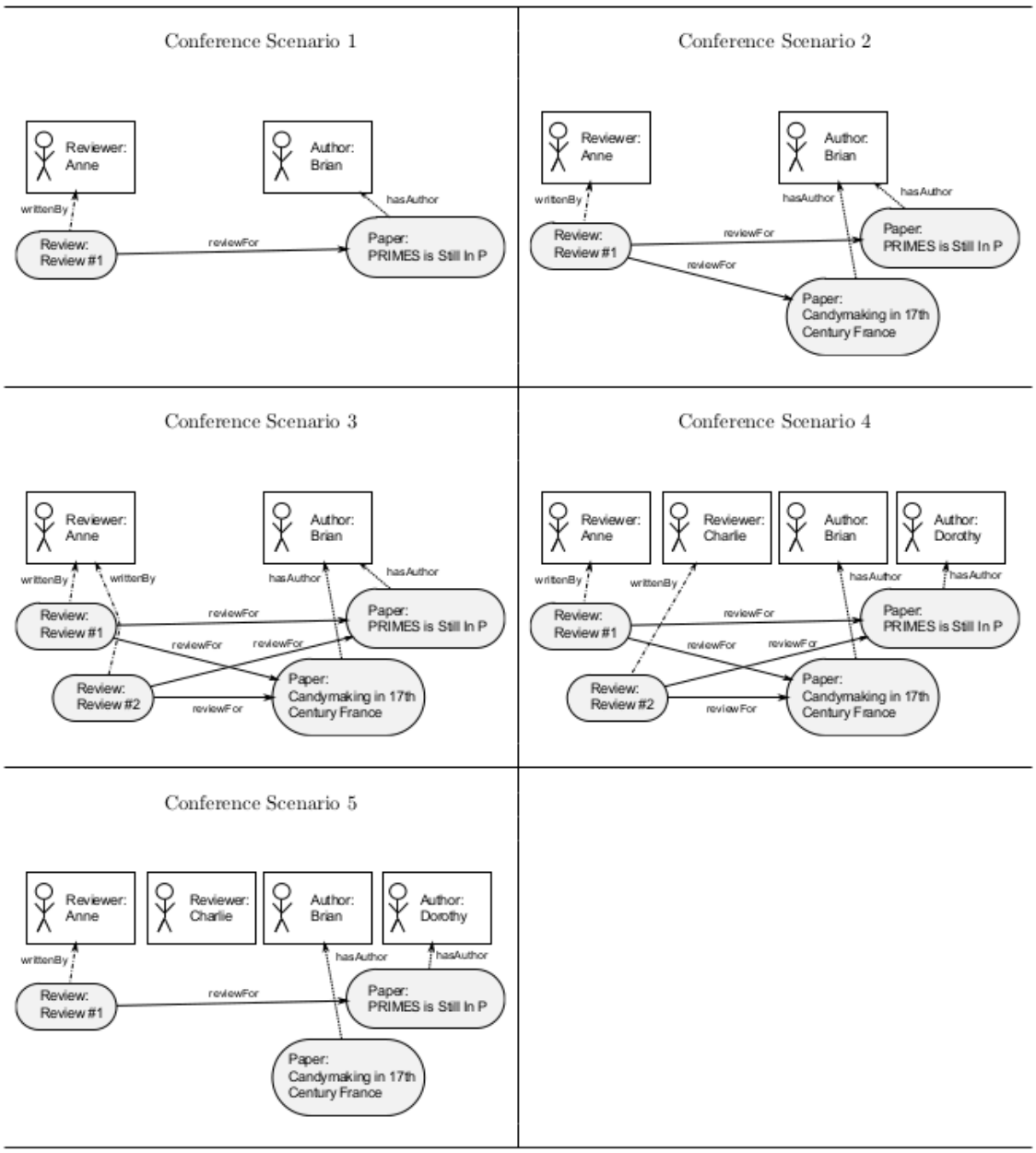


Figure 4.6: Conference Manager Models

Table 4.1.3 denotes the percentage of experts who marked each model as being useful to see first. The models are additionally categorized by the type and quantity of additional information it provides over the minimal counterpart, as described above.

Model #	Model Type	Voted to See First
1	Minimal	57.14%
2	Some new binary relation tuples	33.33%
3	Many new binary relation tuples	28.57%
4	Some new binary relation tuples	33.33%
5	Few new binary relation tuples	83.33%

Table 4.3: Conference Manager Preference Results

In this specification, a non-minimal example is explicitly preferred. Furthermore, the model is only slightly larger than the minimal one. The even larger examples begin to decrease in preference. While there were no comments explicitly mentioning sources of confusion, there seems to be an information overload issue as more binary facts are added to the minimal model.

4.2 Initial Crowd-sourced Preference Study

We sought out to replicate this bi-modal preference with a sample size that could produce confident results. In the initial following crowd-sourced study, we presented 40 non-experts the addressbook specification, informally translated and rephrased as a phonebook. The users were then asked to analyze a set of models. Finally, they ranked each model in distinct consecutive order. The informal translation and phrasing of the addressbook specification is shown in figure 4.7.

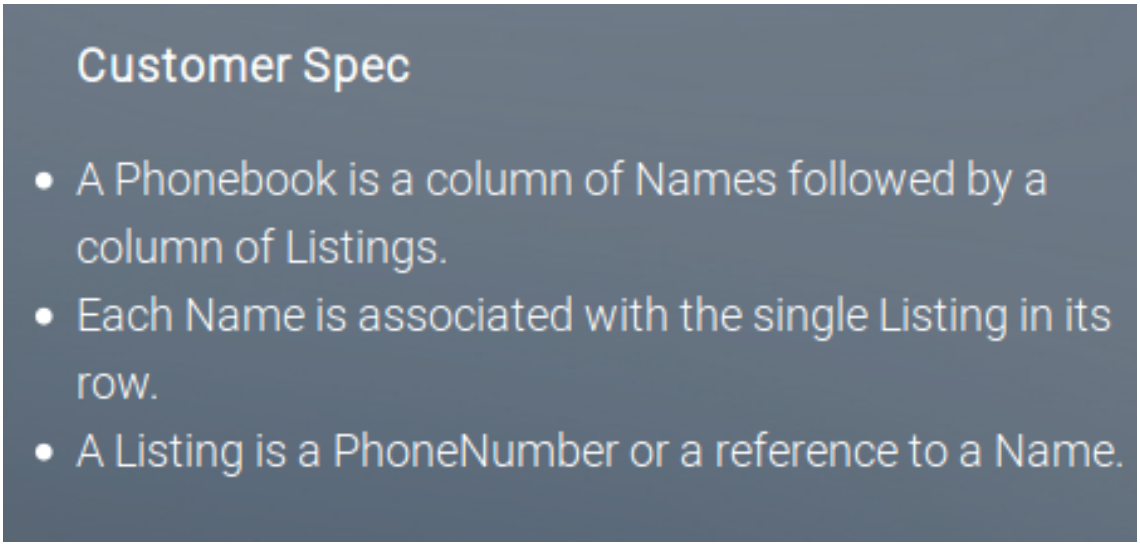


Figure 4.7: Phonebook Specification

The models the respondents had to rank can be seen in figure 4.8. Similar to the expert study, each model exhibited a particular interesting property allowed by the specification. Example 1 is classified as the normal model, because it keeps in line with user intuition and the general expectation from the specification. Example 3 has two aliases (names) that resolve to the same address (phone number). Example 4 has a missing address in the lookup for the alias Jack, as this is not explicitly denied in the specification. Example 5 is exhibits the same issue as the previous example, except using a cyclic reference. Again, this was not explicitly denied in the phonebook specification above. The final example is a union of all the previous examples.

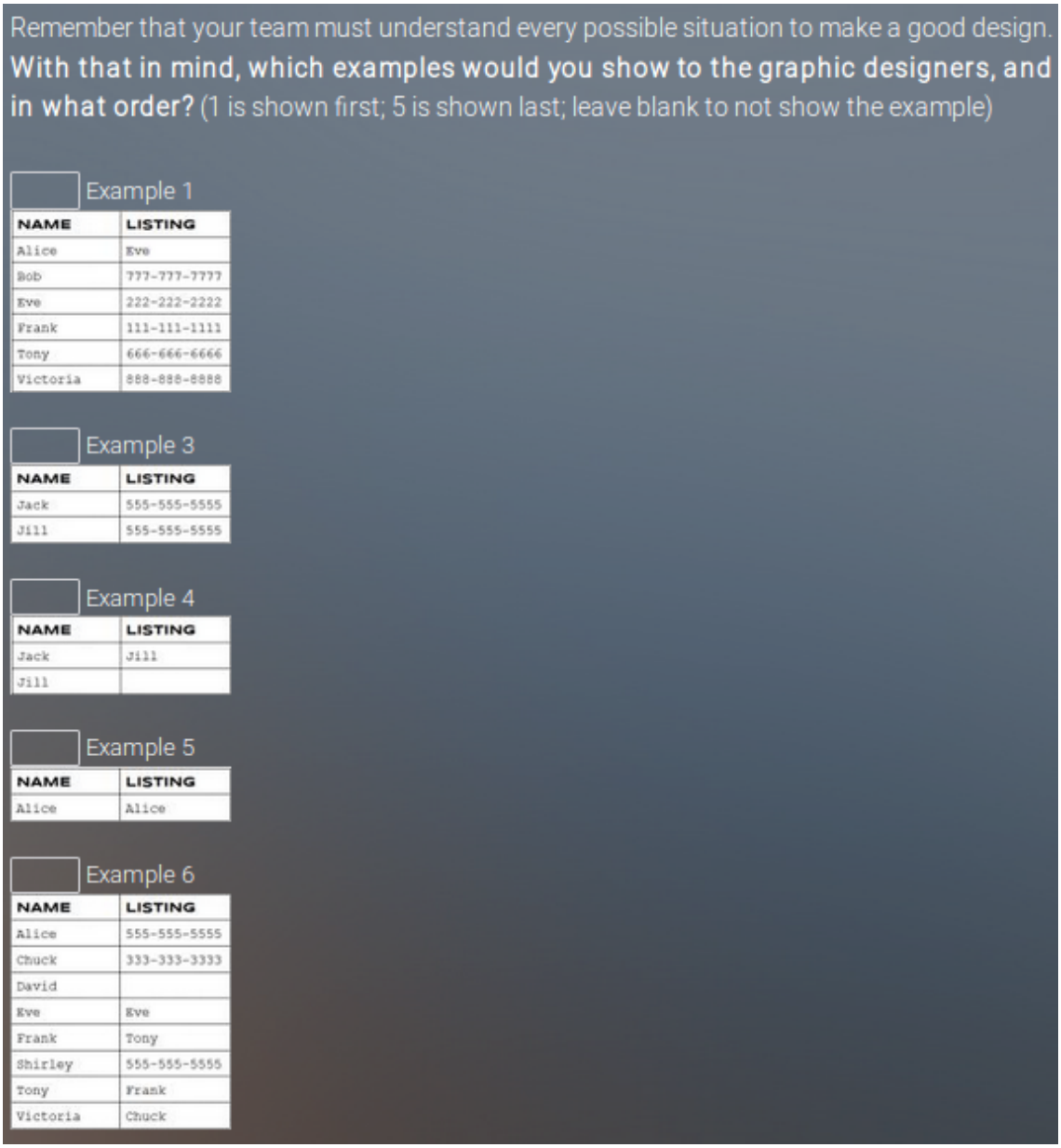


Figure 4.8: Phonebook Models

The next table displays the voting data for the models ranked by preference. Each example listed in the leftmost column. For each example, the number of votes for each ranking is listed, as well as a shorthand name for the descriptions included above.

Example Type vs Rank Tally	1st	2nd	3rd	4th	5th
Normal	16	17	2	5	0
Shared Number	15	12	12	0	0
Empty Reference	1	4	10	12	6
Cyclic Lookup	0	1	6	12	12
Complex Combination	8	6	5	3	12

Table 4.4: Unguided User Preference

Without an explicit assertion to test, each respondent ranked the models according to their own criteria. Some defined preference based on model size. Others preferred those with interesting properties, such as cyclic lookups or missing references. Without an assertion to guide their intuition, the respondents came up with their own. Thus, the rankings above hold no significance. Giving the respondents an explicit task should resolve this issue.

4.3 Followup Crowd-sourced Preference Study

In the followup study, we gave 200 non-experts a refined informal translation and phrasing, as well as explicit assertion to check. The respondents were then asked to rank the models based on their usefulness for understanding the problem in the specification. Given this explicit refinement task, their rankings significantly reflect what the experts exhibited in the ABZ study.

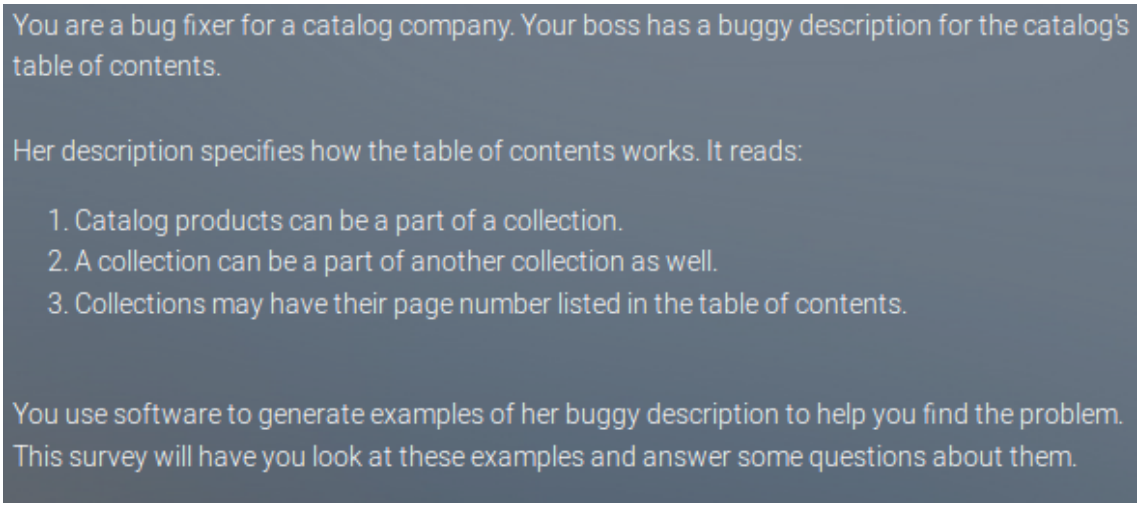


Figure 4.9: Catalog Specification

As seen in figure 4.9, we changed the phrasing to a catalog table of contents. The phonebook was not a good domain for some of the allowed properties of the addressbook specification, such as cycles, shared addresses, and alias lookups that do not end in addresses. Additionally, the assertion did not make much sense in the phonebook phrasing. The assertion is "every alias must end in a valid address." A phonebook name listing that does not end in a phone number does not seem like a plausible specification mistake. However, a catalog with a missing page listing is a plausible specification issue. The issue of phrasings came up in many of our studies. Phrasing variations of logical statements is known to have an impact on user performance [6]. We fully discuss this finding in results chapter 6.

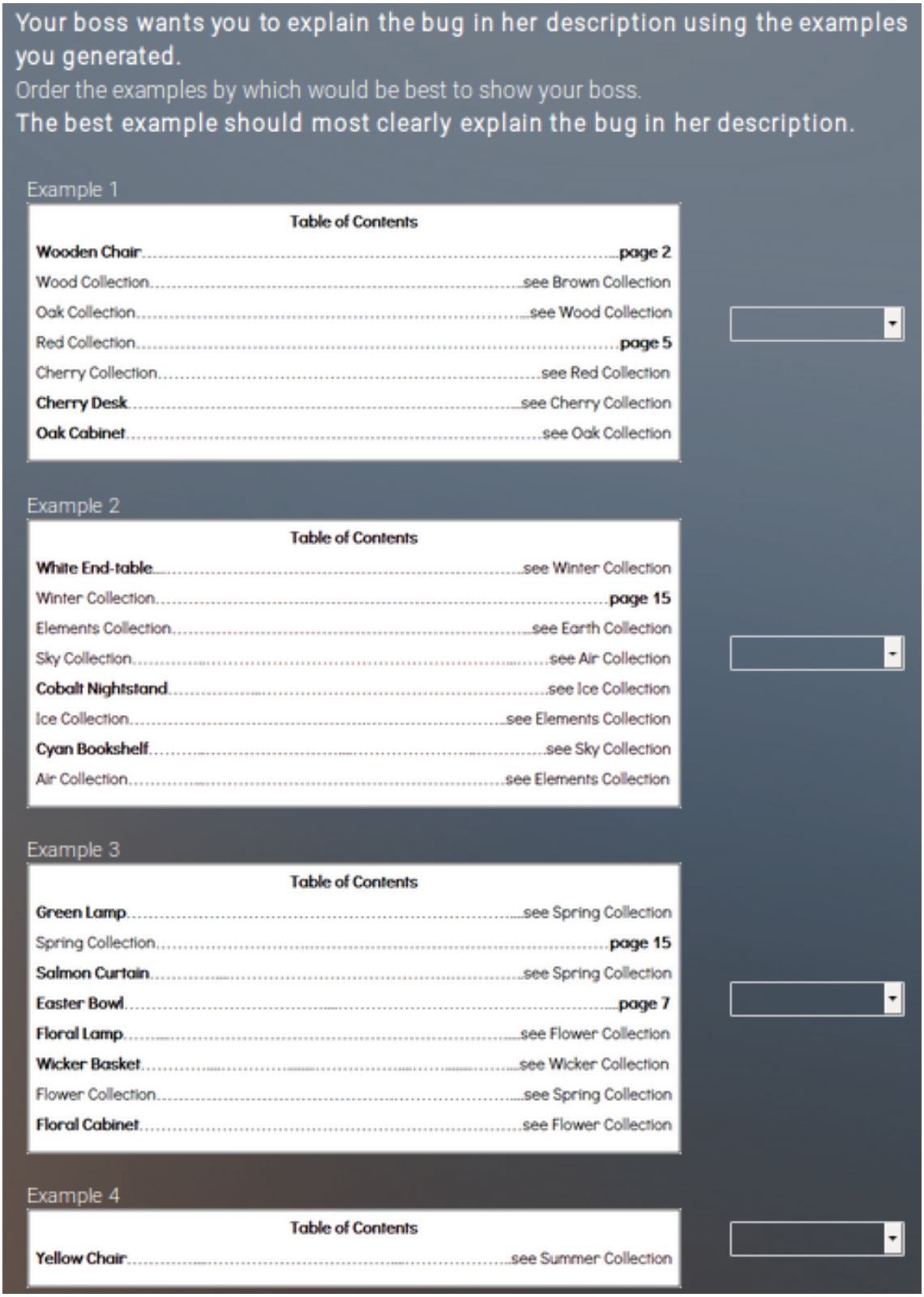


Figure 4.10: Catalog Models

We also changed the models to all be counterexamples varying in size, instead of models exhibiting properties that may or may not be possible according to the specification. This change was done to keep the models consistent with the newly added assertion, as well as directly evaluate preference for particular model sizes, such as minimal models. The resulting models can be seen in figure 4.10. Each model varies in size, the number of bugs present, and the size of the bugs. Bugs are defined as portions of the model exhibiting a failure of the assertion. In this case, a bug is an item lookup that does not end in a page number. The number of failing lookups and the size of the failing lookups varies for each model presented. The categorization of each model can be seen in the results discussion.

Model Size	Model Problem Type	Round 1	Round 2	Round 3
Medium	1 Medium Bug	16	n/a	n/a
Large	2 Large Bugs	35	44	66
Large	1 Minimal Bug	25	30	n/a
Minimal	1 Minimal Bug	53	55	63

Table 4.5: Guided User Preference

Table 4.5 shows the ranked preference data run through an Instant Runoff Voting simulation. In each round, the contending models had their top votes counted. The least preferred model was removed from the running. Then, the ranks are changed based on the missing model. The top votes are then recalculated for each model until a winner is chosen. Each round was tested for rank differences using a Friedman test. The first two rounds achieved a $p < .05$. However, the difference in ranks for round 3 are not significant. The large model 1 and the minimal model 3 are essentially tied in preference. This can be seen by the minuscule difference in votes for round 3 in the table.

Chapter 5

Results: First-Order Information Translation and Syntax

Our results have uncovered two factors that can be obstacles for users of formal methods tools. First, excessively informal translations from mathematical specification to natural language cause confusion. We suggest that without full syntactic representation in the translation, useful keywords are lost, and the reader loses grasp of the big picture. Second, inconsistent syntax between the specification and model translations obfuscates the connections between the two. For instance, if a relation is translated to one natural language fragment in the specification, but a different fragment in the model, the user will not know those two phrases are talking about the same thing. Together, these problems cause both global and local information loss.

Our earlier crowd-sourced studies assumed that any level of formality would be too burdensome for non-experts to handle. However, as our results in sections 5.1, 5.2, and 5.3 show, the informality yielded wholly inconsistent results; that is, the equivalent to statistically random data. These results suggested that even non-

expert participants benefit from tasks that avoid ambiguity. This finding motivated our approach in section 5.2.

Beyond decreasing ambiguity, the results suggested another option for mitigating noise in these studies: to improve the interface participants use to take the experiment. As we increased formalism while paying careful attention to HCI concerns, the variance in the results began to stabilize. We arrived at a translation method to plain English that produces consistent enough to measure effectively. The method is a two-step pipeline. Step one mechanically translates from first-order logic to pseudo-natural-language: a set of sentences consisting of natural language fragments that are not grammatically correct. Step two manually massages out any grammatical inconsistencies, without destroying the existing syntax. The translation pipeline is evaluated in chapter 7, and is one of two major contributions to future studies.

5.1 Initial Crowd-sourced Preference Study

In our original preference study on Mechanical Turk, we measured non-experts' ability to analyze models. Both the models and the addressbook specification were presented as informal English translations. As shown in figure 6.1, most respondents could not describe abstract properties the models exhibited, even when given a concrete list of properties to choose from. For instance, example 3 showed a translated model with two person listings having the same phone number. Very few respondents provided a similar description to the last sentence. A majority of the respondents also did not indicate, via a yes/no questions, that they learned something about the specification from looking at the example.

The following table shows the mean positive response for each question category for the examples (see section A.2 for detailed descriptions). The first two examples

only asked whether the model met the specification requirements or not. The last four examples only asked if anything was learned about the specification due to it. Each of the free-form responses were graded leniently by a single coder. A proper description is described as loosely explaining the situation exhibited in the example.

Example #	Correct SAT Answer	Possibly Learned	Proper Description
Generally	**High**	**Low**	**Very Low**
1	90%	-	-
2	90%	-	-
3	-	27.5%	12.5%
4	-	32.5%	12.5%
5	-	22.5%	17.5%
6	-	50%	2.5%

Table 5.1: Analysis & Property Learning Ability

With just this study, there are several possible explanations for this. One is that the participants just did not understand the examples. However, even without explicit instructions, an overwhelming majority of respondents could figure out whether model 1 or 2 satisfied the specification. Another explanation is the lack of explicit instructions or assertion to verify. This is the refinement we made in our next study.

5.2 Followup Crowd-sourced Preference Study

In the follow-up study, respondents were given an explicit assertion to check, to address the participants' inability to describe abstract properties. They had to analyze models as well as isolate the problem in the specification. The specification had 3

possible sentences that originated the counterexamples presented. The respondents chose one of those lines, and then explained their choice. An explanation was considered correct if it mentioned the empty alias issue in the addressbook specification. In this case, a perfect response mentioned that the 3rd sentence used a qualifying "may" which enabled aliases to be empty. An acceptable response just had to mention that aliases referenced can not be missing. A wrong response failed to mention anything regarding missing aliases.

In the following table, the average subject ability to properly pick apart examples is displayed. Also included are structural information about each model. The size describes the number of facts in the model, whereas the the type loosely describes the bugs the subjects were meant to highlight. In general, more bugs, or larger bugs, means the respondent had to highlight more of the example.

Model	Size	Type	Mean Proper Model Highlighting
Teaching Example	Medium	1 Medium Bug	72%
Model 1	Large	2 Large Bugs	82%
Model 2	Large	1 Minimal Bug	77%
Model 3	Minimal	1 Minimal Bug	90%

Table 5.2: Counterexample Comprehension Rates

The next table describes the specification refinement portion of the study. This binomial table shows the number of users that did/did not select the problematic portion of the specification and also gave a good/bad explanation of the problem. Again, the explanations were graded leniently by one coder. Any free-form response that even remotely referenced parts of the problem were given a good score.

	Good Explanation of bug	Bad Explanation of bug
Marked the buggy line	80	44
Did not mark the buggy line	6	70

Table 5.3: Model to specification Abstraction Ability

Figure 5.3 shows a binomial table comparing their multiple choice answer to the quality of their explanation. The explanations were graded as described above. Most respondents who failed to isolate the bug gave poor explanations. However, even those who did choose the correct line only had a 65% chance to give a decent explanation, even with leniency in grading. The informality of the presentation exhibits a disconnect between the correct responses, and level of understanding.

5.3 Initial Crowd-sourced Performance Study

Even after making the specification, assertion, and models more precise in the first performance study, the informal approached yielded inconsistent results. Respondents were shown a single model which they analyzed, and then were asked to choose a specification fix from four possible options. We conducted a between subjects study on the addressbook specification using 4 different model types and 3 different phrasings.

Model Type	Catalog	Webpage	Mail	Averaged
Minimal	62.5%	40%	44%	48.98%
Larger Context	44%	18.75%	57.89%	40.36%
Multiple Bugs	50%	23.07%	69.5%	47.55%
Valid and Buggy	43.75%	38.89%	54.17%	45.61%

Table 5.4: Addressbook Across Phrasings

The percentage of correct fixes chosen for each equal sample is shown in table 6.2. Looking at the averaged percent of correct responses across phrasings, we see that respondents had a 50:50 chance of answering the question correct, regardless of which model they were presented. Statistically speaking, these results are no better than a random generation of data. While there are multiple possible causes at this point, we decided to address informality, and redid this study with a significant increase in formalism. As reported in the next section, the more formal approach was a success.

5.4 Followup Crowd-sourced Performance Study

In the followup study, we presented the same specifications comparing minimal and non-minimal models, except with a formal translation and interaction environment. We attempted to gather data within subjects. To elaborate, each respondent was given the same specification twice: once with a minimal models, once with control models. In order to present the same specification again, we rephrased the specification for one of the tasks. One possible issue with rephrasing the specification is that participants will recognize the variation. However, in the informal studies, we saw phrasing confusing the respondents. We hoped this effect would carry over and

obfuscate the similarities in the task.

The next table shows the within-subjects data for each specification task. Different groups were exposed to the two possible orderings, and phrasings of each task: resulting in four total permutations per task. To clarify, the first row shows results for subjects given the addressbook specification, with the rephrased control task first, and the original phrased minimal task second. The difference of the mean and median number of attempts is taken from the same subject sample. That is, the within subject data for both the control and minimal task were subtracted from each other. Positive values indicate the control attempts were greater than the minimal attempts For example, in the first row, both the mean and median are positive. This means that users required fewer guesses to reach the correct answer with minimal models.

Surprisingly, the phrasing did not obfuscate anything. As shown in table 6.3, the respondents appeared to recognize the specification the second time around,perhaps due to a decrease in the differential when the control is shown second. In each study, grouped by specification, being presented second has a significant impact on the difference in attempts needed to finish the task. Put simply, the second time a specification is presented, the respondent needs many less attempts, regardless of the model variant. The syntactic confusion and inconsistency that existed in the previous informal studies is no longer present, because of the ordering effect. To elaborate, if participants can recognize a rephrased specification the second time and solve the task in less attempts, they recognized the syntactic equivalence between the phrasings. That, along with the rest of the results, suggests informality was indeed the cause of the confusion and inconsistent results.

specification	Shown Second	Phrasing Kept	Median C-M	Mean C-M
Addressbook	Minimal	Minimal	2	1.172
Addressbook	Control	Minimal	0	.546
Addressbook	Minimal	Control	.5	.16
Addressbook	Control	Control	1	.077
Gradebook	Minimal	Minimal	1	.484
Gradebook	Control	Minimal	0	-.16
Gradebook	Minimal	Control	1	.828
Gradebook	Control	Control	-1	-.784
Bad Worker	Minimal	Minimal	3	1.114
Bad Worker	Control	Minimal	-2	-.857
Bad Worker	Minimal	Control	2.5	.584
Bad Worker	Control	Control	-2	-.857
Other Groups	Minimal	Minimal	1	.286
Other Groups	Control	Minimal	-2	-.1571
Other Groups	Minimal	Control	2	1.139
Other Groups	Control	Control	0	-1.314

Table 5.5: Within-Subjects Differential; "C-M" means "Control minus Minimal"

Chapter 6

Results: Translation Phrasing Semantics and User Intuition

Some phrasings of the translated specification combines poorly with the users pre-conceived knowledge. The phrasing effects how the respondents approach their problem solving. Experts understand the difference between the English phrasing used, and the logical structure of the specification. However, non-experts do not understand the difference, or even realize that there is an inherent mathematical structure underneath. Instead, we suggest they approach these tasks as everyday word problems, bringing their real world intuition into every response. This can have quite an insidious effect on results, regardless of the formality in presentation.

Throughout the studies, the chosen real world semantics of the first order specification have been an issue, especially with non-expert samples. We stopped using the phonebook phrasing of the addressbook specification after the study in section 6.1 because it was generally confusing. In a later study discussed in section 6.2, we measured several phrasings and found them to have a much larger effect than the models being presented. Even after fixing the translation formality discussed

in chapter 5, phrasing had a huge impact on our results, as reported in section 6.3. While we do not have a convenient solution to this problem, we contribute that phrasing should be treated delicately and reported as a part of bias meta-analysis.

6.1 Initial Crowd-sourced Preference Study

In our first Mechanical Turk study, we gave 40 respondents a specification, and some examples to analyze. Even without an explicit assertion, and a fairly informal specification, they had some notion of satisfiablity. The specification was the addressbook phrased as an everyday phonebook, and was presented as such. For the first two models, the respondents had to answer whether the model satisfied the specification or not.

Example #	Correct SAT Answer	Possibly Learned	Proper Description
1	90%	n/a	n/a
2	90%	n/a	n/a
3	n/a	27.5%	12.5%
4	n/a	32.5%	12.5%
5	n/a	22.5%	17.5%
6	n/a	50%	2.5%

Table 6.1: Analysis & Property Learning Ability

The first model satisfied the specification, while the second one did not. In both cases, 36 of the 40 respondents correctly marked the models. However, their responses to the rest of the questions, as detailed in chapter 5, shows that they did not respond according to just the spec. They answered the question based on their understanding of a real world phonebook, not the specification presented.

6.2 Initial Crowd-sourced Performance Study

In the first performance study, phrasing was the only discernible effect measured in the data from thousands of Mechanical Turkers. Rephrasing the specification into a different real world scenario had a huge impact on user performance in this study. A particular phrasing caused some respondents to bring in their own thoughts on what they think is right and wrong, regardless of the specification and assertion presented.

The following table illustrates the population percentage that properly identified the problem in the specification, for the particular model type and phrasing they were exposed to. The minimal model type is as defined in section 2.2. The others are various non-minimal models that were roughly categorized by the unnecessary facts they contained. Their classification is not relevant to the data being reported. We gave separate populations three different phrasings of the addressbook: a catalog table of contents, webpage with links, and a postal mail system.

Model Type	Catalog	Webpage	Mail	Averaged
Minimal	62.5%	40%	44%	**48.98%**
Larger Context	44%	18.75%	57.89%	**40.36%**
Multiple Bugs	50%	23.07%	69.5%	**47.55%**
Valid and Buggy	43.75%	38.89%	54.17%	**45.61%**

Table 6.2: Addressbook Study Across Phrasings

Each column dictates a different rephrasing of the addressbook specification. The final column is simply an average of the other three columns. For instance, row one states that 62.5% of participants properly identified the problem in the specification for the catalog rephrasing of the addressbook translation. This can be

further compared to the 40% correctness for the webpage phrasing, and the 44% of the mail phrasing. The average correctness given a minimal model was 48.98%.

For each phrasing, the real world scenario had a drastic impact on how the non-expert perceived the information. For example, in the mail phrasing, empty addressbook aliases were translated as missing post offices. To a non-expert, this failure of the assertion is much more alarming in the mail phrasing compared to the other phrasings. Furthermore, the different scenarios exhibited no effect on the results, as seen in the final column. The final column averages across phrasings, which accounts for any biasing effect introduced by any particular phrasing. Accounting for the phrasing effect shows that each group of participants had a mean correct response of 50%. The phrasing introduced so much noise that the results without this effect are equivalent to flipping a coin.

6.3 Followup Crowd-sourced Performance Study

Even with increased formalism, phrasing has significant effects on user performance. In the followup performance study, hundreds of Mechanical Turkers were given a specification, assertion, models of a certain type, and had to pinpoint the part of the spec causing the assertion to fail. Even with the formal translation, and the respondents recognizing rephrases within the same study, phrasing had an impact on the number of attempts it took to trace the problem. Additionally, the effect was more intense the easier the task was. Formality did not appear to help many respondents use the models, because their response time suggests they only read the specification quickly and began to answer.

Table 6.3 shows the within-subjects grouped data for each specification task. Different groups were exposed to the two possible orderings, and phrasings of each

task: resulting in four total permutations per task. To clarify, the first row shows results for subjects given the addressbook specification, with the rephrased control task first, and the original phrased minimal task second. The difference of the mean and median number of attempts is taken from the same subject sample. That is, the within subject data for both the control and minimal task were subtracted from each other. Positive values indicate the control attempts were greater than the minimal attempts.

Spec	Shown Second	Phrasing Kept	Median C-M	Mean C-M
Addressbook	Minimal	Minimal	2	1.172
Addressbook	Control	Minimal	0	.546
Addressbook	Minimal	Control	.5	.16
Addressbook	Control	Control	1	.077
Gradebook	Minimal	Minimal	1	.484
Gradebook	Control	Minimal	0	-.16
Gradebook	Minimal	Control	1	.828
Gradebook	Control	Control	-1	-.784
Bad Worker	Minimal	Minimal	3	1.114
Bad Worker	Control	Minimal	-2	-.857
Bad Worker	Minimal	Control	2.5	.584
Bad Worker	Control	Control	-2	-.857
Other Groups	Minimal	Minimal	1	.286
Other Groups	Control	Minimal	-2	-.1571
Other Groups	Minimal	Control	2	1.139
Other Groups	Control	Control	0	-1.314

Table 6.3: Within-Subjects Differential; "C-M" means "Control minus Minimal"

Looking at the even and odd pairs of rows in each specification, one can see the drastic disparity in the difference in attempts for only a change in phrasing. The specifications are also sorted by difficulty, from hardest (Addressbook) to easiest (Other Groups). As the difficulty increases, the consistent favoring of minimality increases. The phrasing impact only seems to obfuscate the true effects of the independent variables for easier tasks.

The following table is merely a restructuring of the data above. Instead of comparing the first task given to a single subject to the second task given, we removed the second task from the data set. This was removed due to the ordering effect discussed in chapter 5. Then, the data is treated as unpaired groups of task results between subjects. That is, each subject only contributed one task result to each category.

Specification	Model Type	Unit	Mean	Median
Addressbook	Control	# Attempts	4.375	5
Addressbook	Minimal	# Attempts	4.154	5
Addressbook	Control	Seconds Taken	82	56
Addressbook	Minimal	Seconds Taken	88	69
Gradebook	Control	# Attempts	3.594	3
Gradebook	Minimal	# Attempts	2.492	2
Gradebook	Control	Seconds Taken	97	58
Gradebook	Minimal	Seconds Taken	93	81

Table 6.4: Between-Subjects Performance Results

These tasks are being solved in about a minute and a half, as shown in table 7.1. Gradebook, the easier task, takes a reasonable number of attempts to solve

in this time, while the harder task, Addressbook, takes almost twice the attempts. Seeing as the respondents took the same amount of time, suggests they are not carefully reading the models and using that information to solve the problem. The gradebook could be solved reasonably because of the intuitive property of the bug. While addressbook, a more mathematical structured specification, was difficult to understand at first glance, causing the respondents to almost exhaust the possible options.

Chapter 7

Evaluation: Study Framework and Minimality Performance

The results chapters reported issues with our earlier studies that we claim to have addressed in our final study iteration. Chapter 5 reported that excessively informal translations from mathematical specification to natural language caused confusion, and inconsistent syntax between the specification and model translations obfuscates the connections between the two. Our final study uses a two-stage translation pipeline driven by these findings to address the syntax issue. Chapter 6 reported that phrasing of the specification combines poorly with the users' preconceived knowledge. While we did not offer a general solution to this problem, the final study avoids the semantic issue by using the original spec phrasing. This is a bias, and our study would have been conducted with multiple phrasings if we achieved significant results with large effect sizes in our final user evaluation. Our final user evaluation is discussed in section 7.1. Whether our final user evaluation truly addressed the issues reported in the results is discussed in section 7.2.

7.1 Minimality Performance

We conducted two final performance studies. One with crowd-sourced users given our translated model-finder facsimile. The other with experts given our model-finder facsimile, sans translations.

7.1.1 Followup Crowd-sourced Performance Study

We surveyed hundreds of Mechanical Turkers to measure the effectiveness of particular model types for refining specifications. They were given a specification, an assertion that is failing, a set of counterexamples of a particular model type, and some possible problem portions of the specification to isolate. Unlike the previous surveys, we formalized the translation from first order relational logic to plain English. The results for both specifications are reported in the following table.

The following table is merely a restructuring of the data above. Instead of comparing the first task given to a single subject to the second task given, we removed the second task from the data set. This was removed for reasons described in the results chapters. Then, the data is treated as unpaired groups of task results between subjects. That is, each subject only contributed one task result to each category.

Specification	Model Type	Unit	Mean	Median
Addressbook	Control	# Attempts	4.375	5
Addressbook	Minimal	# Attempts	4.154	5
Addressbook	Control	Seconds Taken	82	56
Addressbook	Minimal	Seconds Taken	88	69
Gradebook	Control	# Attempts	3.594	3
Gradebook	Minimal	# Attempts	2.492	2
Gradebook	Control	Seconds Taken	97	58
Gradebook	Minimal	Seconds Taken	93	81

Table 7.1: Between-Subjects Performance Results

For both tasks, users took less attempts on average to trace the problem in the specification when given minimal models. The mean difference was not significant for the addressbook task. However, the mean and median difference for the gradebook task was significant. A Shapiro-wilk test was performed to confirm that the data is non-normal. Then, a Mann-Whitney's U test was performed to test the difference between the two unpaired minimal and control groups. The difference in the number of attempts to trace the problem in the specification is significant to 99.8%, with a medium effect size of 0.275.

7.1.2 Expert Performance Study

We surveyed two dozen Brown undergraduates highly familiar with Alloy. We sought to measure their ability to trace and fix problems in specifications when given an unaltered refinement tasks. They were also given a specification, an assertion that is failing, a set of counterexamples of a particular model type. However, they were

allowed to freely edit the specification, as they did not require a simplified presentation format due to their experience.

This table is the dual to the between-subjects data from the previous study. Each expert only saw one task each, therefore the results are between subjects. Again, the mean and median attempts used / time taken to fix the specification are reported.

Specification	Model Type	Unit	Mean	Median
Addressbook	Control	# Attempts	2.765	2
Addressbook	Minimal	# Attempts	2.867	3
Addressbook	Control	Seconds Taken	2462	410
Addressbook	Minimal	Seconds Taken	4966	295
Gradebook	Control	# Attempts	2.294	2
Gradebook	Minimal	# Attempts	2.267	2
Gradebook	Control	Seconds Taken	5608	347
Gradebook	Minimal	Seconds Taken	340	387

Table 7.2: Expert Performance Results

While there are some small differences in the number of attempts, none of them turned out to be significant. Even the more notable differences in time taken to complete the task are statistically insignificant. This is an exemplary case of a small sample size producing insignificant trends- the very reason we began developing this framework.

7.2 Practical Study Framework

In the previous section, we reported results from our final two performance studies: one crowd-sourced, one with experts. The experiments are exactly the same, except the crowd-sourced one is translated, and only asks the user to trace the problem, not also fix it. The variances between the minimal and control groups for each spec and study are reported in the following table.

Specification	Subjects Type	Unit	Mean Variance	Median Variance
Addressbook	Crowd	# Attempts	0.024	0
Addressbook	Expert	# Attempts	0.005	0.5
Addressbook	Crowd	Seconds Taken	18	84.5
Addressbook	Expert	Seconds Taken	3135008	6612.5
Gradebook	Crowd	# Attempts	0.607	0.5
Gradebook	Expert	# Attempts	0.0004	0
Gradebook	Crowd	Seconds Taken	8	264.5
Gradebook	Expert	Seconds Taken	13875912	800

Table 7.3: Cross-referenced Performance Results

Looking at the mean and median attempt variances between the crowd-sourced and expert groups, the difference between them is quite small, and statistically insignificant. The difference in time variances is quite large, but this is to be expected due to the differences in expertise and study design. The crowd-sourced respondents only had to trace the problem, not fix it. Fixing the problem and parsing the unaltered first-order information takes longer than solving an above average word problem logic puzzle. The minimal differences between the non-expert and expert results suggests the reported effects to handle, and final study design is indeed a

trustworthy starting point to conducting a future crowd-sourced model-finding user evaluation.

Chapter 8

Related Work

We are unaware of any previous user evaluations of model-finding research. In this section we provide context for the two main areas our research has drawn upon: formal methods and user studies, and also point to some aspects of human-computer interaction that influenced our work.

8.1 Formal Methods

8.1.1 Model Finding

Many model finders rely on SAT/SMT solving techniques. These tools, also known as MACE-style model finders [24], instantiate and flatten the original first-order specifications into propositional logic problems. Satisfiability (SAT) solvers are specialized tools designed to efficiently solve boolean logic problems [2]. SAT modulo theory (SMT) solvers extend SAT solving techniques past boolean variables, such as linear arithmetic, equality, and uninterpreted functions [28]. Both of our previous minimal model-finders are MACE-style. Aluminum is a modification of Alloy, which implements a minimization algorithm on standard Alloy models [29]. Razor

is a standalone minimal model finder, that supports exploration of all models, and provenance, which originates facts in the specification [34].

Our minimal model finders rely on several formal methods techniques. Logic programming languages restricted to horn-clauses innately produce single, least models [22]. Our Razor model-finder operates over first-order formulas in geometric form. Geometric form is an expressive equivalent to first-order logic, that consists of universally quantified horn-clauses, with disjunction and existential quantification allowed in the consequent. A more general definition of minimality is used for the semantics of non-monotonic reasoning [33] and database updates [11]. Razor relies on a database update algorithm called the Chase. A modified version of the Chase constructs the Herbrand Base, which is used to instantiate the first-order clauses down to propositional logic. Minimal model generation often relies on tableaux [30], or hyper-resolution techniques [4]. The original Chase algorithm is similar to a tableaux method. An earlier version of Razor used this decision procedure to find models without implementing MACE-style solving.

Other model-finders have produced minimal models under various definitions of minimality. Janota's algorithm generates all minimal models given a single model [19]. The first minimal model is produced by modifying a traditional SAT solver. Koshimura et al. compute minimal models in propositional logic to solve job-scheduling problems efficiently [21]. The Cryptographic Protocol Shapes Analyzer [10] produces minimal models specifically for analyzing crypto-protocols. These works generate variations minimal models, or propose slightly different definitions of minimality. We are not concerned with user evaluating these different minimality definitions at the moment.

Minimality is not the only option for which models to generate. Fu and Malik develop efficient algorithms for generating maximal propositional models [12]. Cunha,

Macedo, and Guimaraes implement a target oriented model finder [8]. They define a distance between models in order to generate next models near or far from the current target model. Both of these works have their own mathematical benefits. However, this work is only concerned with evaluating minimal models.

8.1.2 Theorem Proving

We are only aware of one other published user evaluation of a formal methods tool, which was conducted while we were also researching this topic. Hentschel, Hahnle, and Bubel conduct a user study to evaluate two different interfaces for KeY, a program verifier [17]. The old control interface presents the user a debugger which focuses on theorem proof objects from the program verification. The new interface provides an interactive, symbolic execution debugger. They performed a user evaluation with 32 experts with various levels of experience with the tools and languages involved. They found that the less experienced users performed significantly better using the new debugging interface for a variety of tasks. Their user study evaluates a theorem proving tool, instead of a model finder. Also, they used a small sample of experts, and found significant positive trends in two of their five hypotheses. Their work only evaluates two interfaces, as well: this is something the HCI community has been evaluating for decades. Our work evaluates a more theoretically complex concept, minimal models. Their work provides a well detailed reference for how to perform a user evaluation consisting of only expert participants. Our work is primarily concerned with surveying non-experts, to enable more ambitious studies requiring larger sample sizes. The authors only report significance values during their null-hypothesis testing. Cumming [7] urges researchers to move away from just null-hypothesis significance testing, as it does not fully discuss bias and variance, which is important for replication. Significance is a good litmus test for

success, but should be further investigated with confidence intervals, probability models, and effect sizes. We touch on this meta-analysis through effect size, but we need to increase this in our future studies. However, since most of our results are currently insignificant, meta-analysis is mostly moot.

8.2 Crowd-sourcing

8.2.1 Amazon Mechanical Turk

We investigated the challenges of using Mechanical Turk for crowd-sourcing before sending out our user studies. Kittur, Chi, and Suh discuss the trade-offs of sample sizes, cost, and quality responses [20]. They stress that Mechanical Turk by design enables large sampling at low cost, but designing the task requires careful attention in order to get quality responses. In order to pay the needed attention, we collected meta-data in our studies to refine our Mechanical Turk task decisions. We spent non-trivial effort making our studies not frustrating to finish, avoiding adding distracting questions, and paid our respondents a living wage. We also developed an informal adversarial model, to reduce the number and filter out the remaining low quality responses. Peer, Vosgerau, and Acquisti futher investigate MTurk's quality [32]. They evaluate Amazon's built in quality controls, and conclude that reputation and productivity do correlate with response quality. Therefore, we restricted our studies to mechanical turkers with thousands of completed tasks, with high approval rates. Mason et al [23] evaluate the differences in the expert and crowd-sourced populations, while also providing a blueprint on how to properly conduct studies on mechanical turk properly. In their opinion, Amazon's crowd-sourcing tool a great resource when considering all categories (cost, stability of responses, quality, speed of feedback, etc.). They were one of the reasons we decided on mechanical turk and a

general resource through experiments. Gould et al [14] discuss the difficult of keeping crowd-source subjects attention. They find that without intervention, crowd-workers reach inattention after about 5 minutes. Additionally, they implement and evaluate their own intervention method that encourages focus whenever the worker begins to multitask. Their work incentivized us to keep our tasks as simple and short as possible. However, their intervention schemes were not very applicable because we have a single task, and it is difficult to measure inattention in our context.

8.2.2 Visualization Community Adoption

The vis community did not always conduct user studies, and only recently started using crowd-sourcing. Heer and Bostock [16] successfully replicate and confirm classic user evaluations from other disciplines using Amazon's Mechanical Turk. Notably Cleveland and McGill's [5] pioneer user study. Not only are we conducting one of the first formal methods user evaluations, we are also using crowd-sourcing. We are simultaneously creating and replicating history. The mentioned papers give us hope that both of these challenges can be overcome.

8.2.3 Past User Evaluations

The HCI community has begun using crowd-sourcing for much more than user-evaluations. Bernstein et al [1] wrote and edited papers using crowd-sourced workers. Bigham et al [3] develop VizWiz, which enables blind users to get crowd-sourced feedback on their current environment in real time. After crowd-sourced user evaluations have become a part of the formal methods community, it may be possible to bring the human further into the loop, enabling advancements such as these.

8.3 Human Computer Interaction and Visualization

We borrow several well founded techniques in the development of our model finder facsimile. Ghoniem, Fekete, and Castagliola [13] show that graphs larger than twenty vertices are better represented as a matrix. We avoided presenting non-expert Alloy graphs for this reason. Simons [35] remarks on dealing with the change-blindness in modern studies. We switched from multi-page, dynamically changing layouts to our current single-page, static layout to address this cognitive effect. Munzner [27] acts as a general aid to designing graphical user interfaces. We used this as a general reference for less significant design decisions with the model-finder facsimile. Wills [37] states that linked text highlighting, or similar linked visualizations, increases the users chances of cognitively linking the information. This inspired our italicization of relations across the spec and model.

We also looked at the few non-expert targeted user interfaces for formal methods tools for inspiration. DeOrio and Bertacco develop a human-powered SAT solver [9]. SAT solving can be thought as propositional model finding. Their work details a method of posing this logically complex task to humans, in a way to optimize performance and engagement. Their use of shapes and highlighting to present boolean logic puzzles to users influenced many of our visualization choices. Moffitt et al. develop a multiplayer, crowdsourced model-checking game [25]. Model-checking is undecidable, so posing this problem to humans is even more complex than a SAT solver. They describe a method of breaking down this insurmountable problem into more accessible pieces, that can be distributed among participants. Unfortunately, model-checking is not the same as model-finding; it is closer related to theorem proving. Thus, many of their design decisions are not applicable.

We investigated visualization research pertaining to various logical models as well. Ottley et al. investigate the effect of textual, graphical, and tandem representations of probability models for Bayesian reasoning tasks [31]. They find that the lone textual representation marked the greatest impact on user's reasoning ability, despite the intuitive notion that visualizations are often better. Even more counter-intuitively, presenting both representations at once drastically decreases performance. This further pushed us away from using Alloy graphs in our model-finder facsimile whatsoever. The main caveat is to present the problem and the feedback model in a similar language, so users can easily make connections between the two. This inspired the matching syntax between the spec and model in our framework. As seen in our results comparing and contrasting levels of syntax matching, we come to the same conclusion as they did.

Chapter 9

Conclusion

As seen in our own user evaluation of minimality, acquiring a sample size large enough to measure effects with confidence is difficult with expert samples. We expanded the population to non-experts through crowd-sourcing. In some cases, there was a large benefit to presenting users minimal counterexamples, but not in all. We have identified several sources of variance that should be controlled in future user studies such as informal translations and chosen phrasings. Further mitigation of these variance sources will allow us to finish our minimality evaluation with confidence. Combating how to simplify both the syntactic and semantic information conveyed in relational first-order logic, we eventually developed a formal model finder facsimile. This survey tool produced results in line with the expert results participating in the same experiment, modulo translation.

9.1 Future Work

Currently the translated model finder facsimile only allows users to trace the problem in the specification, not fix it. We hope to develop an Alloy block language interaction to enable even non-experts to edit the specification. Additionally, the

crowd-sourced respondents took an undesirably short amount of time to read the task. Forcing interaction with the information to slow and direct their thinking should increase time taken, and improve the accuracy. Both of these should bridge the final gap in the results comparison, yielding almost equivalent data sets.

We believe that minimality should help for some user tasks and not help (or even hinder) others. Current results seem to align with our hypothesis. However, only one of the specifications was significant, and the effect size was not large. The above should enable us to produce a more robust, convincing set of results. Fully evaluating this hypothesis is our next step.

The most interesting study effect reported was the specification phrasing semantics combining with the user's preconceived intuition. This problem can be dodged by sticking to the original phrasing, but this is unsatisfactory. The logical structure is completely the same, so as long as the phrasing makes general sense, it should not affect user behavior. Confirming the source of the problem from the user's end is the first step. Being able to control user intuition for particular phrasings would be ideal in the investigation.

Bibliography

[1] M. S. Bernstein, G. Little, R. C. Miller, B. Hartmann, M. S. Ackerman, D. R. Karger, D. Crowell, and K. Panovich. Soylent: A Word Processor with a Crowd Inside. *the 23nd annual ACM symposium*, pages 313–322, 2010.

[2] A. Biere, M. Heule, and H. van Maaren. *Handbook of satisfiability*, volume 185. ios press, 2009.

[3] J. P. Bigham, C. Jayant, H. Ji, G. Little, A. Miller, R. C. Miller, R. Miller, A. Tatarowicz, B. White, S. White, et al. Vizwiz: nearly real-time answers to visual questions. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, pages 333–342. ACM, 2010.

[4] F. Bry and A. Yahya. Positive unit hyperresolution tableaux and their application to minimal model generation. *Journal of Automated Reasoning*, 25(1):35–82, 2000.

[5] W. S. Cleveland and R. McGill. Graphical perception: Theory, experimentation, and application to the development of graphical methods. *Journal of the American statistical association*, 79(387):531–554, 1984.

[6] L. Cosmides. The logic of social exchange: Has natural selection shaped how humans reason? studies with the wason selection task. *Cognition*, 31(3):187–276, 1989.

[7] G. Cumming. The new statistics why and how. *Psychological science*, page 0956797613504966, 2013.

[8] A. Cunha, N. Macedo, and T. Guimaraes. Target oriented relational model finding. In *FASE'14*, pages 17–31. Springer, 2014.

[9] A. DeOrio and V. Bertacco. Human computing for eda. In *Proceedings of the 46th annual design automation conference*, pages 621–622. ACM, 2009.

[10] S. F. Doghmi, J. D. Guttman, and F. J. Thayer. Searching for shapes in cryptographic protocols. In *Tools and Algorithms for the Construction and Analysis of Systems*, pages 523–537. Springer, 2007.

[11] R. Fagin, J. D. Ullman, and M. Y. Vardi. On the semantics of updates in databases. In *Proceedings of the 2nd ACM SIGACT-SIGMOD symposium on Principles of database systems*, pages 352–365. ACM, 1983.

[12] Z. Fu and S. Malik. On solving the partial max-sat problem. In *Theory and Applications of Satisfiability Testing-SAT 2006*, pages 252–265. Springer, 2006.

[13] M. Ghoniem, J.-D. Fekete, and P. Castagliola. A comparison of the readability of graphs using node-link and matrix-based representations. In *Information Visualization, 2004. INFOVIS 2004. IEEE Symposium on*, pages 17–24. Ieee, 2004.

[14] S. Gould, A. L. Cox, and D. P. Brumby. Diminished Control in Crowdsourcing: An Investigation of Crowdworker Multitasking Behavior. *ACM Transactions on Computer- ....*

[15] R. Hahnle. Is there a place for user experiments in the ar community? *AAR Newsletter*, 116, 2016.

[16] J. Heer and M. Bostock. Crowdsourcing Graphical Perception: Using Mechanical Turk to Assess Visualization Design. *Proceedings of the 28th international conference on Human factors in computing systems*, pages 203–212, 2010.

[17] M. Hentschel, R. Hahnle, and R. Bubel. An empirical evaluation of two user interfaces of an interactive program verifier. In *International Conference on Automated Software Engineering*, 2016.

[18] D. Jackson. *Software Abstractions: logic, language, and analysis*. MIT press, 2012.

[19] M. Janota. *SAT solving in interactive configuration*. PhD thesis, University College Dublin, 2010.

[20] A. Kittur, E. H. Chi, and B. Suh. Crowdsourcing user studies with mechanical turk. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 453–456, New York, NY, USA, 2008. ACM.

[21] M. Koshimura, H. Nabeshima, H. Fujita, and R. Hasegawa. Minimal model generation with respect to an atom set. *First-order Theorem Proving FTP 2009*, page 49, 2009.

[22] J. Lobo, J. Minker, and A. Rajasekar. *Foundations of Disjunctive Logic Programming*. MIT press, 1992.

[23] W. Mason and S. Suri. Conducting behavioral research on Amazon's Mechanical Turk. 44(1):1–23, 2011.

[24] W. McCune. Mace4 reference manual and guide. *arXiv preprint cs/0310055*, 2003.

[25] K. Moffitt, J. Ostwald, R. Watro, and E. Church. Making hard fun in crowd-sourced model checking–balancing crowd engagement and efficiency to maximize output in proof by games. In *CrowdSourcing in Software Engineering (CSI-SE), 2015 IEEE / ACM 2nd International Workshop on*, pages 30–31. IEEE, 2015.

[26] H. Motulsky. *Intuitive biostatistics: a nonmathematical guide to statistical thinking*. Oxford University Press, USA, 2014.

[27] T. Munzner. *Visualization Analysis and Design*. CRC Press, 2014.

[28] G. Nelson and D. C. Oppen. Simplification by cooperating decision procedures. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 1(2):245–257, 1979.

[29] T. Nelson, S. Saghafi, D. J. Dougherty, K. Fisler, and S. Krishnamurthi. Aluminum: principled scenario exploration through minimality. In *Proceedings of the 2013 International Conference on Software Engineering*, pages 232–241. IEEE Press, 2013.

[30] I. Niemelä. A tableau calculus for minimal model reasoning. In *Theorem Proving with Analytic Tableaux and Related Methods*, pages 278–294. Springer, 1996.

[31] A. Ottley, E. M. Peck, L. T. Harrison, D. Afergan, C. Ziemkiewicz, H. A. Taylor, P. K. Han, and R. Chang. Improving bayesian reasoning: the effects of phrasing, visualization, and spatial ability. *Visualization and Computer Graphics, IEEE Transactions on*, 22(1):529–538, 2016.

[32] E. Peer, J. Vosgerau, and A. Acquisti. Reputation as a sufficient condition for data quality on amazon mechanical turk. *Behavior Research Methods*, 46(4):1023–1031, 2014.

[33] A. Robinson and A. Voronkov. *Handbook of Automated Reasoning*, volume 1. Elsevier, 2001.

[34] S. Saghafi, R. Danas, and D. J. Dougherty. Exploring theories with a model-finding assistant. In *Automated Deduction-CADE-25*, pages 434–449. Springer, 2015.

[35] D. J. Simons. Current approaches to change blindness. *Visual cognition*, 7(1-3):1–15, 2000.

[36] D. Van Dalen. *Logic and structure*, volume 3. Springer, 1983.

[37] G. J. Wills. Visual exploration of large structured datasets. *Proceedings of New Techniques and Trends in Statistics (NTTS)*, pages 237–246, 1997.

Appendix A

Documentation: Studies and Data

This chapter provides full documentation of studies performed, and data not discussed in the results chapters. Each study is described in chronological order as follows.

- Section A.1: Expert Preference Study
- Section A.2: Initial Crowd-sourced Preference Study
- Section A.3: Followup Crowd-sourced Preference Study
- Section A.4: Initial Crowd-sourced Performance Study
- Section A.5: Followup Crowd-sourced Performance Study
- Section A.6: Expert Performance Study

The study descriptions explain the overall intent and design outside of the contributions reported in this paper.

A.1 Expert Preference Study

We gave a survey to 18 Brown graduate students, Brown faculty, and Alloy-B-Z (ABZ) 2012 conference attendees. The survey consisted of two parts: some short response questions, and an analysis of example specifications with their respective models. We intended to garner a sense of how experts in field use formal methods tools, and if they have any strong, possibly consistent opinions on models presented.

A.1.1 Design

Short Response Questions

Nine questions were given to the respondents. The first question asked which formal-methods tools they used (Alloy, B, Z, ASM, SAT solvers, model checkers, or others). The second question recorded demographics. Question three surveyed familiarity with Alloy, and other tools. The fourth question asked when and how often models were useful during tool use. The fifth question further gauged interaction with the models. Question six asked how often they think about necessity of facts in the models. The seventh question queried the amount of new knowledge provided by the model set. Question eight broadly requested favorite or desired features of the tools. The ninth question solicited additional comments.

Specification and Model Analysis

Respondents were given three specifications, each with a set of five models. The users reviewed a specification, chose which models they would prefer to see first, and explained their model choice. The specifications were presented by type hierarchy, relations, and factual constraints for brevity and quick comprehension. The addressbook specification is from the Alloy book [18], and describes a standard ad-

Short Questions

This survey uses the term **scenario** for a concrete example that witnesses the success or failure of a property or query. It includes instances, counterexamples, and models (from mathematical logic).

1. Which of the following formal-methods tools do you use? Check all that apply:

   ___ Alloy ___ ASM
   ___ B ___ SAT solvers (directly, not through tools like Alloy)
   ___ Z ___ Model checkers
   Other (please specify): _____

2. Which of the following best describe your current position?
   - ☐ *Masters student*
   - ☐ *PhD student*
   - ☐ *University faculty (professor, lecturer, etc.)*
   - ☐ *Industry or government employee*
   - ☐ *Other:* _____

3. For Alloy and other formal methods tools, which statement best describes your usage?

	Alloy	Other tools
I've never used it		
I've worked through small examples (e.g., from a tutorial or textbook)		
I've used it in class (as a student or professor), but not outside of class		
I do research on it, but don't model or analyze significant projects with it		
I've built other systems that incorporate it, but don't use it directly		
I've used it for a nontrivial part of a research project		
Other (please explain):		

4. How often do you find scenarios useful for each of the following tasks?

	Rarely	Sometimes	Often
Help find the source of a behavior or bug in a specification			
Suggest additional properties to check of a specification			
Suggest additional constraints required in a specification			
Identify unexpected relationships between parts of a specification			
Foster human confidence in the behavior of a specification			
Other (please describe):			

Figure A.1: Short Answer Questions

5. Imagine you use a tool to run a property or query against a specification. The tool produces a set of scenarios. How often do you use each of the following approaches to explore those scenarios?

	Rarely	Sometimes	Often
Trace each scenario through the specification before viewing another			
Inspect several scenarios at a high-level, then go back and trace (some of) their contents to the specification			
Refine query to include some fact(s) in a recently-viewed scenario			
*Ask a query that includes some fact(s) **not** seen in the scenarios so far*			
Ask a refined query that checks whether two distinct elements in an existing scenario could merge into a single element			
Edit the specification as soon as a scenario indicates an issue, then re-run the original query			
Other (please describe):			

6. When using a tool that produces scenarios in response to a query, how often do you wonder whether a particular fact or element in the scenario is *necessary* to satisfy the query?

Rarely Sometimes Often

7. When using a tool that produces scenarios, how often do the contents of one scenario suggest a new property or query that you hadn't previously identified?

Rarely Sometimes Often

8. What are your favorite or desired features in tool support for exploring scenarios?

9. Any additional comments on your usage of scenarios in formal analysis?

Thank you! If you'd be willing to respond to further questions about how you use scenarios, please continue to the next section and/or provide a contact email address:_____

Figure A.2: Short Answer Questions

	Alloy	Model Checkers	Z	SAT Solvers
# Familiar	11	4	2	1

Table A.1: Question 1 Results

	Undergraduate Student	Graduate Student	Academic Faculty	Industry Employee
# in Position	5	3	5	1

Table A.2: Question 2 Results

dress book. Our gradebook specification describes a grading policy similar to some universities. Our conference manager specification describes the roles in a conference paper review system.

A.1.2 Data

18 experts were surveyed. 12 were from ABZ 2012; 6 were from Brown 2015. The results reported in this section were statistically insignificant. The significant results are discussed in sections 3.3.1 and 4.1.

Familiar with	Alloy	Other Tools
Never Used	4	n/a
Small Examples	4	3
In Class	4	6
Trivial Research	2	0
As a Plugin	1	2
Nontrivial Research	6	4

Table A.3: Question 3 Results

	Often	Sometimes	Rarely
Trace each scenario through the specification before viewing another	5	5	4
Inspect several scenarios at a high-level, then go back and trace (some of) their contents to the specification	4	7	3
Refine query to include some fact(s) in a recently-viewed scenario	5	5	4
Ask a query that includes some fact(s) not seen in the scenarios so far	4	5	5
Ask a refined query that checks whether two distinct elements in an existing scenario could merge into a single element	2	1	10

Table A.4: Question 5 Results

	Often	Sometimes	Rarely
When using a tool that produces scenarios in response to a query, how often do you wonder whether a particular fact or element in the scenario is necessary to satisfy the query?	7	4	3
When using a tool that produces scenarios, how often do the contents of one scenario suggest a new property or query that you hadnt previously identified?	6	6	2

Table A.5: Question 6&7 Results

A.2 Initial Crowd-sourced Preference Study

We surveyed 40 amazon mechanical turkers. This preliminary survey explored non-expert ability to interact with specifications and models, while attempting to measure preference for particular model types.

A.2.1 Design

The respondents were given six examples, each meant to illustrate a particular aspect of the addressbook specification. Similarly to the previous study, we presented the specification as a set of facts and constraints. In order to make the technical information legible to this audience, we manually translated the factset into a few simple statements. The models were also presented in a simple, domain specific fashion. To accomplish this, the addressbook specification was rephrased as a phonebook. The models were "snippets from a phonebook page." In the first two examples, we gauged whether respondents could verify if an example satisfied the given specification. The first example was straight-forward, other than the name in the listings. This is possible as the addressbook spec allows for aliases. Example two is completely incongruent to the spec as there are addresses (phone numbers) as aliases to look up. The following four examples were explicitly marked as satisfying the specification. We explored how the respondents perceived the specifications, models, and the information they convey. Specifically, we were concerned if non-experts had a sense of abstract properties of the spec, and which models exhibited them. Once the respondents thoroughly analyzed the examples, they were asked which models were best to present. We phrased this as "showing interesting ex-

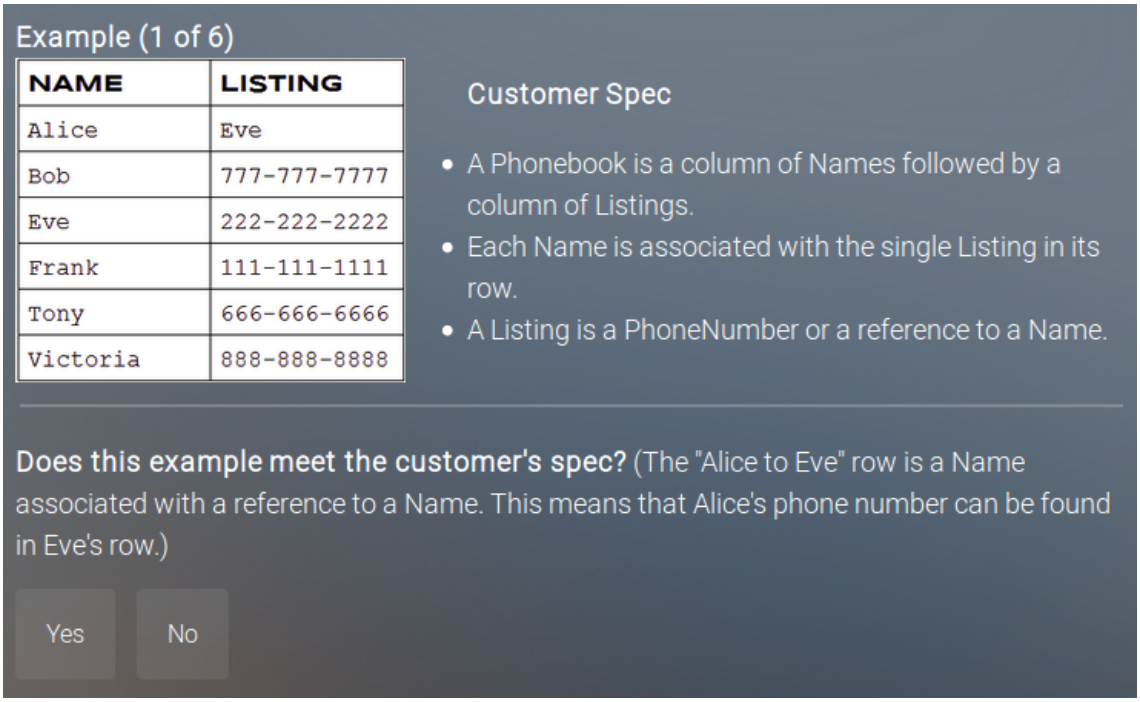


Figure A.3: Example 1

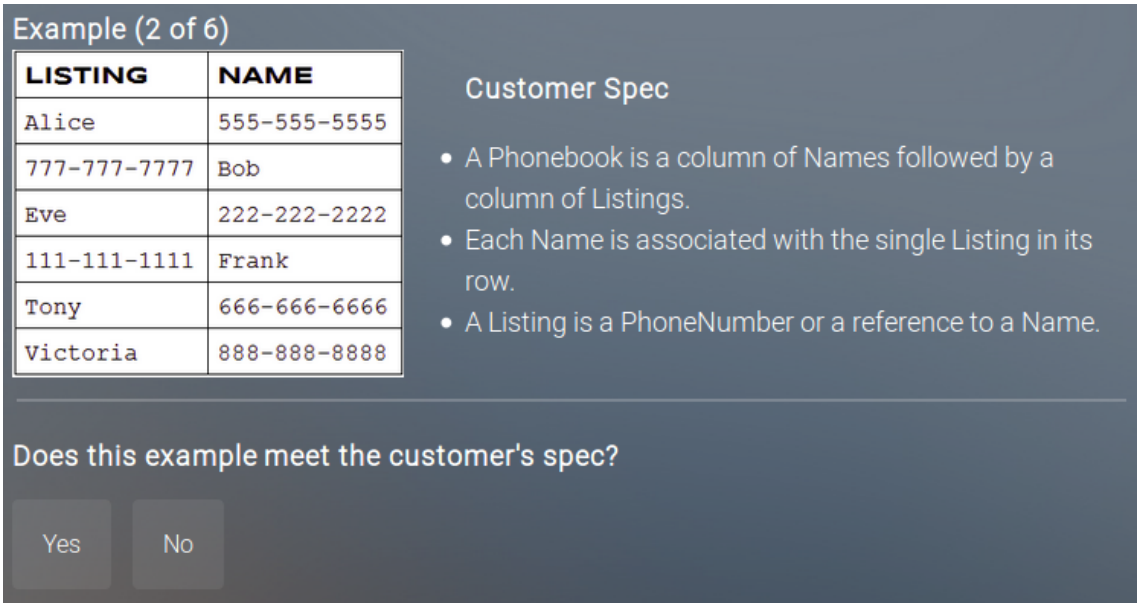


Figure A.4: Example 2

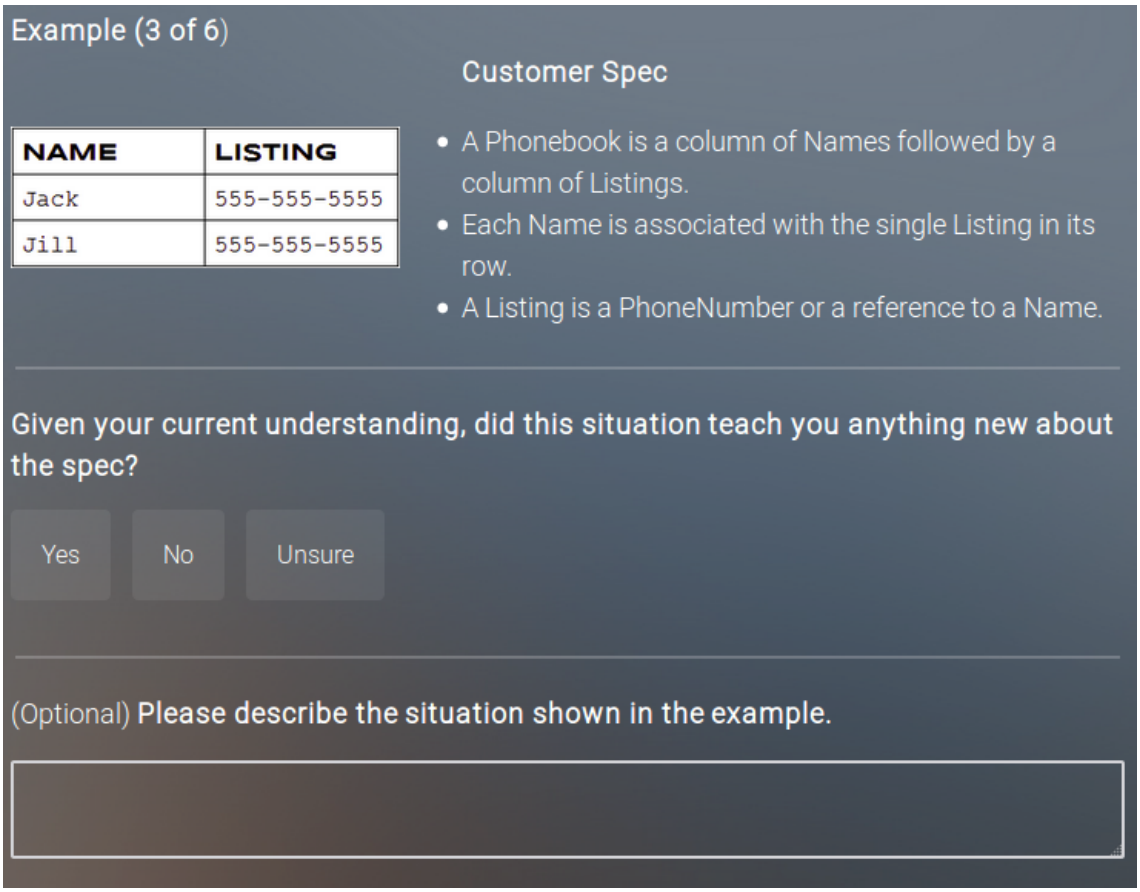


Figure A.5: Example 3

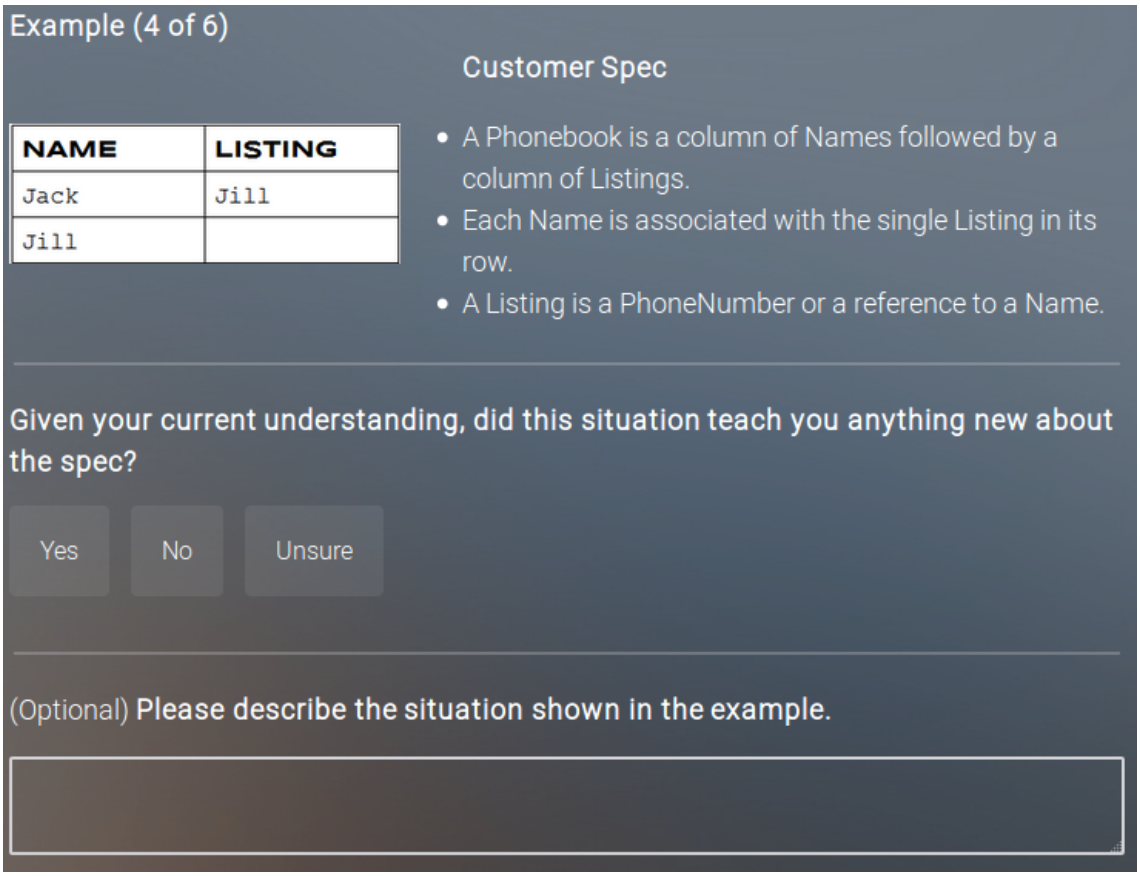


Figure A.6: Example 4

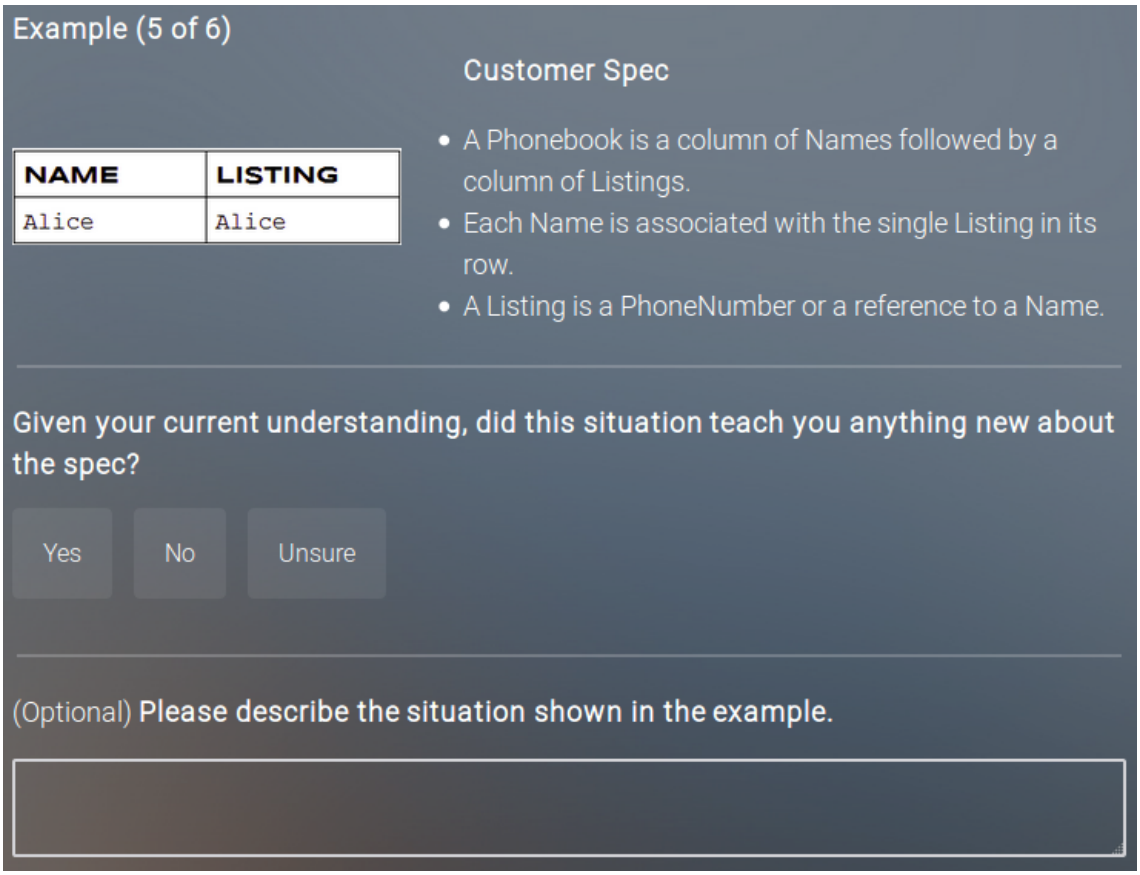


Figure A.7: Example 5

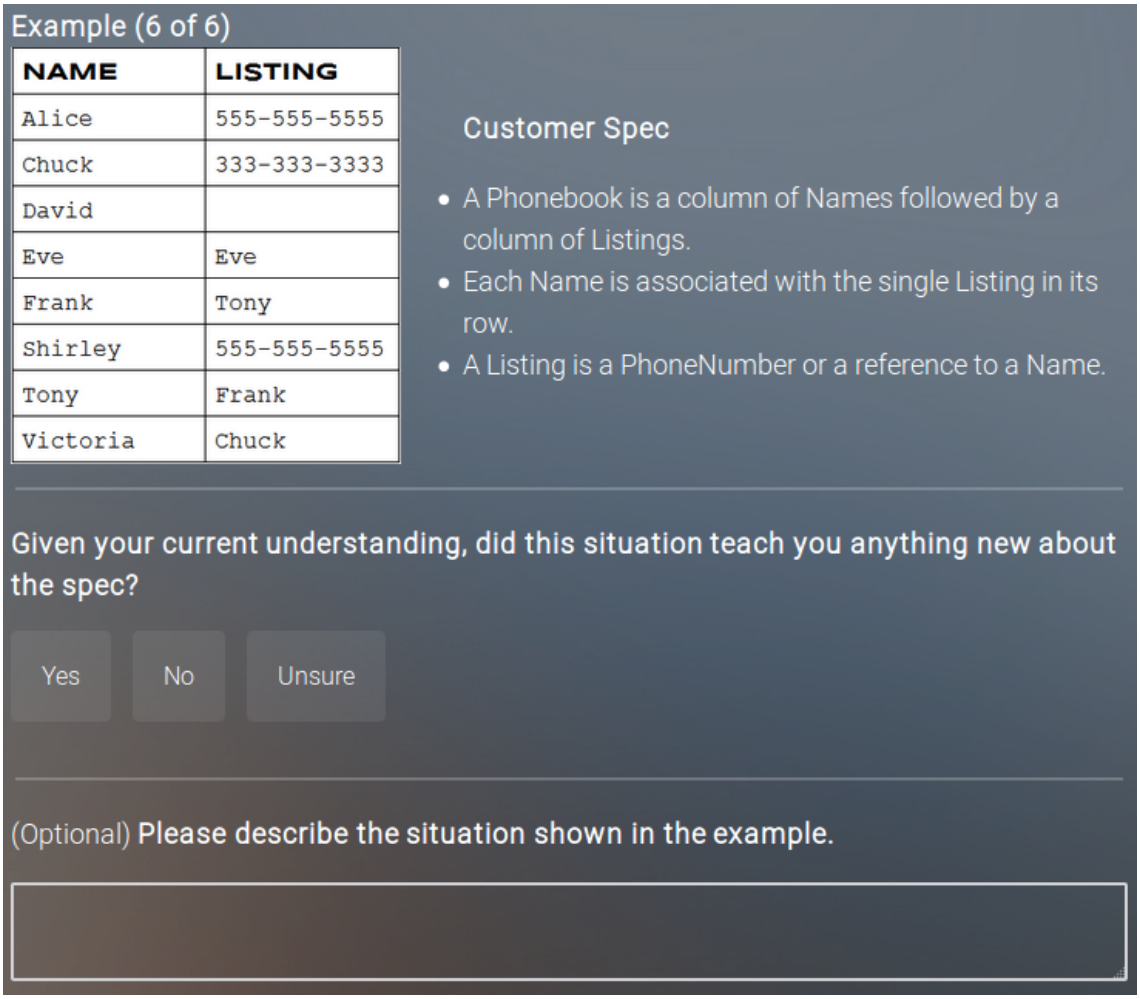


Figure A.8: Example 6

amples to a designer working on the phone books" in an attempt to guide their intuition on what criteria to judge the models on. The rankings they provided were required to be consecutive and total.

A.2.2 Data

The short response results are discussed in section 5.1. The model preference results are discussed in section 4.2.

A.3 Followup Crowd-sourced Preference Study

We gave 200 Mechanical Turkers a similar preference study, except with an explicit failing assertion to check. They were also asked to roughly point out what part of the specification led to the assertion failing. The respondents were then asked to rank the models based on their usefulness for understanding the problem in the specification. The specification is the same, other than the phrasing being changed from a phone book to a catalog table of contents.

A.3.1 Design

The assertion stated that "every name lookup must end in a valid phone number." This assertion fails in the original addressbook specification (empty aliases). It also fails in this catalog phrasing, as none of the mathematical structure of the spec has been modified. For each of the four examples, the respondent picked apart the example by highlighting which facts violated the given assertion. A similar simplified domain specific representation was given. They were also provided with explicit instructions for how to mark the model for failing the assertion. After seeing all four counterexamples, the respondents were asked to point out the part

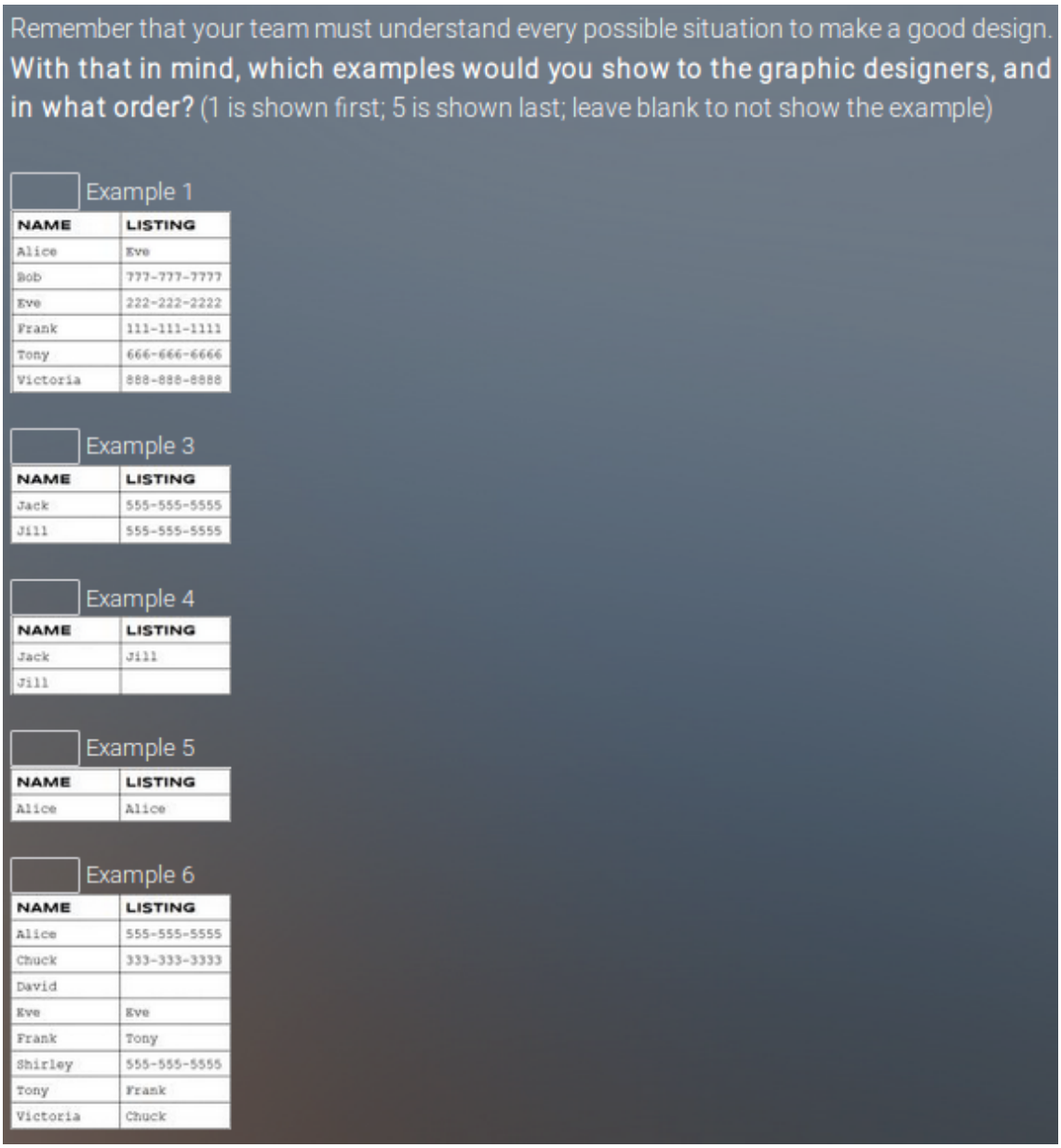


Figure A.9: Example Preference

You are a bug fixer for a catalog company. Your boss has a buggy description for the catalog's table of contents.

Her description specifies how the table of contents works. It reads:

1. Catalog products can be a part of a collection.
2. A collection can be a part of another collection as well.
3. Collections may have their page number listed in the table of contents.

You use software to generate examples of her buggy description to help you find the problem. This survey will have you look at these examples and answer some questions about them.

Figure A.10: Survey Prompt

Example 1 of 4

The bug in the example: some product lookups do not end in a page number.

Highlight the bugs in this example by performing a product lookup as follows.

For each bolded product listed in the left column:

1. Follow the item in the left column to the listing in the right column. If the listing is a:
   - **Page Number: the lookup is valid. Move on to the next product.**
   - **Collection: go to step 2.**
2. Search for the collection in the left column. If you:
   - **Find the collection: repeat step 1 for the collection in the left column.**
   - **Cannot find the collection: this lookup is bugged. Highlight every row in the lookup.**

Table of Contents

Wooden Chair	**page 2**
Wood Collection	see Brown Collection
Oak Collection	see Wood Collection
Red Collection	**page 5**
Cherry Collection	see Red Collection
Cherry Desk	see Cherry Collection
Oak Cabinet	see Oak Collection

Figure A.11: Example 1

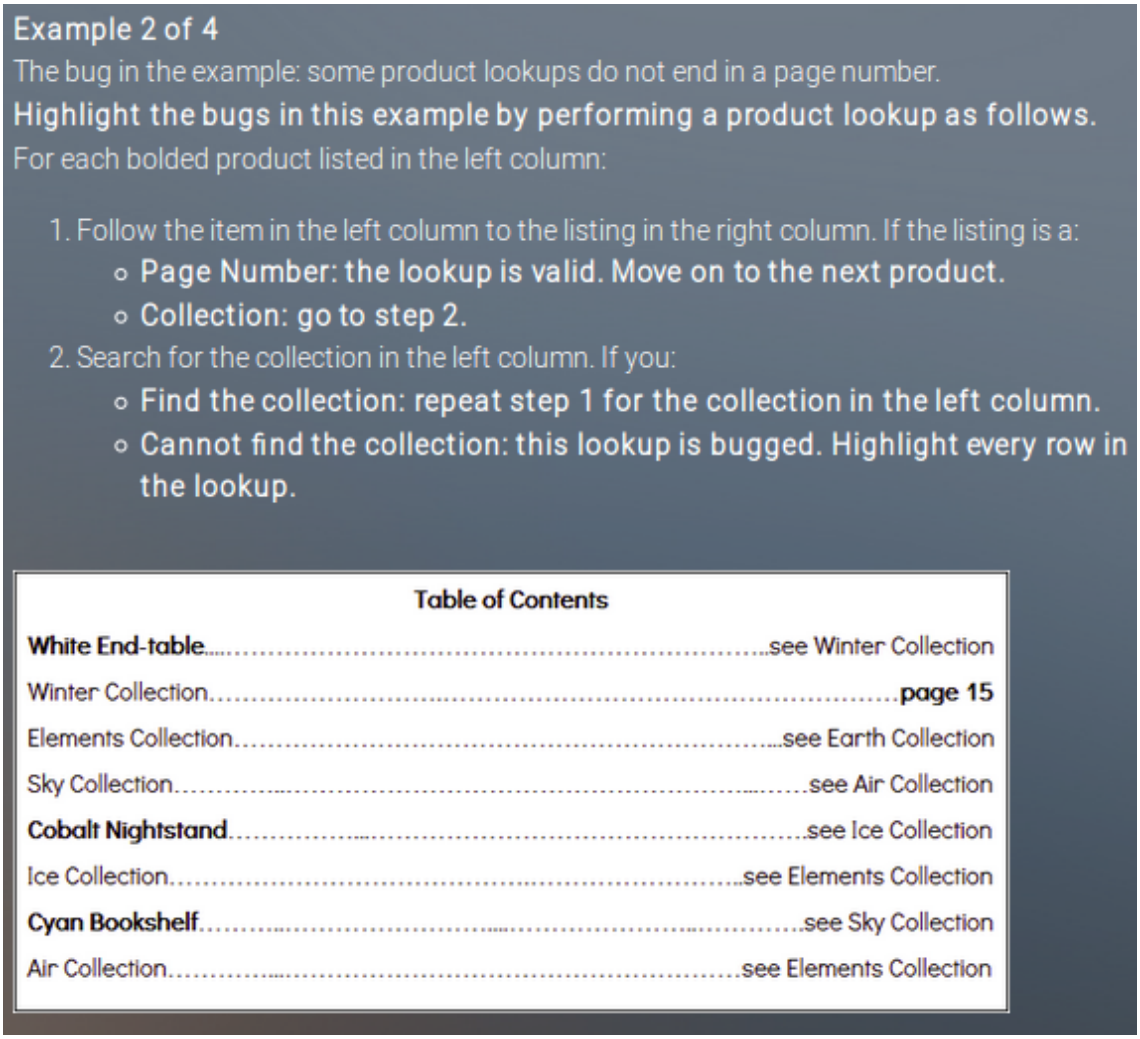


Figure A.12: Example 2

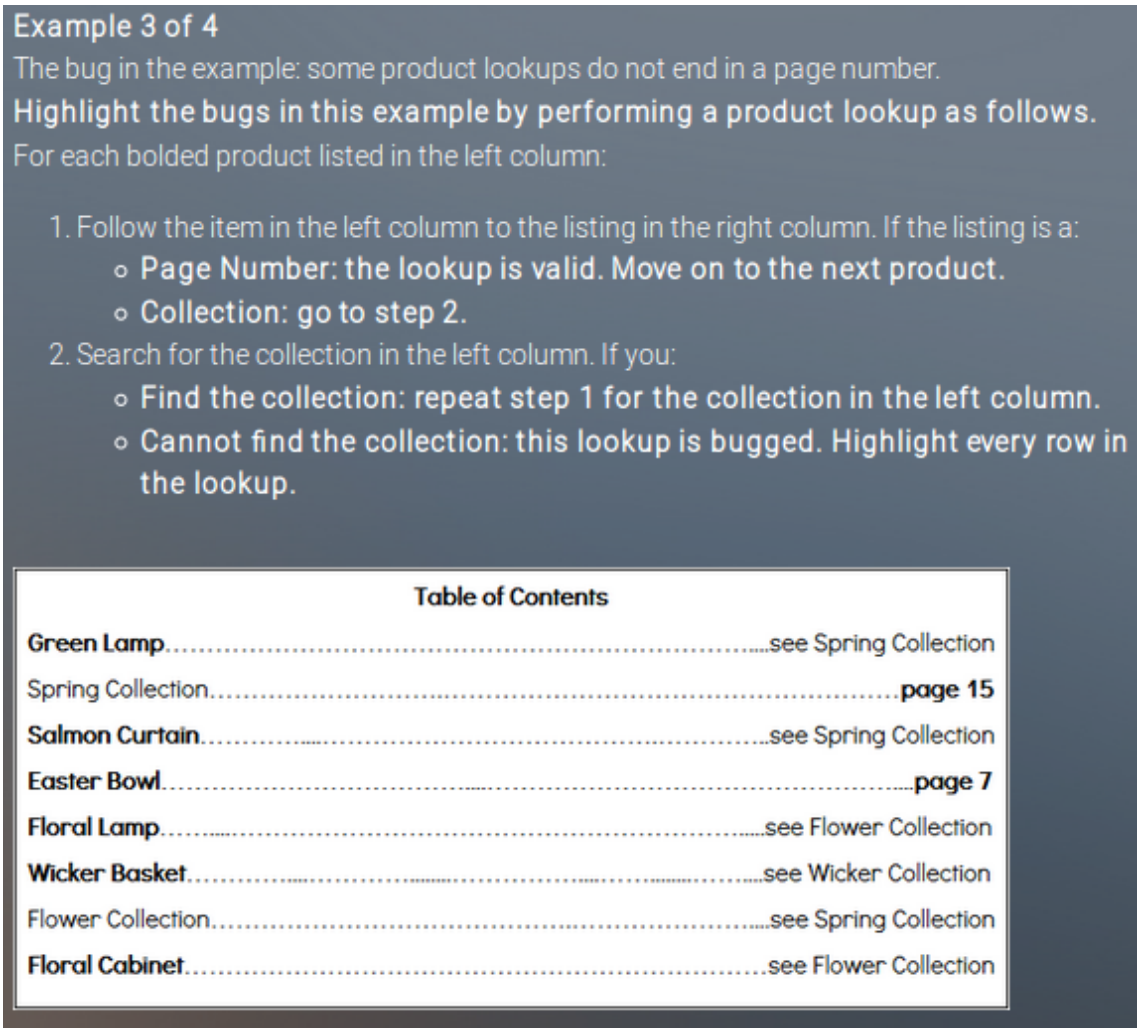


Figure A.13: Example 3

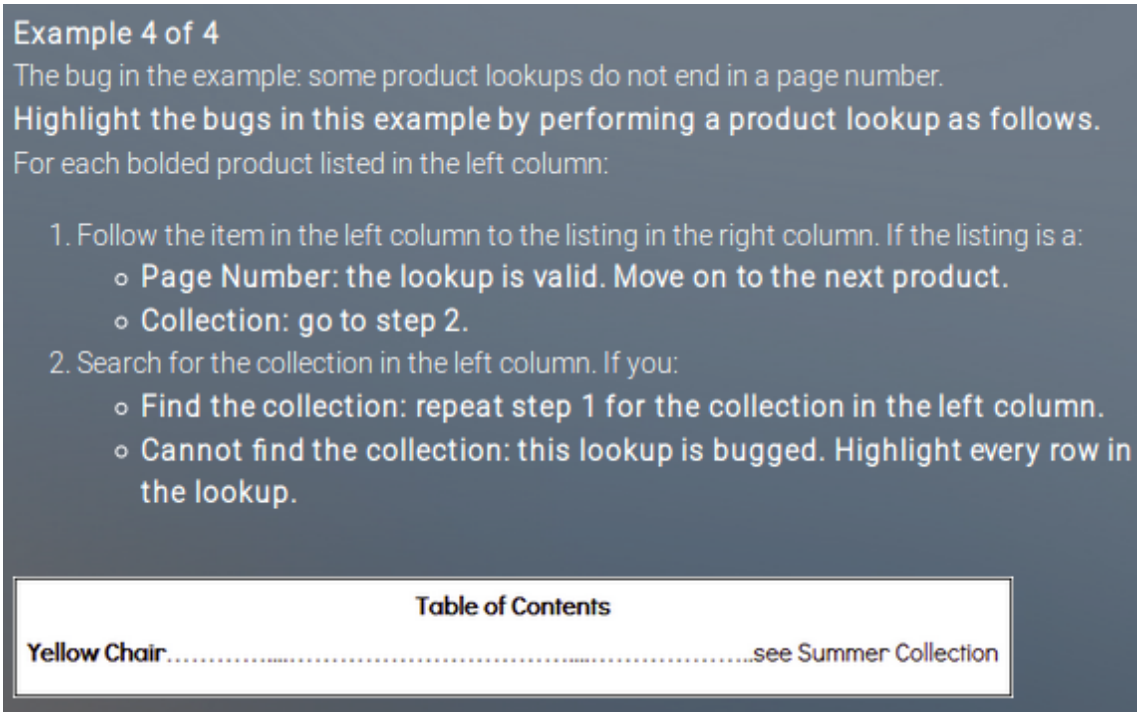


Figure A.14: Example 4

of the specification that caused the assertion to fail. They only had to point out the problematic sentence in the spec, due to the simplified representation having no formal structure. They were also asked to explain their choice and why they chose it. The respondents were asked a similar ranking task to the previous survey. However, now that there is an explicit assertion, their intuition was further guided by the problem in the specification. Their choice should reflect what they believe is the most exemplary counterexample. Again, the rankings are consecutive and total.

A.3.2 Data

The user model preference results are discussed in section 4.3. The model analysis and specification refinement results are discussed in section 5.2.

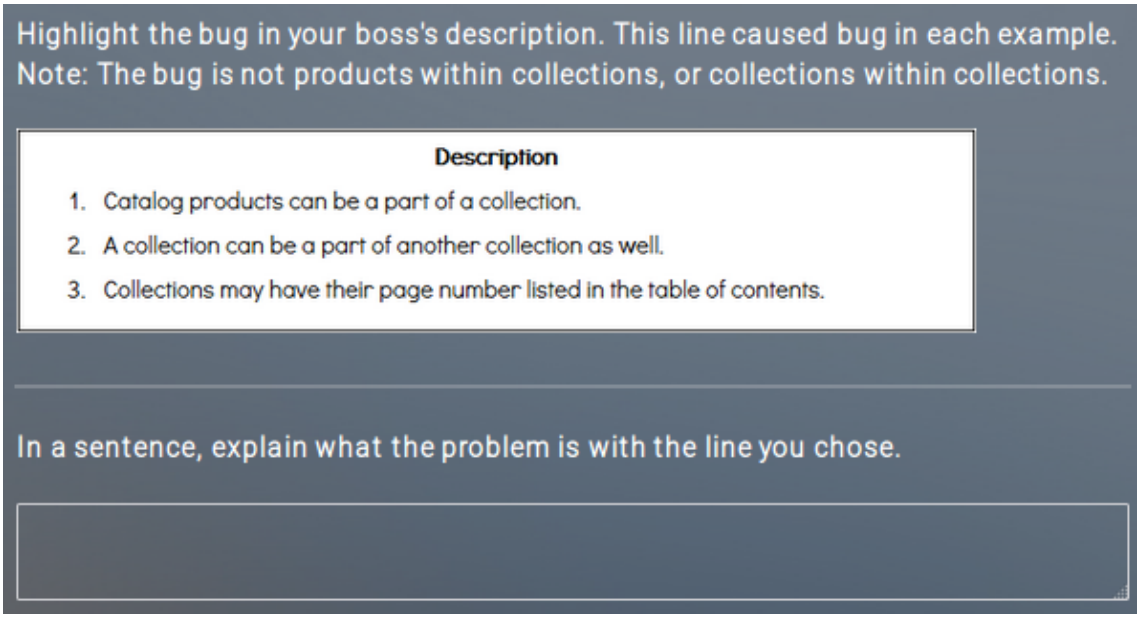


Figure A.15: Specification Refinement

A.4 Initial Crowd-sourced Performance Study

We surveyed hundreds of Mechanical Turkers to measure the effectiveness of particular model types for refining specifications. They were given a specification, assertion to check, instructions on how to analyze each counterexample, as well as teaching feedback. At the end of the survey, they were given a set of possible fixes for the spec, only one of which was completely correct. A between-subjects study was then conducted with varying model types. One variant was minimal while the others were various non-minimal models.

A.4.1 Design

The first example was a teaching example to give the respondents a chance to analyze an example without being graded. Only one true counterexample was presented as the independent variable. The minimal variant is shown in figure A.19. The only difference was the model the respondents were asked to highlight. The grading was

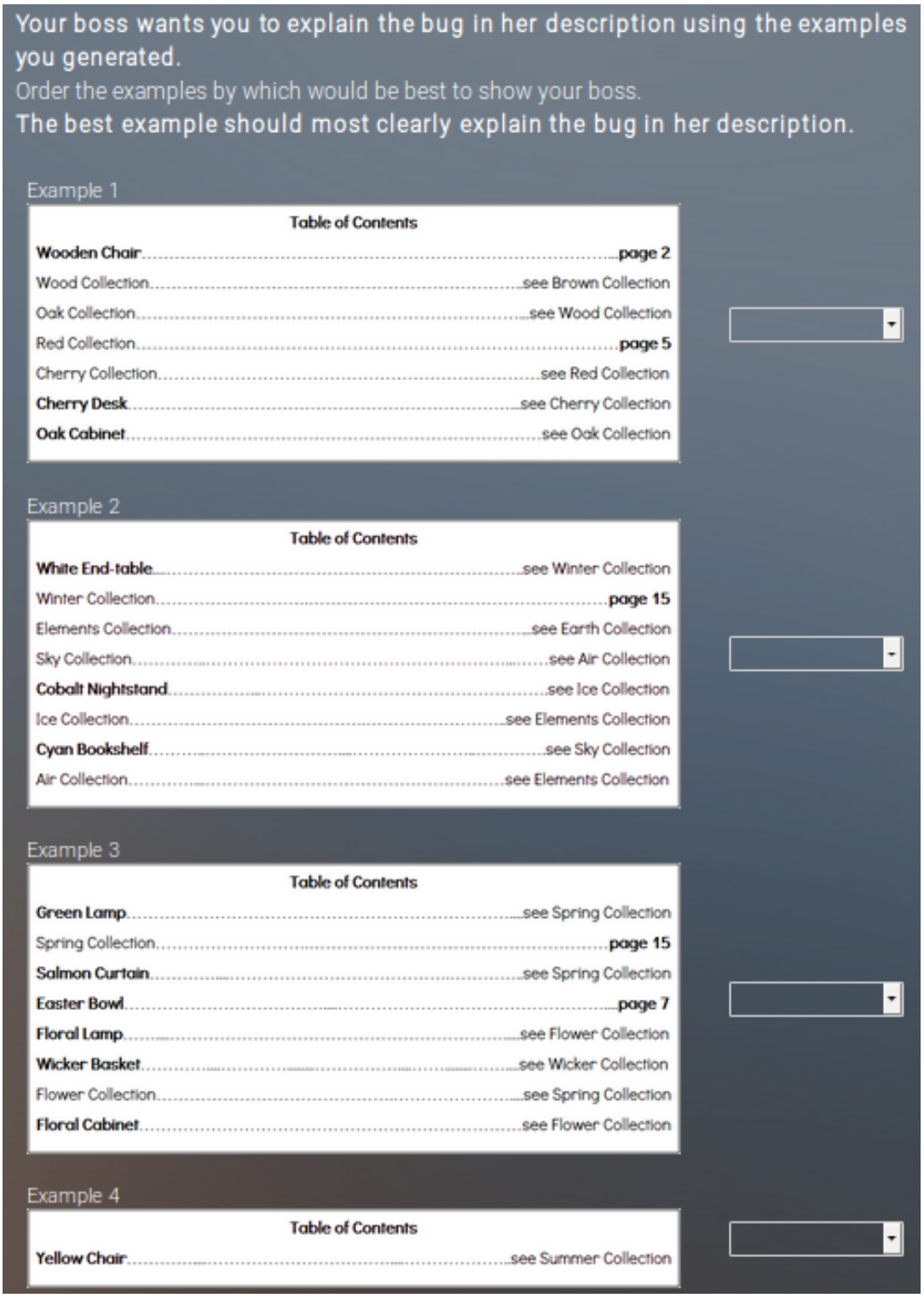


Figure A.16: Example Preference

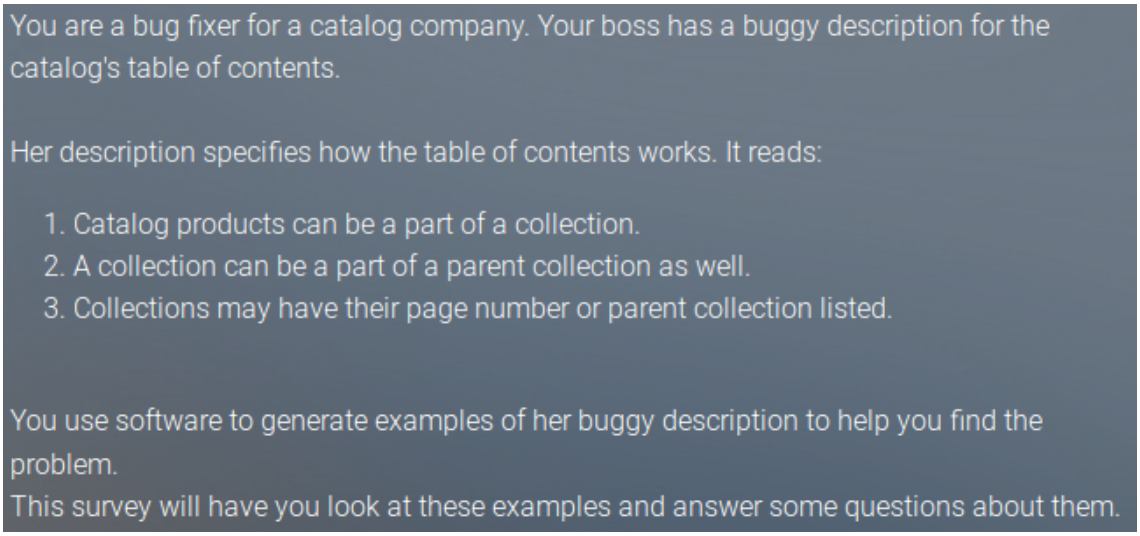


Figure A.17: Survey Prompt

adjusted for the variance in model size. That is, the grade was divided by the number of options, yielding a normalized percentage. After only being exposed to a single counterexample, either minimal or non-minimal, the respondents were asked to choose a proper fix. Since there are multiple ways to fix a specification, some of the options were plausible, but would invalidate a portion of the model presented. They were then asked to explain their choice and why it works.

A.4.2 Data

The user model performance results are discussed in section 6.2.

A.5 Followup Crowd-sourced Performance Study

We surveyed hundreds of Mechanical Turkers to measure the effectiveness of particular model types for refining specifications. They were given a specification, an assertion that is failing, a set of counterexamples of a particular model type, and some possible problem portions of the specification to isolate. Unlike the previous

Example 1 of 2

The bug in the example: some product lookups do not end in a page number.

Highlight the bugs in this example by performing a product lookup. Product lookup instructions and a graphical version of the example are included below.

Table of Contents

Wooden Chair.....*page 2*

Red Collection.....*page 5*

Cherry Collection.....see Red Collection

Cherry Desk.....see Cherry Collection

For each italicized product listed in the left column:

1. Follow the item in the left column to the listing in the right column. If the listing is a:
   - **Page Number: the lookup is valid. Move on to the next product.**
   - **Collection: go to step 2.**
2. Search for the collection in the left column. If you:
   - **Find the collection: repeat step 1 for the collection in the left column.**
   - **Cannot find the collection: this lookup is bugged. Highlight this row.**

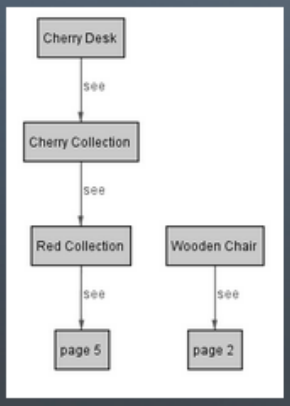


Figure A.18: Example 1

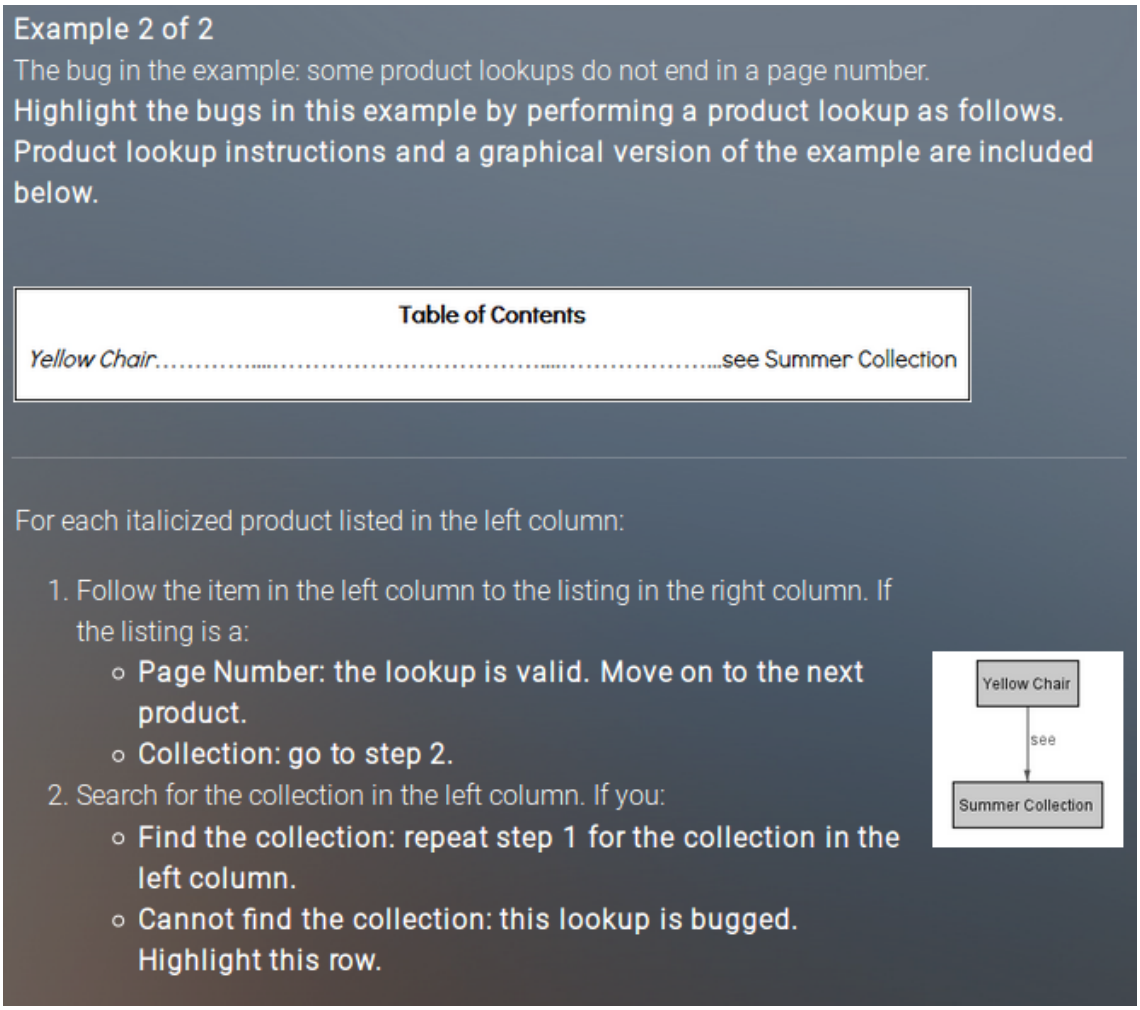


Figure A.19: Example 2

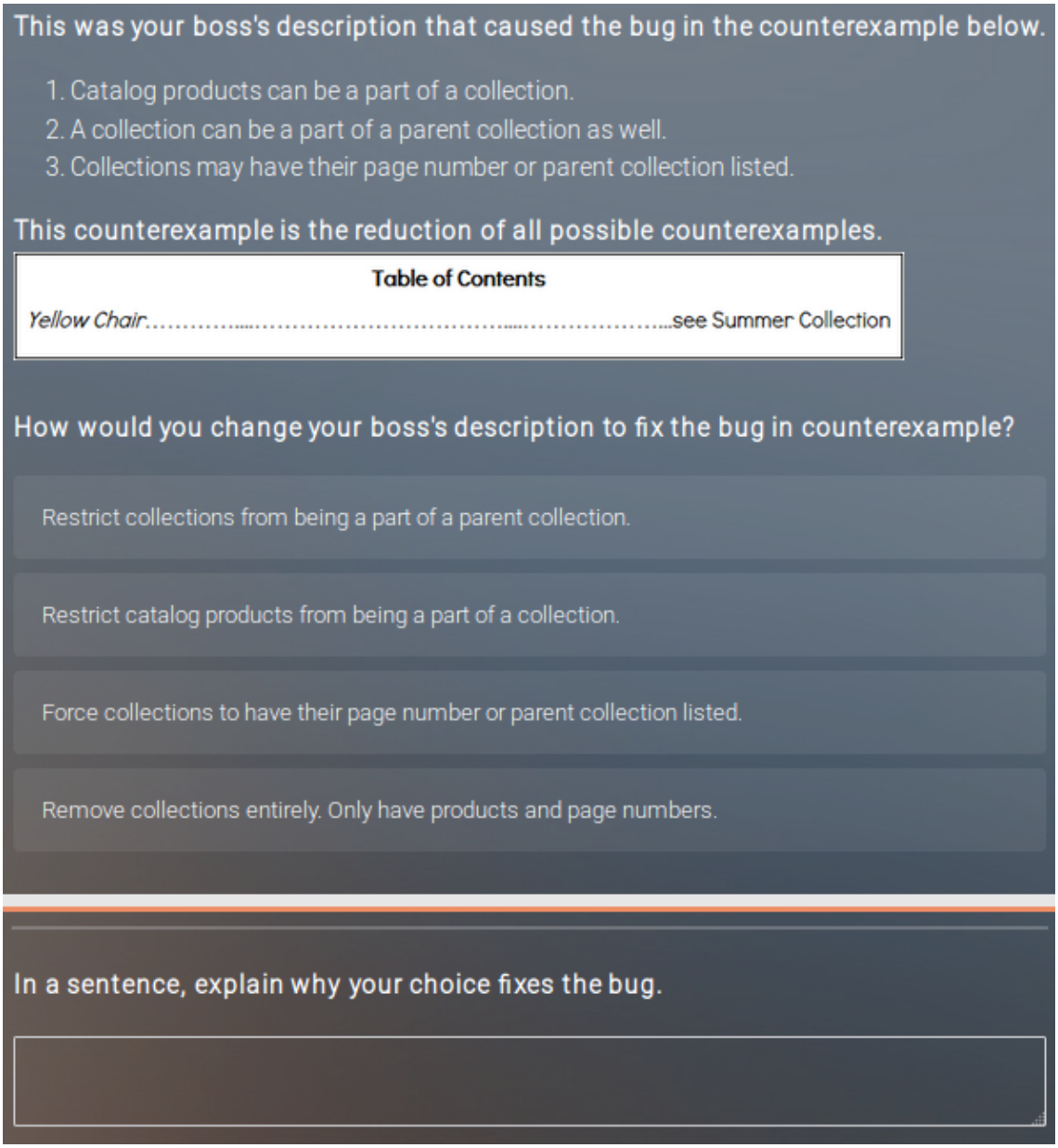


Figure A.20: Specification Refinement

surveys, we formalized the translation from first order relational logic to plain english. This enabled the wording to be consistent between specification and model. It also allowed us to present a succinct task on a single page. Most importantly, the possible problem portions of the spec mapped to definite portions of the original spec, clarifying the criteria for correctness.

A.5.1 Design

Examples of the presented tasks and interaction environment can be seen in figures A.21 and A.24. The users could highlight and select the portion of the spec they thought was responsible for the assertion failing after they had a look at the counterexamples. We originally conducted a within-subjects study, presenting both minimal and non-minimal models to each respondent. To make sure the specification and assertion did not affect their performance, the same spec and assertion was used for both the non-minimal and minimal task variants. We attempted to disguise the spec the second time by rephrasing the entire thing without altering the mathematical structure.

A.5.2 Data

As reported in sections 5.4 and 6.3, there were undesired variances caused by our within-subjects structuring. To mitigate this problem, we converted the data to between-subjects results, which are discussed in section 6.3 and evaluated in section 7.1.1.

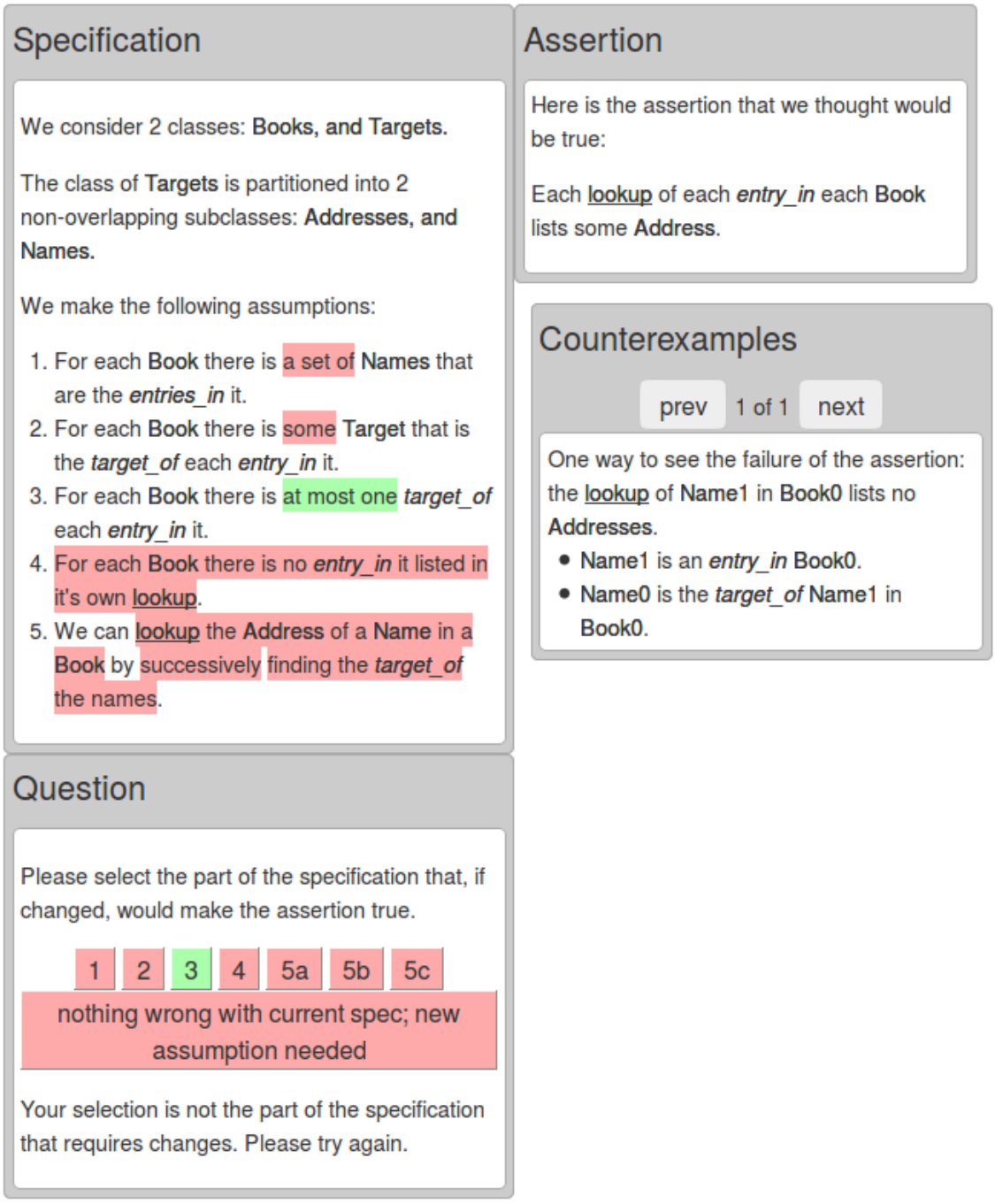


Figure A.21: Addressbook Minimal Task

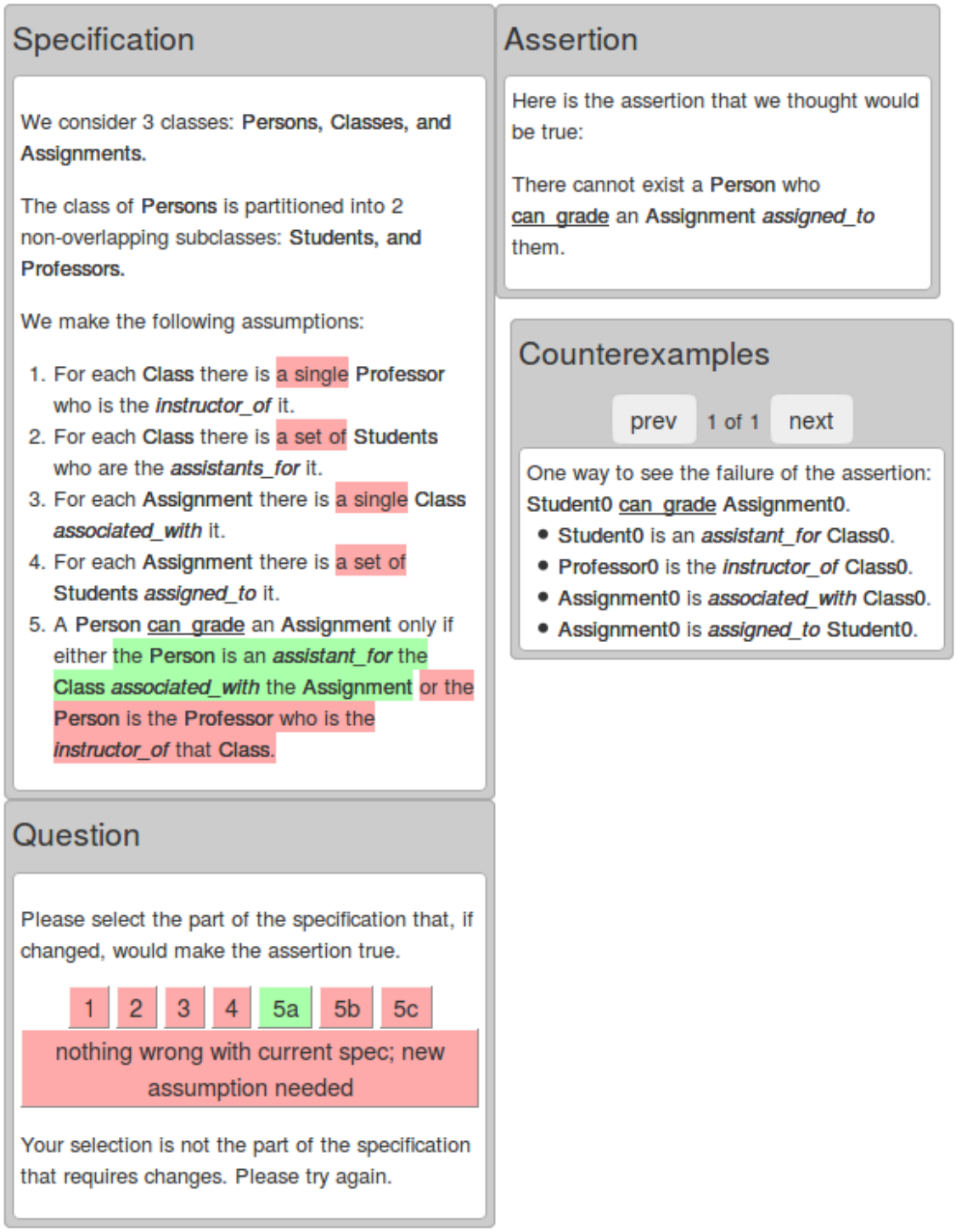


Figure A.22: Gradebook Minimal Task

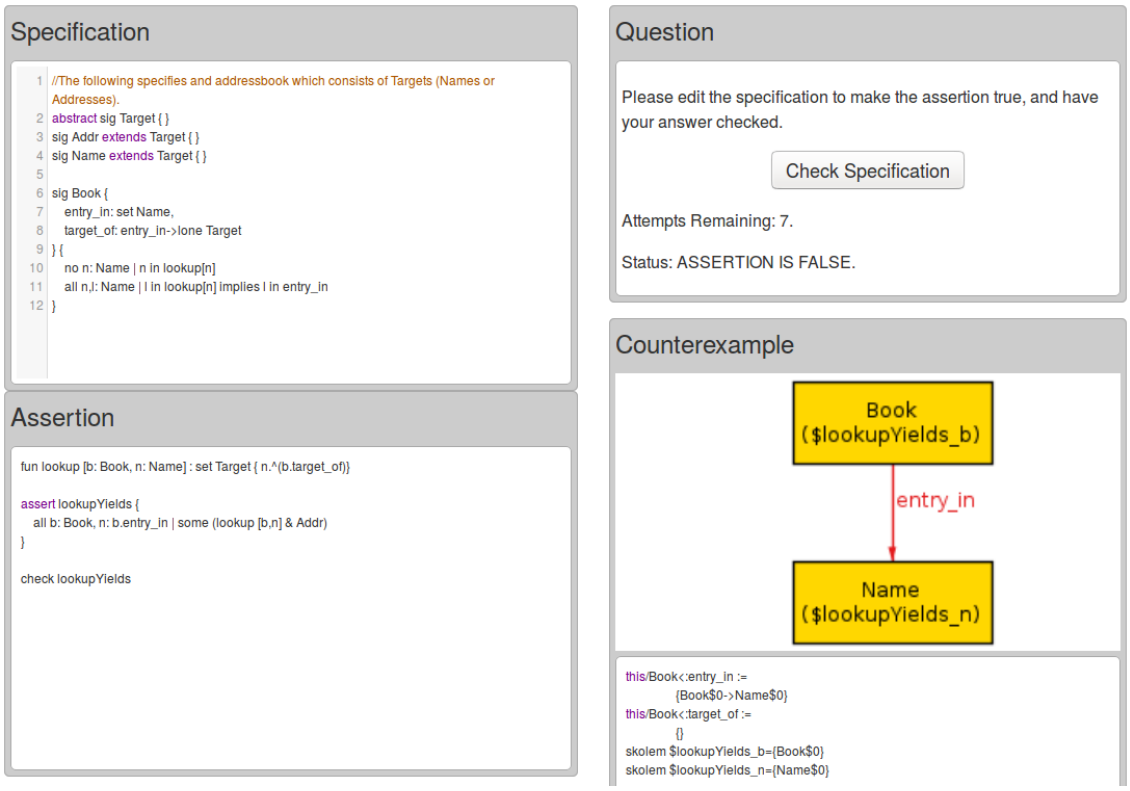


Figure A.23: Addressbook Minimal Task

A.6 Expert Performance Study

We surveyed two dozen Brown undergraduates highly familiar with Alloy. We sought to measure their ability to refine specifications, and compare it to that of the Mechanical Turkers. They were also given a specification, an assertion that is failing, a set of counterexamples of a particular model type. However, they were allowed to freely edit the specification, as they did not require a simplified presentation format due to their experience.

A.6.1 Design

As you can see in figures A.23 and A.24, the task is exactly the same as what was given to the Mechanical Turkers, sans english translations. They also had to fully

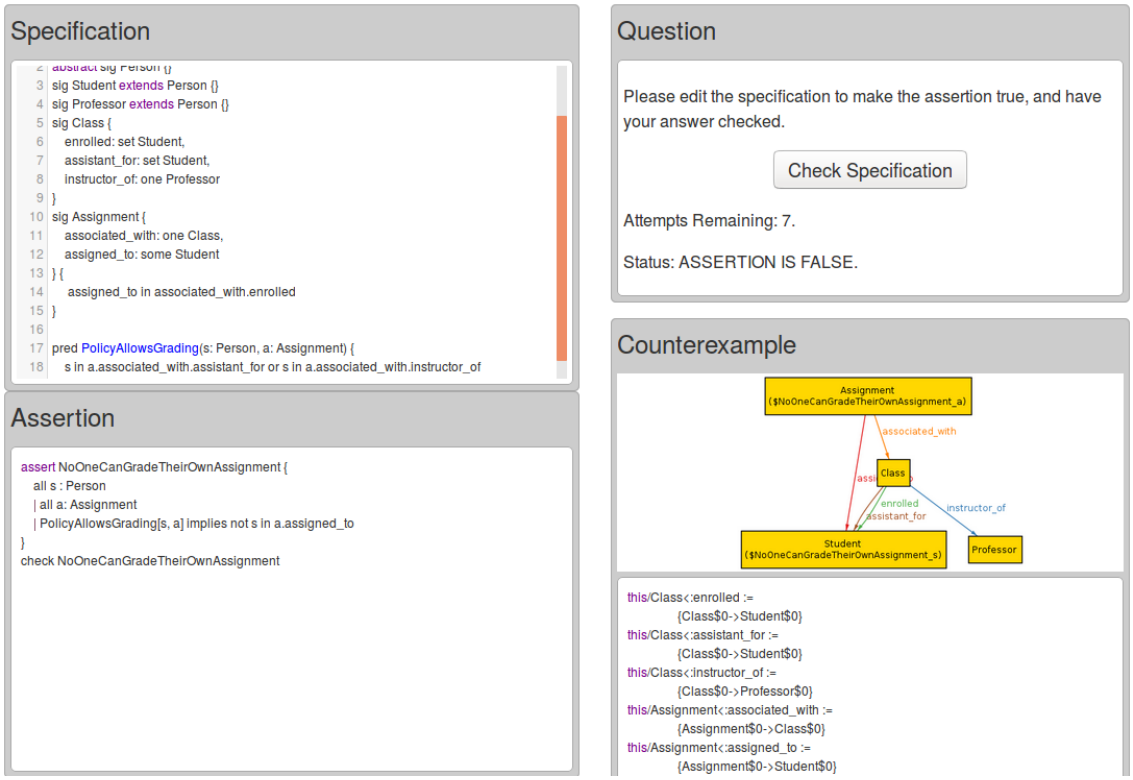


Figure A.24: Gradebook Minimal Task

edit the spec and submit their fix, instead of just tracing the problem to a piece of the spec, and not editing it.

A.6.2 Data

The results are evaluated in section 7.1.2.