Exercise 0F-2. Set Theory [5 points]. This answer should appear after the first page of your submission and may be shared during class peer review.

This exercise is meant to help you refresh your knowledge of set theory and functions. Let X and Y be sets. Let $\mathcal{P}(X)$ denote the powerset of X (the set of all subsets of X). There is a 1-1 correspondence (i.e., a bijection) between the sets A and B, where $A = X \to \mathcal{P}(Y)$ and $B = \mathcal{P}(X \times Y)$. Note that A is a set of functions and B is a (or can be viewed as a) set of relations. This correspondence will allow us to use functional notation for certain sets in class. This is Exercise 1.4 from page 8 of the Winskel textbook.

Demonstrate the correspondence between A and B by presenting an appropriate function and proving that it is a bijection. For example, you might construct a function $f : B \to A$ and prove that f is an injection and a surjection.

Define function $f : B \to A$ as:
for set $M \in B$, $f(M) = h$, where function $h : X \to \mathcal{P}(Y)$ is defined as $h(x) = \{y|(x, y) \in M\}$ for $\forall x \in X$

Define function $g : A \to B$ as:
for function $s \in A$, $g(s) = \{(x, y)|y \in s(x), \forall x \in X\}$

We'll prove that the correspondence between A and B is a bijection by proving both f and g are injective.

Suppose there exists two elements in B, set M and N, such that $f(M) = f(N)$. Let $m = f(M)$, $n = f(N)$ (m, n are functions). For $\forall x \in X$, m(x) = n(x). Since $m(x) = \{y|(x, y) \in M\}$, $n(x) = \{y|(x, y) \in N\}$ for all $x \in X$, and $M, N \in B$, we must have $M = N$. So f is an injection.

Similarly, suppose there exists two elements in A, function s and t, such that $g(s) = g(t)$. That is, $\{(x, y)|y \in s(x), \forall x \in X\} = \{(x, y)|y \in t(x), \forall x \in X\}$. So $s(x) = t(x)$ for all $x \in X$. We know that the domain of both s and t are X, so $s = t$. g is also an injection.

Therefore, we've proved the correspondence between A and B is a bijection.

Exercise 0F-3. Model Checking [10 points]. This answer should appear after the first page of your submission and may be shared during class peer review.

Download the CPAChecker software model-checking tool using the instructions on the homework web-page. Read through enough of the manual to run the tool on the `tcas.i` testcase provided on the homework web-page. Check the three properties given. For each command, copy or screenshot the last ten non-empty lines of output from CPAChecker and include them as part of your answer to this question.

It is your responsibility to find a machine on which CPAChecker works properly (but feel free to check the class forum if you are getting stuck).

Hint: CPAChecker 2.0 should find a violation for `Property1a`, verify that `Property1b` is safe, and find a violation for `Property2b`. If your output does not match that and you are using version 2.0 then you may not have not set things up correctly.

What is going on when you run CPAChecker using the commands listed? In at most three paragraphs, summarize your experience with the CPAChecker tool. What does `Property1a` mean? Is `tcas.i` a reasonable test suite? What has been proved? Did you find CPAChecker to be a usable tool? You may find the graphical reporting option of CPAChecker to be helpful here. For full credit, do not restate my lecture on counter-example guided abstraction refinement; instead, discuss your

Peer Review ID: 62059929 — enter this when you fill out your peer evaluation via gradescope

thoughts and experience using this tool. Focus on threats to validity (e.g., imagine that you were writing a paper and using this as an experiment) over usability.

Both your ideas and also the clarity with which they are expressed (i.e., your English prose) matter. A reader should be able to identify your main claim, the arguments you are making, and your conclusion.

I ran CPAChecker with macOS. When reading the CPAChecker readme file, I noticed that it mentioned possible configuration problems because of MathSAT binaries. So I tried the example commands first, and it failed with "Error: Invalid configuration" as expected. I did some research (on piazza) and finally installed docker to run CPAChecker. The updated command is:

```
docker run -v "$(pwd):/workdir" -u $UID:$GID
↪ registry.gitlab.com/sosy-lab/software/cpachecker:2.0 -predicateAnalysis
↪ -spec Property1a.spc tcas.i
```

Commands for Property1b.spc and Property2b.spc are similar. When running the commands, the `-predicateAnalysis` tells CPAChecker to enable predicate analysis and the `-spec` tells it to expect a specification file (instead of a configuration file, either should be given). Then it looks for our specification file (Property1a.spc, Property1b.spc or Property2b.spc) and verifies our source code (tcas.i) with the corresponding specification.

`Property1a` is checking if variable `Down_Separation` is under `threshold` while variable `Up_Separation` $\leq$ `threshold`. If so, this path leads to an error state.

`tcas.i` is a reasonable test suite because it verifies both cases: model check reaches a "no error" state, and model check finds counter examples.

It is proved that `Property1b` (error case: (`Up_Separation` $<$ `thresh`) $\wedge$ (`Down_Separation` $<$ `thresh`) $\wedge$ (`Up_Separation` $<$ `Down_Separation`)) will never be violated.

CPAChecker is usable with the graphical report. It shows all the states, and highlights related states when a counterexample is reached for further verification. However, according to my experience with CPAChecker in this question, tools are needed to relate each node in the graph to its corresponding variable state in the source code. Current labels in the report graph (like "N183") don't make sense to me.

Submission. Turn in your assignment as a single PDF document via the `gradescope` website. Your name and Michigan email address must appear on the first page of your PDF submission but may not appear anywhere else.

Peer Review ID: 62059929 — enter this when you fill out your peer evaluation via gradescope

1 HW0

- **0 pts** Correct