Grade 6 Math Circles
November 18, 2020

Bijectons and Binary Strings

Introduction
Consider the following scenario: Say that you have just gone to the store with your family and have bought 6 board games. At home, you have 4 different shelves on which you could place a board game. We could then say that at least one of the shelves would have to house two of the new board games in order for all the new board games to be placed on the shelves. Why? Because we have more new board games than we do shelves. This is an idea known as the Pigeonhole Principle. If we have more objects than containers, at least one container has to have two objects.

This makes the idea of knowing if groups of objects are the same size very important in mathematics. Today, we will talk about bijectons and how they can show that two groups have the same size. We will also talk about binary strings, their decompositions and uses in bijectons. Before we can dive into bijectons however, we need a bit more background.

Sets
A set is a collection of objects: any collection can be called a set! The objects contained in a set are called elements. When writing a set, we wrap the elements in \{\} to represent that we are writing a set.

Examples:
Numbers: \{1, 2, 3, 4, 5\}
Letters: \{a, b, c\}
Grocery List: \{apple, bread, chicken, grapes\}
Sets of Sets: \\{\{1, 2\}, \{a, b, c\}, \{apple\}\}

Sets can be as long as we want, but they can’t have any duplicates and don’t have a specific order. So, the set \{1, 2, 3\} is the same as the set \{2, 1, 3\}. The number of elements in a set is known as its cardinality. We represent cardinality with | | around the curly brackets. For example, |\{apple, bread, chicken, grapes\}| = 4.
Relations

A relation is a way to relate objects from two sets. It may be helpful to think of a relation as a path from one set to another. For example, consider set \( f = \{a, b, c, d\} \) and set \( g = \{1, 2, 3, 4\} \). One relation between these two sets is shown here:

![Diagram showing a relation between sets f and g]

Note we have created connected pairs \(\{1, d\}\), \(\{2, a\}\), \(\{3, c\}\), \(\{4, c\}\). The initial set is known as the **domain** and the second set is known as the **codomain**.

Functions

A function is a relation where each element of the domain is connected to exactly one element in the codomain. For example:

1. Not a function:

   ![Diagram showing a relation that is not a function]

   Our first relation is not a function as element 2 from the domain is connected to both elements \(a\) and \(c\) from the codomain. However, our second relation has no such doubles and is thus a function.

2. Function:

   ![Diagram showing a function]

Exercise 1. Practice distinguishing between functions and other types of relations here: [https://www.geogebra.org/m/fmferybu](https://www.geogebra.org/m/fmferybu).
There are a few different categories of functions. Today, we will look at three types of functions: injective, surjective and bijective.

**Definition 1. Injective Function**
An **injective** or **one-to-one** function is a relation in which each element from the domain is connected to a unique element from the codomain. In other words, if we pick any element from the codomain, it is either connected to exactly one or no elements from the domain.

**Definition 2. Surjective Function**
A **surjective** or **onto** function is one where every element in the codomain connects to an element from the domain. Thus, if we choose any element from the codomain, we can find at least one one element from the domain that it is connected to.

**Definition 3. Bijective Function**
A **bijective** function is a relation that is both injective and surjective. What this means is that each element in the domain is mapped to exactly one element in the codomain and vice versa. So, a bijection perfectly matches up all the elements in the two sets.

**Exercise 2.** Practice finding injective, surjective and bijective functions here: [https://www.geogebra.org/m/dh7yhu7u](https://www.geogebra.org/m/dh7yhu7u).
Bijections

We can use bijections to show that different sets are equal in cardinality, since we are making pairs of elements where each element in each set must have exactly one connection. The most useful bijective functions are ones that use a pattern to create the matching of pairs. For example:

In the first bijection, we change each number in the given element to a letter corresponding to its place in the alphabet. So, all 1’s become a’s, 2’s become b’s, etc. In the second, we remove the largest number from the element.

Exercise 3. Try this activity: https://www.geogebra.org/m/sv85rjrb to practice finding patterns in bijections.

Binary Strings

A string is a sequence of letters or numbers. A binary string is a string that is composed of only 1’s and 0’s. The length of a binary string is the total number of 1’s and 0’s in the string. For example, 101 is a binary string of length 3. We represent the empty binary string, or binary string of length 0 by \( \varepsilon \), the Greek letter epsilon.

We can represent sets of binary strings using decomposition. To understand decomposition, we first need a few definitions:

Definition 4. Concatenation

The binary string produced by concatenating two binary strings \( a \) and \( b \) is \( ab \). For example, the concatenation of 101 and 11 would be 10111. We can also concatenate sets of binary strings. \{1, 1\}\{01, 00, 0\} is the set of all binary strings that could be created by concatenating one string from the first set and one from the second. In our example, this is \{101, 100, 10, 1101, 1100, 110\}.

Definition 5. \( A^* \)

The \( * \)-operation represents the set of binary strings created by concatenating binary strings in set \( A \) with each other any number of times (including zero!). For example, \{1\}\(^* = \{\varepsilon, 1, 11, 111, 1111, 11111, \ldots \} \).
**Definition 6. Decomposition**

The decomposition of a set of binary strings is a way of using smaller sets, concatenation and the *-operator to express the larger set. For example, a decomposition of the set of all binary strings would be \( \{0, 1\}^* \), since at each step in writing a binary string, we choose to add another 0, 1 or stop. Note that decompositions are not unique: another way to decompose the set of all binary strings is \((\{1\} \ast \{0\}) \ast \{1\}^*\).

**Watch this video:**
https://youtu.be/cmZnorptwj0 for examples of determining if a given binary string can be created by a certain decomposition.

When creating binary string decompositions, we prefer to only have one way to create each binary string in the resultant set*. If there is more than one way to create a particular binary string from a decomposition, we call that decomposition ambiguous. For example, \(\{00, 0\}^*\) is ambiguous because I could create the binary string 00 from using 00 once or from sticking two 0’s together.

**Watch this video:**
https://youtu.be/HiiroCQK978 for an example of showing a decomposition is ambiguous.

*Note that the reason we prefer the decomposition to be unambiguous is to help with determining the number of strings in the set created by the decomposition. Doing this requires mathematics that you will learn in university and we won’t go into it here.

**Exercise 4.** Try this activity: https://www.geogebra.org/m/pe8bqsft to practice determining if a binary string is in the set represented by a given decomposition.

**Exercise 5.** Show that \(\{1, 11\}\{10, 0, 01\}^*\) is ambiguous. Check your answer by posting your response on Piazza!

**Bijection with Binary Strings:**

In this bijection, we change each 2 from the first set to a 1 and each pair of 1’s to a 0 to get the elements of the second set.