CS 70 Discrete Mathematics and Probability Theory
Summer 2022 Jingjia Chen, Michael Psenka and Tarang Srivastava **HW 4**

1 Lagrange? More like Lamegrange.

In this problem, we walk you through an alternative to Lagrange interpolation.

(a) Let's say we wanted to interpolate a polynomial through a single point, (x_0, y_0). What would be the polynomial that we would get? (This is not a trick question.)

(b) Call the polynomial from the previous part $f_0(x)$. Now say we wanted to define the polynomial $f_1(x)$ that passes through the points (x_0, y_0) and (x_1, y_1). If we write $f_1(x) = f_0(x) + a_1(x - x_0)$, what value of a_1 causes $f_1(x)$ to pass through the desired points?

(c) Now say we want a polynomial $f_2(x)$ that passes through (x_0, y_0), (x_1, y_1), and (x_2, y_2). If we write $f_2(x) = f_1(x) + a_2(x - x_0)(x - x_1)$, what value of a_2 gives us the desired polynomial?

(d) Suppose we have a polynomial $f_i(x)$ that passes through the points (x_0, y_0), ..., (x_i, y_i) and we want to find a polynomial $f_{i+1}(x)$ that passes through all those points and also (x_{i+1}, y_{i+1}). If we define $f_{i+1}(x) = f_i(x) + a_{i+1} \prod_{j=0}^{i} (x - x_j)$, what value must a_{i+1} take on?

Solution:

(a) We want a degree zero polynomial, which is just a constant function. The only constant function that passes through (x_0, y_0) is $f_0(x) = y_0$.

(b) By defining $f_1(x) = f_0(x) + a_1(x - x_0)$, we get that

$$f_1(x_0) = f_0(x_0) + a_1(x_0 - x_0) = y_0 + 0 = y_0.$$

So now we just need to make sure that $f_1(x_1) = y_1$. This means that we need to choose a_1 such that

$$f_1(x_1) = f_0(x_1) + a_1(x_1 - x_0) = y_1.$$

Solving this for a_1, we get that

$$a_1 = \frac{y_1 - f_0(x_1)}{x_1 - x_0}.$$

(c) We apply similar logic to the previous part. From our definition, we know that

$$f_2(x_0) = f_1(x_0) + a_2(x_0 - x_0)(x_0 - x_1) = y_0 + 0 = y_0.$$

and that

$$f_2(x_1) = f_1(x_1) + a_2(x_1 - x_0)(x_1 - x_1) = y_1 + 0 = y_1.$$

Thus, we just need to choose a_2 such that $f_2(x_2) = y_2$. Putting in our formula for $f_2(x)$, we get that we need a_2 such that

$$f_1(x_2) + a_2(x_2 - x_0)(x_2 - x_1) = y_2.$$

Solving for a_2, we get that

$$a_2 = \frac{y_2 - f_1(x_2)}{(x_2 - x_0)(x_2 - x_1)}.$$

(d) If we try to calculate $f_{i+1}(x_k)$ for $0 \le k \le i$, we know one of the $(x - x_j)$ terms (specifically the kth one) will be zero. Thus, we get that

$$f_{i+1}(x_k) = f_i(x_k) + a_{i+1}(0) = y_k + 0 = y_k.$$

So now we just need to pick a_i such that $f_{i+1}(x_{i+1}) = y_{i+1}$. This means that we need to choose a_{i+1} such that

$$f_i(x_{i+1}) + a_{i+1}\prod_{j=0}^{i}(x_{i+1} - x_j) = y_{i+1}.$$

Solving for a_{i+1}, we get that

$$a_{i+1} = \frac{y_{i+1} - f_i(x_{i+1})}{\prod_{j=0}^{i}(x_{i+1} - x_j)}.$$

The method you derived in this question is known as Newtonian interpolation. (The formal definition of Newtonian interpolation uses divided differences, which we don't cover in this class, but it's in effect doing the same thing.) This method has an advantage over Lagrange interpolation in that it is very easy to add in extra points that your polynomial has to go through (as we showed in part (c), whereas Lagrange interpolation would require you to throw out all your previous work and restart. However, if you want to keep the same x values but change the y values, Newtonian interpolation requires you to throw out all your previous work and restart. In contrast, this is fairly easy to do with Lagrange interpolation–since changing the y values doesn't affect the δ_is, you don't have to recalculate those, so you can skip most of the work.

2 How Many Polynomials?

Let $P(x)$ be a polynomial of degree at most 2 over GF(5). As we saw in lecture, we need $d+1$ distinct points to determine a unique d-degree polynomial, so knowing the values for say, $P(0)$, $P(1)$, and $P(2)$ would be enough to recover P. (For this problem, we consider two polynomials to be distinct if they return different values for any input.)

(a) Assume that we know $P(0) = 1$, and $P(1) = 2$. Now consider $P(2)$. How many values can $P(2)$ have? How many distinct possibilities for P do we have?

(b) Now assume that we only know $P(0) = 1$. We consider $P(1)$ and $P(2)$. How many different $(P(1), P(2))$ pairs are there? How many distinct possibilities for P do we have?

(c) Now, let P be a polynomial of degree at most d on $\mathrm{GF}(p)$ for some prime p with $p > d$. Assume we only know P evaluated at $k \leq d+1$ different values. How many different possibilities do we have for P?

(d) A polynomial with integer coefficients that cannot be factored into polynomials of lower degree on a finite field, is called an irreducible or prime polynomial.

Show that $P(x) = x^2 + x + 1$ is a prime polynomial on $\mathrm{GF}(5)$.

Solution:

(a) 5 polynomials, each for different values of $P(2)$.

(b) Now there are 5^2 different polynomials.

(c) p^{d+1-k} different polynomials. For $k = d+1$, there should only be 1 polynomial.

(d) We can try all possible inputs for x and show that in each case $P(x) \pmod{x} \neq 0$, which means that $P(x)$ does not have any root on the finite field $\mathrm{GF}(5)$.

$$\begin{aligned}
x = 0 &\Rightarrow P(0) \equiv 1 \pmod{5} \\
x = 1 &\Rightarrow P(1) \equiv 3 \pmod{5} \\
x = 2 &\Rightarrow P(2) \equiv 2 \pmod{5} \\
x = 3 &\Rightarrow P(3) \equiv 3 \pmod{5} \\
x = 4 &\Rightarrow P(4) \equiv 1 \pmod{5}
\end{aligned}$$

Hence $P(x)$ is a prime polynomial.

3 Monke

In the Monke Kingdom, there are seven bald uakaris, four black howlers, two mandrills, two gelada and one baboon. The Monke King has told the five species about an uncountably large supply of bananas; however, Monke doesn't want to reveal where the bananas are unless at least two species can cooexist in harmony.

(a) Help Monke create a scheme that allows the monkeys to find the key only if all members of one species in addition to at least one member from a different specie agree to work together.

For example, if all seven bald uakaris and one mandrill agree, they can find the key. However, if six bald uakaris and one mandrill agree, or if only seven bald uakaris agree, then the key cannot be found.

(b) Seeing that all the uakaris and all howlers have teamed up, Monke wants to encourage even more monkey friendship. He wants to set up an additional scheme so that, if at least one mandrill, one gelada and one baboon agree, the five parties can uncover exactly two more supplies of bananas, one countably infinite and one finite. How can Monke implement part 2 of their scheme?

Solution:

(a) Create a two part scheme, part one involving a species-wide secret, and part two involving a multi-monkey secret. In part one, if all the monkeys in one species agree work together, they can uncover species-wide secret S, with S being the same between all five species. To uncover S, give the n monkeys of a species each a distinct point on a $n-1$ degree polynomial. E.g Give all seven uakaris a distinct point on a degree-6 polynomial. Since there's only one baboon, give them S. The other species can find S as the y-intercept of their $n-1$ polynomial.

Part two involves finding a multi-monkey secret M. Give all monkeys of one species a point on a degree-1 polynomial, $P(x)$. For example, all howlers know P(1), all mandrills know P(2), etc. If any two monkeys from different species work together, they can uncover $P(x)$ and hence $M = P(0)$.

Finally, let M and S uniquely determine a degree-1 polynomial $R(x)$. Only if an entire species plus a monkey from a different species work together, can the group uncover both M and S, and find the key to the banana supply at R(0).

(b) Given that the uakaris and howlers have $R(x)$, they can uncover two more secrets if R(x) intercepts some degree-2 polynomial, $G(x)$ at two points. Give all mandrills, all geladas, all baboons one point on $G(x)$. For example, mandrills know $G(1)$, geladas know $G(2)$, etc... If one monkey from the three species work together, they can find $G(x)$, and if the uakaris and howlers work with the three monkeys, they can find the two intersection points and hence two more keys.

Alternatively, allow the five species can find a degree-4 polynomial with two distinct roots. Let the roots be the secrets.

4 That's Numberwang!

Congratulations! You've earned a spot on the game show "Numberwang".

(a) How many permutations of NUMBERWANG contain "GAMER" as a substring? How about as a subsequence (meaning the letters of "GAMER" have to appear in that order, but not necessarily next to each other)?

(b) In round 1 of Numberwang, each player chooses an ordered sequence of 5 digits. A valid sequence must have the property that it is non-increasing when read from left to right. For example, 99621 is a valid sequence, but 43212 is not. How many choices of valid sequences are there? (**Hint:** Relate the problem to balls and bins.)

(c) To win round 2 of Numberwang, a contestant must choose five nonnegative integers x_0, x_1, x_2, x_3, x_4 such that $x_0 + x_1 + x_2 + x_3 + x_4 = 100$, and $x_i \equiv i \pmod 5$. How many ways are there to pick a winning set of integers?

Solution:

(a) If we need "GAMER" to be a substring, then this is equivalent to finding the number of ways to arrange 6 items in a row ("GAMER", "N", "U", "B", "W", and "N"), where two of the items are the same (the two "N"s). This amounts to $6!/2! = 360$ ways.

If we need "GAMER" to be a subsequence, let's first choose which 5 positions contain the letters of GAMER; then, that uniquely determines the positions of G, A, M, E, R, since they have to be in that order. There are $\binom{10}{5}$ ways to do this, and $5!/2!$ ways to choose the positions of the remaining five letters (two of which are the same), so there are a total of $\binom{10}{5}\frac{5!}{2!} = 15120$ valid subsequences.

(b) This is actually a "balls and bins" (or "stars and bars") problem in disguise! We have five digits ("balls"), and 10 "bins" describing the values of the digits from 0 to 9: one bin for nines, one bin for eights, etc. This is because we know that the digits are non-increasing, so all the nines (if any) must come first, then all the eights (if any), and so on. So we just need to count how many ways there are to distribute the digits into the 10 bins: so our answer is $\binom{10+5-1}{10-1} = \binom{14}{9}$.

(c) Let $x_i = 5y_i + i$ for nonnegative integers y_i (we do this because of the modulo conditions). Then, the equation we must satisfy becomes $(5y_0 + 0) + (5y_1 + 1) + (5y_2 + 2) + (5y_3 + 3) + (5y_4 + 4) = 100$, which simplifies down to $y_0 + y_1 + y_2 + y_3 + y_4 = 18$ for nonnegative integers y_i. This is a standard balls and bins (or stars and bars) problem, with 18 balls and 5 bins, so our answer is $\binom{22}{4}$.

Why is $y_0 + y_1 + y_2 + y_3 + y_4 = 18$ a balls and bins problem? y_i is the number of balls in bin i, and 18 is the number of balls in total. Here are a few possible arrangements, with $\star$s representing balls:

$$\underbrace{\star\star\star\star\star\star}_{y_0=6} \mid \underbrace{\quad}_{y_1=0} \mid \underbrace{\star\star\star\star\star\star\star}_{y_2=7} \mid \underbrace{\star\star\star\star\star}_{y_3=5} \mid \underbrace{\quad}_{y_4=0}$$

$$\underbrace{\star\star}_{y_0=2} \mid \underbrace{\star\star\star\star}_{y_1=4} \mid \underbrace{\star\star\star\star}_{y_2=4} \mid \underbrace{\star\star\star\star\star\star}_{y_3=6} \mid \underbrace{\star\star}_{y_4=2}$$

Notice that we arrange the balls into 5 bins using 4 dividers. The total number of ways we can arrange 18 balls and 4 dividers side-by-side in a line is $\binom{22}{4}$. Why?

- 22 is the total number of objects in the line, 18 balls + 4 dividers.
- 4 is the number of dividers to be placed in the line.

"22 choose 4", or $\binom{22}{4}$, is the number of ways we can position 4 dividers on a line with 18 indistinguishable balls.

5 M, Mi, Mic, Mich, Micha, Michae, Michael!

In the vowel substring problem, we are tasked to find the number of vowels in every substring of an input string S. A substring is defined as any contingious set of characters in the original order of S. For example, $S = "aba"$, the substrings of S would be "a", "ab", "aba", "b", "ba", "a", counting a total of 6 vowels. We will make a combinatorial argument to count the number of vowels in all substrings.

- How many substrings are there in a string of length n?
- Suppose there is a vowel at 1st position of S (the beginning). Given that S has length n, how many substrings contain that vowel at the beginning of S?
- Suppose there is a vowel at the k^{th} position of S. How many substrings of S contain that vowel at the k^{th} position?
- Given that there are vowels in positions $i_1, i_2, ..., i_k$ what are the total number of vowels in all substrings of S?

Solution:

In the vowel substring problem, we are tasked to find the number of vowels in every substring of an input string S. A substring is defined as any contingious set of characters in the original order of S. For example, $S = "aba"$, the substrings of S would be "a", "ab", "aba", "b", "ba", "a", counting a total of 6 vowels. We will make a combinatorial argument to count the number of vowels in all substrings.

(a) We can define a substring to be two indices i and j such that $i \leq j$. Since there are n indices, there are $\binom{n}{2}$ unique ways to pick out i and j such that $i < j$. We add n in the end to account for $i = j$, giving us $\binom{n}{2} + n$ different substrings.

(b) There are n substrings that contain the first vowel. We can simply keep appending onto the first vowel until we reach the end of the string.

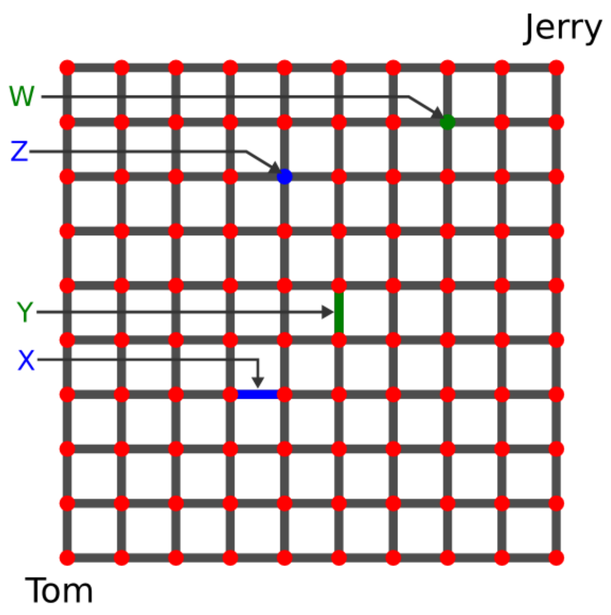
(c) We want to find all substrings such that $i \leq k \leq j$. There are k indices such that $i \leq k$, and there are $n - k + 1$ indices such that $k \geq j$, giving us a total of $k(n - k + 1)$ substrings.

(d) Each substring that has a vowel adds one to that specific vowel. Thus, for each vowel, we need to find all substrings that contain that vowel. Thus, summing together the number of substrings that contain the vowel at each k positions, the final answer is then:

$$\sum_{j=1}^{k} i_j(n - i_j + 1)$$

6 Maze

Let's assume that Tom is located at the bottom left corner of the 9×9 maze below, and Jerry is located at the top right corner. Tom of course wants to get to Jerry by the shortest path possible.



(a) What is the length of the shortest path from Tom to Jerry?

(b) How many such shortest paths exist?

(c) How many shortest paths pass through the edge labeled X? The edge labeled Y? Both the edges X and Y? Neither edge X nor edge Y?

Solution:

(a) 18. Each row in the maze has 9 edges, and so does each column. Any shortest path that Tom can take to Jerry will have exactly 9 horizontal edges going right (let's call these "H" edges) and 9 vertical edges going up (let's call these "V" edges).

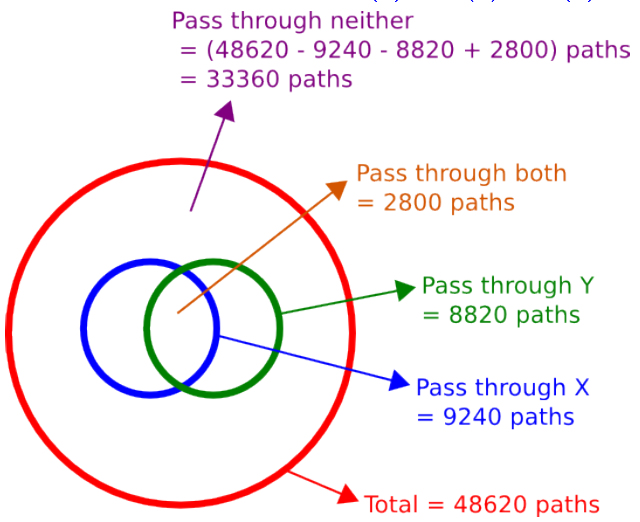
(b) Observe also that every shortest path from Tom to Jerry can be described by a unique sequence consisting of 9 "H"s and 9 "V"s. For example, one such path is HHHHHHHHHVVVVVVVVV (which represents the path that goes all the way to the right, and then all the way to the top). Conversely, every such sequence of exactly 9 "H"s and 9 "V"s corresponds to a unique shortest path from Tom to Jerry.

Therefore, the number of shortest paths is exactly the same as the number of ways of arranging 9 "H"s and 9 "V"s in a sequence, which is $\binom{18}{9} = 48620$.

(c) For a shortest path to pass through the edge X, it has to first get to the left vertex of X. So the first portion of the path has to start at the origin, and end at the left vertex of X. Using the same logic as above, there are exactly $\binom{6}{3} = 20$ ways to complete this "first leg" of the path (consisting of 3 "H" edges and 3 "V" edges). Having chosen one of these 20 ways, the path has to then go from the right vertex of X to the top right corner of the maze (the "second leg"). This second leg will consist of 5 "H" edges and 6 "V" edges, and using the same logic, there are exactly $\binom{11}{5} = 462$ possibilities. Therefore, the total number of shortest paths that pass through the edge X is $20 \times 462 = 9240$.

Using similar logic, any shortest path that passes through Y has to consist of 2 legs, the first leg going from the origin to the bottom vertex of Y, and the second leg going from the top vertex of Y to the top right corner of the maze. The first leg will consist of exactly 5 "H"s and 4 "V"s, while the second leg will consist of exactly 4 "H"s and 4 "V"s. So the total number of such shortest paths will be $\binom{9}{5} \times \binom{8}{4} = 8820$.

By a similar argument, let's try to figure out how many paths will pass through both X and Y. Clearly, any such path has to consist of 3 legs, with the first leg consisting of 3 "H"s and 3 "V"s (going from the origin to the left edge of X), the second leg consisting of 1 "H" and 1 "V" (going from the right vertex of X to the bottom vertex of Y), and the third leg consisting of 4 "H"s and 4 "V"s (going from the top vertex of Y to the top right corner of the maze). The total number of such shortest paths is therefore $\binom{6}{3} \times \binom{2}{1} \times \binom{8}{4} = 2800$.



Finally, we know that there are 48620 shortest paths in all, of which 9240 pass through X, 8820 pass through Y, and 2800 pass through both. So the number of paths that pass through neither is 33360 (see the figure above for an intuitive explanation).

7 Count It!

For each of the following collections, determine and briefly explain whether it is finite, countably infinite (like the natural numbers), or uncountably infinite (like the reals):

(a) The integers which divide 8.

(b) The integers which 8 divides.

(c) The functions from $\mathbb{N}$ to $\mathbb{N}$.

(d) The set of strings over the English alphabet. (Note that the strings may be arbitrarily long, but each string has finite length. Also the strings need not be real English words.)

(e) The set of finite-length strings drawn from a countably infinite alphabet, $\mathscr{A}$.

(f) The set of infinite-length strings over the English alphabet.

Solution:

(a) Finite. They are $\{-8,-4,-2,-1,1,2,4,8\}$.

(b) Countably infinite. We know that there exists a bijective function $f : \mathbb{N} \to \mathbb{Z}$. Then the function $g(n) = 8f(n)$ is a bijective mapping from $\mathbb{N}$ to integers which 8 divides.

(c) Uncountably infinite. We use Cantor's Diagonalization Proof:

Let $\mathscr{F}$ be the set of all functions from $\mathbb{N}$ to $\mathbb{N}$. We can represent a function $f \in \mathscr{F}$ as an infinite sequence $(f(0), f(1), \cdots)$, where the i-th element is $f(i)$. Suppose towards a contradiction that there is a bijection from $\mathbb{N}$ to $\mathscr{F}$:

$$\begin{aligned}
0 &\longleftrightarrow (f_0(0), f_0(1), f_0(2), f_0(3), \dots) \\
1 &\longleftrightarrow (f_1(0), f_1(1), f_1(2), f_1(3), \dots) \\
2 &\longleftrightarrow (f_2(0), f_2(1), f_2(2), f_2(3), \dots) \\
3 &\longleftrightarrow (f_3(0), f_3(1), f_3(2), f_3(3), \dots) \\
&\vdots
\end{aligned}$$

Consider the function $g : \mathbb{N} \to \mathbb{N}$ where $g(i) = f_i(i) + 1$ for all $i \in \mathbb{N}$. We claim that the function g is not in our finite list of functions. Suppose for contradiction that it were, and that it was the n-th function $f_n(\cdot)$ in the list, i.e., $g(\cdot) = f_n(\cdot)$. However, $f_n(\cdot)$ and $g(\cdot)$ differ in the n-th argument, i.e. $f_n(n) \neq g(n)$, because by our construction $g(n) = f_n(n) + 1$. Contradiction!

(d) Countably infinite. The English language has a finite alphabet (52 characters if you count only lower-case and upper-case letters, or more if you count special symbols – either way, the alphabet is finite).

We will now enumerate the strings in such a way that each string appears exactly once in the list. We will use the same trick as used in Lecture note 10 to enumerate the elements of $\{0,1\}^*$ We get our bijection by setting $f(n)$ to be the n-th string in the list. List all strings of length 1 in lexicographic order, and then all strings of length 2 in lexicographic order, and then strings of length 3 in lexicographic order, and so forth. Since at each step, there are only finitely many strings of a particular length ℓ, any string of finite length appears in the list. It is also clear that each string appears exactly once in this list.

(e) Countably infinite. The total number of programs is countably infinite, since each can be viewed as a string of characters (so for example if we assume each character is one of the 256 possible values, then each program can be viewed as number in base 256, and we know these numbers are countably infinite). So the number of halting programs, which is a subset of all programs, can be either finite or countably infinite. But there are an infinite number of halting programs, for example for each number i the program that just prints i is different for each i. So the total number of halting programs is countably infinite. (Note also that this result together with the previous one in (g) implies that not every function from $\mathbb{N}$ to $\mathbb{N}$ can be written as a program.)

(f) Countably infinite. Let $\mathscr{A} = \{a_1, a_2, \dots\}$ denote the alphabet. (We are making use of the fact that the alphabet is countably infinite when we assume there is such an enumeration.) We will provide two solutions:

Alternative 1: We will enumerate all the strings similar to that in part (b), although the enumeration requires a little more finesse. Notice that if we tried to list all strings of length 1, we would be stuck forever, since the alphabet is infinite! On the other hand, if we try to restrict our alphabet and only print out strings containing the first character $a \in \mathscr{A}$, we would also have a similar problem: the list

$$a, aa, aaa, \dots$$

also does not end.

The idea is to restrict *both* the length of the string and the characters we are allowed to use:

(a) List all strings containing only a_1 which are of length at most 1.

(b) List all strings containing only characters in $\{a_1, a_2\}$ which are of length at most 2 and have not been listed before.

(c) List all strings containing only characters in $\{a_1, a_2, a_3\}$ which are of length at most 3 and have not been listed before.

(d) Proceed onwards.

At each step, we have restricted ourselves to a finite alphabet with a finite length, so each step is guaranteed to terminate. To show that the enumeration is complete, consider any string s of length ℓ; since the length is finite, it can contain at most ℓ distinct a_i from the alphabet. Let k denote the largest index of any a_i which appears in s. Then, s will be listed in step $\max(k, \ell)$, so it appears in the enumeration. Further, since we are listing only those strings that have not appeared before, each string appears exactly once in the listing.

Alternative 2: We will encode the strings into ternary strings. Recall that we used a similar trick in Lecture note 10 to show that the set of all polynomials with natural coefficients is countable. Suppose, for example, we have a string: $S = a_5 a_2 a_7 a_4 a_6$. Corresponding to each of the characters in this string, we can write its index as a binary string: $(101, 10, 111, 100, 110)$.

Now, we can construct a ternary string where "2" is inserted as a separator between each binary string. Thus we map the string S to a ternary string: 101210211121002110. It is clear that this mapping is injective, since the original string S can be uniquely recovered from this ternary string. Thus we have an injective map to $\{0,1,2\}^*$. From Lecture note 10, we know that the set $\{0,1,2\}^*$ is countable, and hence the set of all strings with finite length over $\mathscr{A}$ is countable.

(g) Uncountably infinite. We can use a diagonalization argument. First, for a string s, define $s[i]$ as the i-th character in the string (where the first character is position 0), where $i \in \mathbb{N}$ because the strings are infinite. Now suppose for contradiction that we have an enumeration of strings s_i for all $i \in \mathbb{N}$: then define the string s' as $s'[i] =$ (the next character in the alphabet after $s_i[i]$), where the character after z loops around back to a. Then s' differs at position i from s_i for all $i \in \mathbb{N}$, so it is not accounted for in the enumeration, which is a contradiction. Thus, the set is uncountable.

Alternative 1: The set of all infinite strings containing only as and bs is a subset of the set we're counting. We can show a bijection from this subset to the real interval $\mathbb{R}[0,1]$, which proves the uncountability of the subset and therefore entire set as well: given a string in $\{a,b\}^*$, replace the as with 0s and bs with 1s and prepend $'0.'$ to the string, which produces a unique binary number in $\mathbb{R}[0,1]$ corresponding to the string.