CS103
Fall 2018

Handout 26
October 19, 2018

Problem Set 4

This fourth problem set explores set cardinality and graph theory. It serves as tour of the infinite (through set theory) and the finite (through graphs and their properties) and will give you a better sense for how discrete mathematical structures connect across these domains. Plus, you'll get to see some pretty pictures and learn about why all this matters in the first place. ☺

Some of the questions on this problem set will assume you've read the online Guide to Cantor's Theorem, which goes into more detail about the mechanics of the proof of Cantor's theorem as well as some auxiliary definitions.

Good luck, and have fun!

Due Friday, October 26th at 2:30PM.
There is no checkpoint problem.

Problem One: Cartesian Products and Set Cardinalities

If A and B are sets, the ***Cartesian product*** of A and B, denoted $\boldsymbol{A \times B}$, is the set

$$\{ (x, y) \mid x \in A \land y \in B \}.$$

Intuitively, $A \times B$ is the set of all ordered pairs you can make by taking one element from A and one element from B, in that order. For example, the set $\{1, 2\} \times \{u, v, w\}$ is

$$\{ (1, u), (1, v), (1, w), (2, u), (2, v), (2, w) \}.$$

For the purposes of this problem, let's have ★ and ☺ denote two arbitrary objects where ★ ≠ ☺. Over the course of this problem, we're going to ask you to prove that $|\mathbb{N} \times \{★, ☺\}| = |\mathbb{N}|$.

i. Draw a picture showing a way to pair off the elements of $\mathbb{N} \times \{★, ☺\}$ with the elements of $\mathbb{N}$ so that no elements of either set are uncovered or paired with multiple elements.

You might want to draw some pictures of the set $\mathbb{N} \times \{★, ☺\}$ so that you can get a better visual intuition.

ii. Based on the picture you came up with in part (i), define a bijection $f : \mathbb{N} \times \{★, ☺\} \to \mathbb{N}$. The inputs to this function will be elements of $\mathbb{N} \times \{★, ☺\}$, so you can define your function by writing

$$f(n, x) = \underline{\hspace{6cm}}$$

where $n \in \mathbb{N}$ and $x \in \{★, ☺\}$.

In defining this function, you cannot assume ★ or ☺ are numbers, since they're arbitrary values out of your control. See if you can find a way to define this function that doesn't treat ★ and ☺ algebraically.

iii. Prove that the function you came up with in part (ii) is a bijection.

The result you've proved here essentially shows that $2\aleph_0 = \aleph_0$. Isn't infinity weird?

Problem Two: Understanding Diagonalization

Proofs by diagonalization are tricky and rely on nuanced arguments. In this problem, we'll ask you to review the formal proof of Cantor's theorem to help you better understand how it works.

(Please read the Guide to Cantor's Theorem before attempting this problem.)

i. Consider the function $f : \mathbb{N} \to \wp(\mathbb{N})$ defined as $f(n) = \emptyset$. Trace through our formal proof of Cantor's theorem with this choice of f in mind. In the middle of the argument, the proof defines some set D in terms of f. Given that $f(n) = \emptyset$, what is that set D? Provide your answer without using set-builder notation. Is it clear why $f(n) \neq D$ for any $n \in \mathbb{N}$?

Make sure you can determine what the set D is both by using the visual intuition behind Cantor's theorem and by symbolically manipulating the formal definition of D given in the proof.

ii. Let f be the function from part (i). Find a set $S \subseteq \mathbb{N}$ such that $S \neq D$, but $f(n) \neq S$ for any $n \in \mathbb{N}$. Justify your answer. This shows that while the diagonalization proof will always find *some* set D that isn't covered by f, it won't find *every* set with this property.

iii. Repeat part (i) of this problem using the function $f : \mathbb{N} \to \wp(\mathbb{N})$ defined as

$$f(n) = \{ m \in \mathbb{N} \mid m \geq n \}$$

Now what do you get for the set D? Is it clear why $f(n) \neq D$ for any $n \in \mathbb{N}$?

iv. Repeat part (ii) of this problem using the function f from part (iii).

v. Give a function $f : \mathbb{N} \to \wp(\mathbb{N})$ such that the set D obtained from the proof of Cantor's theorem is the set $\{ n \in \mathbb{N} \mid n \text{ is even} \}$. Briefly justify your answer.

Problem Three: Simplifying Cantor's Theorem?

Below is a purported proof that $|S| \neq |\wp(S)|$ that doesn't use a diagonal argument:

> ***Theorem:*** If S is a set, then $|S| \neq |\wp(S)|$.
>
> ***Proof:*** Let S be any set and consider the function $f : S \to \wp(S)$ defined as $f(x) = \{x\}$. To see that this is a valid function from S to $\wp(S)$, note that for any $x \in S$, we have $\{x\} \subseteq S$. Therefore, $\{x\} \in \wp(S)$ for any $x \in S$, so f is a legal function from S to $\wp(S)$.
>
> Let's now prove that f is injective. Consider any $x_1, x_2 \in S$ where $f(x_1) = f(x_2)$. We'll prove that $x_1 = x_2$. Because $f(x_1) = f(x_2)$, we have $\{x_1\} = \{x_2\}$. Since two sets are equal if and only if their elements are the same, this means that $x_1 = x_2$, as required.
>
> However, f is not surjective. Notice that $\emptyset \in \wp(S)$, since $\emptyset \subseteq S$ for any set S, but that there is no x such that $f(x) = \emptyset$; this is because $\emptyset$ contains no elements and $f(x)$ always contains one element. Since f is not surjective, it is not a bijection. Thus $|S| \neq |\wp(S)|$. ■

Unfortunately, this argument is incorrect. What's wrong with this proof? Justify your answer by pointing to a specific incorrect claim that's made here and explaining why it's incorrect.

Problem Four: Paradoxical Sets

What happens if we take *everything* and throw it into a set? If we did, we would get back a set called the ***universal set***, which we denote $\mathcal{U}$:

$$\mathcal{U} = \{ \ x \mid x \text{ exists } \}$$

Absolutely everything would belong to this set: we'd have $1 \in \mathcal{U}$, $\mathbb{N} \in \mathcal{U}$, CS103 $\in \mathcal{U}$, whimsy $\in \mathcal{U}$, etc. In fact, we'd even have $\mathcal{U} \in \mathcal{U}$, which is strange but not immediately a problem.

Unfortunately, the set $\mathcal{U}$ doesn't actually exist, as its existence would break mathematics.

i. Prove that if A and B are arbitrary sets where $A \subseteq B$, then $|A| \leq |B|$.

Look at the online Guide to Cantor's Theorem. Formally speaking, if you want to prove that $|A| \leq |B|$, what do you need to prove? Your answer should involve defining some sort of function between A and B and proving that function has some specific property or properties.

ii. Using your result from (i), prove that if $\mathcal{U}$ exists, then $|\wp(\mathcal{U})| \leq |\mathcal{U}|$.

iii. The ***Cantor-Bernstein-Schroeder Theorem*** says that if A and B are sets such that $|A| \leq |B|$ and $|B| \leq |A|$, then $|A| = |B|$. Using the Cantor-Bernstein-Schroeder Theorem and the formal definitions of the different cardinality relations, prove that if A and B are sets where $|A| \leq |B|$, then $|B| \not< |A|$.

In this proof you're showing that the ≤ and < relations involving set cardinality work like the ≤ and < relations over regular numbers. Since the goal of the proof is to show that these cardinality relations work like regular inequality symbols, this result isn't "obvious" and you'll need to use formal definitions.

Be careful not to use relations we haven't defined. For example, we know that $|A| < |B|$ means because we have a definition for it in terms of bijections and injections. We don't, however, have a definition of what $|A| > |B|$ means, so you should not use that notation in your solution (after all, we don't know what it means!) In particular, make sure not to assume that the negation of $|A| \leq |B|$ is $|A| > |B|$, since we don't have any reason to believe this.

iv. Using your results from parts (i), (ii), and (iii) of this problem, prove that $\mathcal{U}$ does not exist.

The result you've proven shows that there is a collection of objects (the collection of everything that exists) that cannot be put into a set (whoa!!). When this was discovered at the start of the twentieth century, it caused quite a stir in the mathematical world and led to a reexamination of logical reasoning itself and a more formal definition of what objects can and cannot be gathered into a set. If you're curious to learn more about what came out of that, take Math 161 (Set Theory) or Phil 159 (Non-Classical Logic).

Problem Five: Avoiding Sampling Bias with Independent Sets

An ***independent set*** in a graph $G = (V, E)$ is a set $I \subseteq V$ with the following property:

$$\forall u \in I.\ \forall v \in I.\ \{u, v\} \notin E.$$

This question explores independent sets and their properties.

i. Explain what an independent set is in plain English and without making reference to first-order logic. No justification is necessary.

You may want to draw some pictures of graphs to see what independent sets look like. Don't just come up with a literal translation of the first-order logic formula above; see if you can find a simple explanation.

ii. Download the starter files for Problem Set Four from the course website, extract them somewhere convenient, then open the file `GraphTheory.cpp` and implement a function

```
bool isIndependentSet(Graph G, std::set<Node> I)
```

that takes as input a graph G and a set I, then returns whether I is an independent set in G. The definition of the `Graph` type is provided in `GraphTheory.h`.

Our provided starter code contains logic that, given a graph G, finds the largest independent set in G by making a lot of repeated calls to `isIndependentSet`. You might want to look over some of the sample graphs to get a feel for what large independent sets look like.

You want to conduct a poll for an election and would like to survey people where no two people in the group know each other so that you can get a good representative sample of the population. Ideally, you'd find a large group of mutual strangers so that your poll has good statistical power and avoids *sampling bias*, a weakness of a poll in which the sampled population isn't representative of the larger population.

iii. Explain how to model this problem as looking for a large independent set in a certain graph. What would the nodes in your graph be? What would the edges be? And why would an independent set have the properties you want? No formal proof is necessary, but do justify your answer.

*You don't need to tell us how to actually **find** that independent set. That's a question for another day…*

A ***vertex cover*** in a graph $G = (V, E)$ is a set $C \subseteq V$ with the following property:

$$\forall x \in V.\ \forall y \in V.\ (\{x, y\} \in E \to x \in C \lor y \in C).$$

That's a bit of a mouthful, but this definition isn't as tough as it seems.

iv. Translate the definition of a vertex cover into plain English, similarly to what you did in part (i).

v. Implement a function

```
bool isVertexCover(Graph G, std::set<Node> C)
```

that takes as input a graph G and a set C, then returns whether C is a vertex cover of G. The definition of the `Graph` type is provided in `GraphTheory.h`.

Our provided starter code contains logic that, given a graph G, finds the smallest vertex cover in G by making a lot of repeated calls to `isVertexCover`. You might want to look over some of the sample graphs to get a feel for what small vertex covers look like.

There's a close connection between independent sets and vertex covers.

vi. Let $G = (V, E)$ be a graph and C be a vertex cover of G. Prove that $V - C$ is an independent set of G.

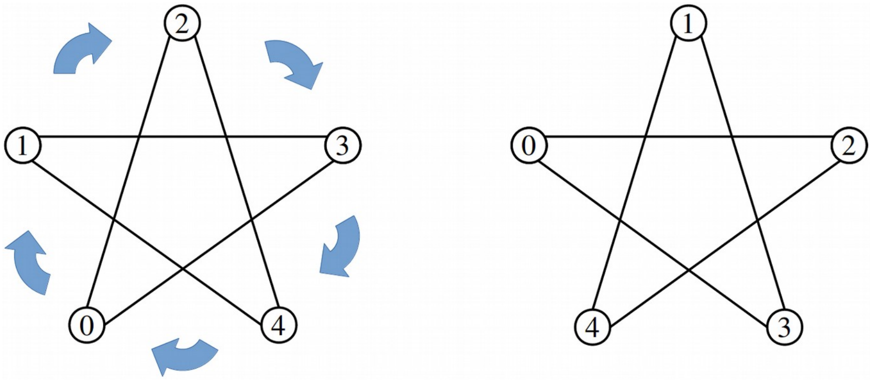
Be sure to use the formal definitions of the relevant terms here. It's a lot easier to see what's going on here if you draw out some sample graphs, along with sample vertex covers and independent sets.

Problem Six: Symmetries, Permutations, and Automorphisms

One of the more beautiful concepts in mathematics is *symmetry*. Chances are that you have some intuitive conception of what "symmetry" means, and not just in the binary relation sense. For example, the letter M looks like itself when you look at it in a mirror, and the number 8 looks like itself if you flip it horizontally or rotate it 180°.

Now that we're studying graphs, we can ask what it means for a *graph* to be symmetric. For example, consider the graph of the {5 / 2} star shown below. This graph has many symmetries – if you flip it horizontally, or rotate it 72°, the resulting graph looks identical. But this notion of "looks identical" appeals to our human visual system, which, as you've probably experienced, is easily tricked. Is there a way to come up with a more rigorous definition of "symmetry?"

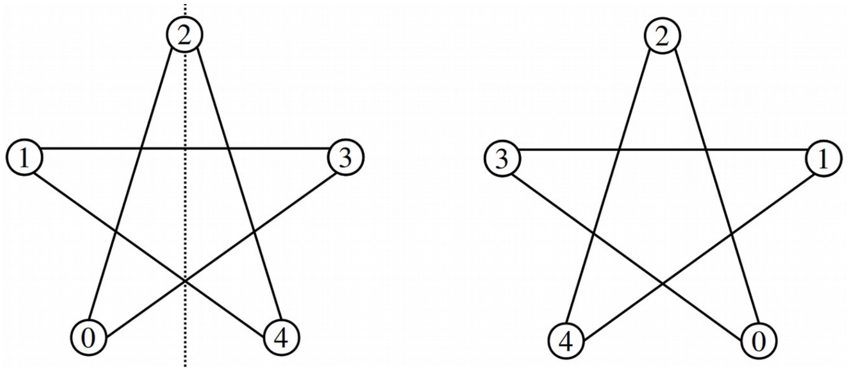
Let's begin by labeling the nodes of this graph in whatever way we'd like. For convenience, we'll use the numbers 0, 1, 2, 3, and 4, following our star-drawing convention. Now, take the graph shown below and rotate it one fifth of a revolution clockwise, as shown here:



If you'll notice, this operation on the graph corresponds to finding a permutation of the labels on the nodes. Specifically, the node 0 ended up where the node 1 used to be, the node 1 ended up where the node 2 used to be, etc. And hey! We have a way of describing transformations like that.

i. Describe the permutation that the above rotation corresponds to using two-line notation. No justification is necessary.

Similarly, let's imagine that we mirror the graph horizontally, as shown here:

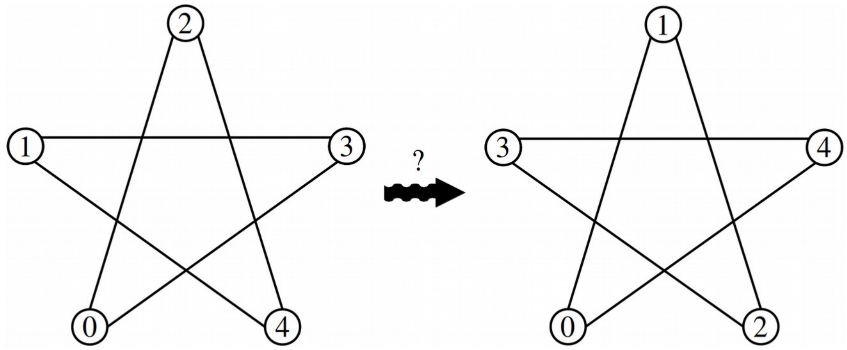


We can similarly think of this as a permutation. For example, node 1 maps to where node 3 used to be.

ii. Describe the permutation that the above mirroring corresponds to using two-line notation. No justification is necessary.

(Continued on the next page…)

There are a number of other symmetries of this graph, and each one of them corresponds to a permutation of the nodes. However, not all node permutations correspond to symmetries. For example, imagine that we permute the nodes as shown here:



Notice that, in the original graph, nodes 0 and 2 were adjacent, but after permuting the nodes they're no longer adjacent to one another. (Remember that *adjacent* means "directly linked by an edge;" contrast this with *connected*, which has a different meaning.) Similarly, in the original graph, nodes 0 and 1 were *not* adjacent, but after applying the permutation they now are.

At this point, we've seen some positive examples of cases where we *can* use permutations to model symmetries, along with a negative example of where we *can't* use permutations to model symmetries. The question is, what was special about those original permutations? It's that they have the following property: those permutations keep adjacent nodes adjacent and keep non-adjacent nodes non-adjacent.

Formally speaking, let $G = (V, E)$ be a graph. A permutation σ of the nodes of V is called an ***automorphism*** if the following is true about σ:

$$\forall u \in V.\ \forall v \in V.\ (\{u, v\} \in E \leftrightarrow \{\sigma(u), \sigma(v)\} \in E).$$

The mathematical term "automorphism" ("self shape") might seem really scary here, but it's just a fancy way of pinning down what we've been describing with the term "symmetry." Restating the above in plain English, a permutation is an ***automorphism*** of G if

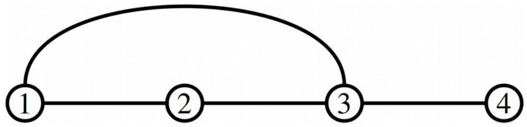
- any two nodes that are adjacent in G stay adjacent after applying σ, and
- any two nodes that aren't adjacent in G stay nonadjacent after applying σ.

The permutations that you described in parts (i) and (ii) are automorphisms, while the permutation shown above on this page isn't. The rest of this question explores properties of automorphisms.

iii. How many automorphisms does the graph of $\{5 / 2\}$ have? Briefly describe what they are. No formal proof is needed.

Use your intuition behind what automorphisms are designed to formalize.

iv. Consider the graph shown below. It has two automorphisms. What are they? Express your answer using two-line notation.



A graph is just a pair of a set of nodes and a set of edges. We can draw a graph $G = (V, E)$ in two-dimensional space as a collection of circles and lines in lots of different ways. Sometimes, those drawings make it really easy to find symmetries. Sometimes, those drawings make it really hard to find symmetries.

To round out our discussion of automorphisms, we'd like you to prove one nice, formal result about them.

v. Let $G = (V, E)$ be a graph and let σ and τ be two automorphisms of G. Prove that $\tau \circ \sigma$ is also an automorphism of G.

Write out the definition of an automorphism and make a list of what you'll need to show.

Problem Seven: Bipartite Graphs

There are a few famous families of graphs that come up over and over again. One of the most important types of graphs in computer science is the *bipartite graph*, which is the focus of this problem.

Let's begin with a formal definition of bipartite graphs. An undirected graph $G = (V, E)$ is called ***bipartite*** if there exist two sets V_1 and V_2 such that

- every node $v \in V$ belongs to exactly one of V_1 and V_2, and
- every edge $e \in E$ has one endpoint in V_1 and the other in V_2.

The sets V_1 and V_2 here are called ***bipartite classes*** of G. To help you get a better sense for why bipartite graphs are important and where they show up, let's work through a couple of examples.

i. Consider a graph where each node represents a square on a chessboard and where there's an edge between any pair of squares that are immediately adjacent in one of the four cardinal directions (up, down, left, and right). Explain why this is a bipartite graph by telling us what the bipartite classes are and briefly explaining why all the edges have one endpoint in each bipartite class.

Draw lots of pictures!

ii. Consider a graph where each node represents a person at a company and there's an edge between any person and their boss. (Since edges are undirected, that also means that there's an edge from each person to all the people who report to them). You can assume everyone has a single boss, except for people who are in charge of a company, who have no bosses at all. Explain why this is a bipartite graph, along the lines of what you did in part (i).

Draw lots of pictures!

Bipartite graphs have many interesting properties. One of the most fundamental is this one:

> ***Theorem:*** An undirected graph G is bipartite if and only if it contains no cycles of odd length.

The forward direction of this implication has a nice intuition.

iii. Explain, intuitively, why if G is bipartite, then it has no cycles of odd length. Specifically, give us a brief, informal explanation as to why every cycle in G has to have even length.

The reverse direction of this implication – that if G has no cycles of odd length, then G is bipartite – requires a different sort of argument.

Let's pick some arbitrary graph $G = (V, E)$ that has no cycles of odd length. For simplicity's sake, we'll assume that G has just one connected component. If G has two or more connected components, we can essentially treat each one of them as independent graphs. *(Do you see why?)*

Now, choose any node $v \in V$. Using node v as an "anchor point," we can define two sets V_1 and V_2 that we'll need for the remainder of this argument:

$$V_1 = \{\ x \in V \mid \text{there is an odd-length path from } v \text{ to } x\ \}$$

$$V_2 = \{\ x \in V \mid \text{there is an even-length path from } v \text{ to } x\ \}$$

This turns out to be a really useful way to group the nodes of G.

iv. Given the choices of G and v from above, prove that V_1 and V_2 have no nodes in common.

Remember that there might be multiple different paths of different lengths from v to some other node x, so be careful not to talk about "the" path between v and x. Also note that these ***do not*** *have to be* ***simple*** *paths.*

v. Using your result from part (iv), prove that G is bipartite.

The most common mistake on this problem is to not address all the parts of the definition of a bipartite graph. So start off by writing down a list of what you need to prove, then address each part in turn.

Problem Eight: Detecting Fake Networks

Social networking sites like Facebook and Twitter are great places to get a pulse on what other people are thinking, to learn about what's going on in the world, and to decide what products and services to buy. In many ways, this has been wonderful. But to add another page to the book of "This is Why We Can't Have Nice Things," this shift toward social media has led to some problems. Unscrupulous actors (advertisers, trolls, nation states, etc.) have swayed opinion by creating fake social media accounts and posting unpopular or fringe views – whether it's that Product *X* can simultaneously cure cancer and make you rich or that Candidate *Y* is a Really Swell Fella – as though they were authentic expressions of support by real people.

Let's imagine that you're working for the next social media giant – which you or your fellow CS103ers may end up doing! – and you want to put an end to these fake accounts. How could you figure out who's real and who isn't? It turns out that graph theory has a pretty rich set of tools to do just this.

i. Think of any reasonably-sized group of (real) people. This gives rise to a graph: each person is a node, and each edge represents a pair of people who regularly interact with one another. Explain why it would be *extremely unlikely* for that graph to be bipartite. No proof is necessary.

The popular company Social Media Helper (SMH) has a product called Super Ethical Opportunity (SEO) that they offer to their customers. If you sign up for SEO, for a small fee, SMH will create a brand new social media account for you. SMH will then create an additional 100,000 brand new social media accounts to LikeFollowFriendSubscribe™ your new account, making it look as though you're really popular. After all, how many people can say they have 100,000 LikeFollowFriendSubscribers™? Aside from LikeFollowFriendSubscribing™ your account, those new accounts just sit there doing nothing. They just exist to boost your apparent popularity. (Of course, nothing like this could ever happen in real life.)

ii. Explain why the social network formed from SMH's fake accounts, plus the accounts they're LikeFollowFriendSubscribing™, must be a bipartite graph. No proof is necessary.

It turns out that it's computationally quite easy to check if a particular graph is bipartite and to find large bipartite clusters within a graph. This means that if you find yourself working at a social media behemoth, you could check to see if you have large, connected, bipartite subgraphs of your social network. The accounts in those subgraphs might be worth looking into. ☺

iii. There's always a cat-and-mouse game between people offering products like SEO and the technologists working at social networks trying to catch them. Describe a technique that SMH could use to prevent the graph of their fake accounts and their customers' accounts from being bipartite.

The analysis here – just looking for bipartite clusters – is extremely simple, and yet you'd be amazed at just how effective it is at spotting fake networks. We first heard about it from a blogger who used it to find fake advertising networks on Twitter.

If you'd like to learn more about real social networks, their properties, and the sorts of analyses we can perform on them, take CS224W. To learn how to check if a graph is bipartite, and to see how you'd find large bipartite clusters in a graph, take CS161. For more information on the ethical responsibilities of running a massive social network, take CS181.

Problem Nine: Chromatic and Independence Numbers

Recall that if G is a graph, then $\chi(G)$ represents the ***chromatic number*** of G, the minimum number of colors needed to paint each node of G so no two adjacent nodes of G are the same color. The ***independence number*** of a graph, denoted $\boldsymbol{\alpha(G)}$, is the size of the largest independent set in G.

Let n be an arbitrary positive natural number. Prove that if G is an arbitrary undirected graph with exactly n^2+1 nodes, then $\chi(G) \geq n+1$ or $\alpha(G) \geq n+1$ (or both).

You should definitely check out the Guide to Proofs on Discrete Structures' advice for proving $P \vee Q$.

On this problem set, we've included three optional fun problems you can play around with. You're welcome to play around with any number of them, but ***please submit answers to at most one of these problems*** with your problem set. We, unfortunately, don't have TA bandwidth to grade both of them. If you submit solutions to more than one of them, we'll choose one to grade arbitrarily.

Optional Fun Problem One: Hugs All Around! (Extra Credit)

There's a party with 137 attendees. Each person is either ***honest***, meaning that they *always* tell the truth, or ***mischievous***, meaning that they *never* tell the truth. After everything winds down, everyone is asked how many ***<u>honest</u>*** people they hugged at the party. Surprisingly, each of the numbers 0, 1, 2, 3..., and 136 was given as an answer exactly once.

How many honest people were at the party? Prove that your answer is correct and that no other answer could be correct.

Optional Fun Problem Two: How Many Functions Are There? (Extra Credit)

If A and B are sets, we can define the set $\boldsymbol{B^A}$ to be the set of all functions from A to B. Formally speaking:

$$B^A = \{\ f \ \mid\ f : A \to B\ \}$$

Prove that $|\mathbb{N}| < |\mathbb{N}^{\mathbb{N}}|$. This shows that $\aleph_0 < \aleph_0^{\aleph_0}$. Isn't infinity weird?

Optional Fun Problem Three: Chemical Automorphisms

Below is a graph of the molecular structure of octomethylcyclotetrasiloxane. How many automorphisms does this graph have? Justify your answer, but no formal proof is required.