Tandon Bridge Program - Homework 7

Archana Balachandran (ab12864) and Rachel Butler (rb3028)

November 2024

Question 3

a. 8.2.2

  b. f(n) = $n^3 + 3n^2 + 4$ and g(n) = n^3
  To prove that f(n) = $\Theta(n^3)$, we need to prove that f(n) is both $O(n^3)$ and $\Omega(n^3)$.
  For this proof, we assume $n^3 = g(n)$

  To prove $f = O(n^3)$, select $c_1 = 8$ and $n_0 = 1$. We will show that for any $n \geq 1$, $f(n) \leq 8.g(n)$.
  Since $n \geq 1$, $n^3 \geq 1$ and $n^3 \geq n^2$
  $n^3 + 3n^2 + 4(1)^3 \leq n^3 + 3n^3 + 4n^3$
  $n^3 + 3n^2 + 4(1)^3 \leq 8n^3$
  $f(n) \leq 8.g(n)$
  which means that f = O(g) or $f = O(n^3)$

  To prove $f = \Omega(n^3)$, select $c_2 = 1$ and $n_0 = 1$. We will show that for any $n \geq 1$, $f(n) \geq 1.g(n)$.
  Since $n \geq 1$
  $3n^2 + 4 \geq 0$
  And since $n \geq 1$, $n^3 \geq 1$. Adding n^3 on both sides of the inequality does not change the direction of the inequality.
  $n^3 + 3n^2 + 4 \geq n^3$
  $f(n) \geq 1.g(n)$
  which means that $f = \Omega(g)$ or $f = \Omega(n^3)$

  Since f is both $O(n^3)$ and $\Omega(n^3)$, f is $\Theta(n^3)$ for $n \geq 1$, $c_1 = 8$ and $c_2 = 1$.

b. 8.3.5

  a. The mystery algorithm reorders the sequence such that all values less than p are to the left of some breakpoint and followed by all values $\geq p$.

  b. i and j are being incremented and decremented respectively n-1 times in a sequence of length n. The answer does not depend on the actual values of the numbers in the sequence. It is only a function of the length of the sequence: f(n) = n-1 where n is the length of the sequence.

  c. The swap operation is maximized at $\frac{n}{2}$ times (rounded down if needed). This is when the values are in reverse sorted order, that is to say, all values less than p are towards the right side of all values greater than or equal to p.
  The number of swaps is minimized when the sequence is in sorted order, that is to say the original sequence has all values $< p$ to the left of all values $\geq p$

If the total number of elements is even in this scenario, the exact number of swaps would be $\frac{n}{2}$ whereas if the number of elements are odd, the number of swaps would be $\lfloor\frac{n}{2}\rfloor$.

d. The asymptotic lower bound of this algorithm is $\Omega(n)$.
It is generally important to take worst case input into consideration to ensure that one has not overestimated the best running time of the algorithm (i.e., one has set a reasonable lower bound).
For every input size in this algorithm, i and j are incremented/decremented n-1 times total, so for the best case scenario the program will run on $\Omega(n)$. In the case that you consider the worst case input, you'll end up with a total number of operations of n-1 + $\lfloor\frac{n}{2}\rfloor$. Given that the addition of the maximum swaps in this case doesn't impact the overall runtime, it's not important to include.

e. The upper-bound on the worst-case complexity of the algorithm is O(n).

Question 4

a. Exercise 5.1.2

b) Strings of lengths 7,8 or 9. Characters can be special characters, digits or letters.
Let S be the set of special characters, $|S| = 4$
Let D be the set of digits, $|D| = 10$
Let L be the set of letters, $|L| = 26$
The set of all allowed characters is $C \quad = \quad S \cup D \cup L$

Since $S \cap D \cap L$ is $\emptyset$, the sum rule can be applied to find the cardinality of C. $|C| = 4 + 10 + 26 = 40$
Let A_j denote the strings of length j over alphabet C. By the product rule, $A_j = 40^j$. For two values j and k, A_j and A_k are disjoint as a password cannot have length j and k at the same time. Therefore using the sum rule
$|A_7 \cup A_8 \cup A_9| = |A_7| + |A_8| + |A_9| = 40^7 + 40^8 + 40^9$

c) Strings of length 7, 8 or 9. Characters can be special characters, digits or letters. The first character cannot be a letter.
Compared to the previous part, this part specifies that the first character cannot be a letter. This reduces the number of available options for the first character to 14 (i.e. $|S| + |D|$).
Effectively for any A_j, the first character can be filled in 14 ways while the remaining j-1 characters can be filled in 40^{j-1} ways.
Following the same logic as above, this gives us $|A_7| + |A_8| + |A_9| = 14 * 40^6 + 14 * 40^7 + 14 * 40^8$

b. Exercise 5.3.2

a) In order to build a string of length 10 over the set of a,b,c, we use the generalized product rule, beginning with the first position.
The first position in this string can be filled in 3 ways - a,b or c.
The second position can only be filled in two different ways, because the letter used in the first position is not available to be used in the second as we cannot have two consecutive characters.
This logic applies to every subsequent position.
Combined, this gives us $3 * 2^9$ ways of creating strings of length 10, in which no two consecutive characters are the same.

c. Exercise 5.3.3

b) Each number plate has seven characters.
The first character is a digit from (0 through 9). Which means there are 10 ways of choosing the first character.

The next four characters are upper cases letters (A to Z). Each of these characters can be chosen in 26 ways, giving us 26^4 ways of filling the next four characters.
The next character is a digit, which is not a repetition of the first digit. Hence we have 9 ways of choosing this digit.
The final character is again a digit, with 8 ways of choosing this digit as it cannot be a repition of digits already used so far.
Combining all the ways of choosing digits and letters, we have a total of $10 * 26^4 * 9 * 8$ ways of creating a license plate number where there is no repition of digits.

c) To create a license plate number where no digit and letter appears more than once, we need to incorporate the fact that we will have one less way of choosing the next digit or letter.
First character: 10 ways of selecting a digit
Second character: 26 ways of selecting an upper case letter
Third character: 25 ways (as the letter already used cannot be used anymore) of selecting an upper case letter
Fourth character: 24 ways of selecting an upper case letter (Same logic as above)
Fifth character: 23 ways of selecting an upper case letter
Sixth character: 9 ways of choosing a digit (Leaving out the digit that has already been chosen as the first character)
Seventh character: 8 ways of choosing a digit (Leaving out the two digits already used)
Combining, there are 10*26*25*24*23*9*8 ways of choosing 7-character number plate such that no digit or letter appears more than once.

d. Exercise 5.2.3

a) B = $\{1, 0\}$
Method 1
B_9 is the set of binary strings with 9-bits. Each bit can be selected 2 ways, hence B_9 has a cardinality of 2^9.
E_{10} is the set of binary strings with n bits that have an even number of 1s. E_{10} would have had a cardinality of 2^{10} if we included both kinds of binary strings, those with an even number of 1s as well as odd number of 1s. When we choose only one type of parity (here, even), we reduce our selected pool by half.
Therefore E_{10} has a cardinality of $2^{10} \div 2 = 2^9$.
When X and Y are two finite sets, the function $f : X \longrightarrow Y$ is a 1-1 correspondence, or a bijection, if for every $y \in Y$, there is exactly 1 $x \in X$ such that f(x) = y. In this case, both B_9 and E_{10} have the same number of elements each, hence it is a bijection.

Method 2

B = $\{1, 0\}$
We will prove that there is a bijection between B^9 and E_{10} by first proving there is a one-to-one relationship and then onto.

Proving one-to-one:
Working with $f(x)$ and $f(y)$ in E_{10}, such that x and $y \in B^9$. We know that there are no repeating elements in B^9, because B^9 is the set of permutations of 9-digit binary strings.
We can establish a simple function:

If some string $z \in B^9$ has an even number of 1s in the string, we add a 0-bit to the end of the string
If some string $z \in B^9$ has an odd number of 1s in the string, we add a 1-bit to the end of the string

We can now solve for $f(x) and f(y)$ using the case that both have either even or odd parity.
If x and y have even parity:

$$f(x) = x + 0 \ f(y) = y + 0$$

If x and y have odd parity:

$$f(x) = x + 1 \ f(y) = y + 1$$

Since we know there aren't any repeats in the set B^9, when we know the results from above are all unique strings in E_{10}. The fact that $f(x)$ and $f(y)$ are not the same in either case proves the function is one-to-one.

Proving onto:
Assume there is some string, $a \in E_{10}$, where E_{10} is the set of all 10-digit strings with even parity. If we apply our function from above in reverse, we get:

If a ends in 1, removing the end digit, we get some string x in B^9.

We know that the resulting string, x, is in B^9, since we know B^9 includes all 9-digit binary strings. We have thus proven that all values in E_{10} map to a value in B^9, which means the function is onto.
We have now proven that the function is both one-to-one and onto, meaning it is a bijection.

b) We know if a function is a bijection, the domain and target have equal parity, thus we can say $|E_{10}| = 2^9 = 512$.

Question 5

a Exercise 5.4.2, sections a, b
At a certain university in the U.S., all phone numbers are 7-digits long and start with either 824 or 825.

   a How many different phone numbers are possible? $1 * 1 * 2 * 10^4 =$ 20,000

     Explanation:

     - The two ones represent the first two digits, 8 and 2, which are already chosen (i.e., there is only one choice for those spaces).
     - The two represents the third spot where it is said that there is either a 4 or 5.
     - The last four digits are represented by 10^4, because each of the four spaces can be filled with a digit 0-9.

   b How many different phone numbers are there in which the last four digits are all different?
     $1 * 1 * 2 * \frac{10!}{6!} =$ 10,080

     Explanation:

     - The two ones represent the first two digits, 8 and 2, which are already chosen (i.e., there is only one choice for those spaces).
     - The two represents the third spot where it is said that there is either a 4 or 5.
     - The last four digits are represented by $\frac{10!}{6!}$, because each digit must be different.

b Exercise 5.5.3, sections a-g

   a $2^{10} = 1024$

   b $2^7 = 128$

   c $2^7 + 2^8 = 384$

   d $2^8 = 256$

   e $\binom{10}{6} = 210$

   f $\binom{9}{6} = 84$

   g $\binom{5}{1} * \binom{5}{3} = 50$

c Exercise 5.5.5, section a
$\binom{30}{10} * \binom{35}{10}$

d Exercise 5.5.8, sections c-f

   c $\binom{26}{5} = 65,780$

   d $\binom{13}{1} * \binom{48}{1} = 624$

- e $\binom{13}{1} * \binom{4}{3} * \binom{12}{1} * \binom{4}{2} = 3744$
- f $\binom{13}{5} * \binom{4}{1} * \binom{4}{1} * \binom{4}{1} * \binom{4}{1} * \binom{4}{1} = \binom{13}{5} * 4^5 = 1{,}317{,}888$

e Exercise 5.6.6, sections a, b

- a $\binom{44}{5} * \binom{56}{5}$
- b $P(44, 2) * P(56, 2)$

Question 6

- a Exercise 5.7.2, sections a, b
  - a $\binom{52}{5} - \binom{39}{5}$
  - b $\binom{52}{5} - \binom{13}{5} * 4^5$
- b Exercise 5.8.4, sections a, b
  - a 5^{20}
  - b $\frac{20!}{4!^5}$

Question 7

a Zero. There is no possible way that a set with five elements could map one-to-one to a set with fewer elements, in this case 4.

b 5! = 120
Rationale: Let's say we have two random 5-element sets the domain, $L = a, b, c, d, e$, and target, $N = 1, 2, 3, 4, 5$. Taking the first element in L (a), there are 5 options for matching. Since we know one-to-one means one value in L will map to one value in N, we know that when we map the second element, there is one option fewer, since the first element was already mapped. Thus we know the second element will have four options for mapping. If you continue this pattern, you end up with 5*4*3*2*1 = 5!.

c P(6,5) = 720
Rationale: There are two key elements here:

(a) Six elements in the target, versus five in the domain, thus only five elements in the target will be mapped to from the domain (i.e., the cardinality of the range will be five).

(b) Order matters. We'll use the same domain from section b, $L = a, b, c, d, e$ and expand the target to $N = 1, 2, 3, 4, 5, 6$. If we take two random mappings of the same five elements from N, we could end up with:

$$\text{Iteration 1: } \{(a,1),(b,2),(c,3),(d,4),(e,5)\}$$
$$\text{Iteration 2: } \{(a,1),(b,2),(c,4),(d,3),(e,5)\}$$

Given that the first option would come from one function f and the second would come from some other function g we know that order matters.

Based on the above we know that we're dealing with the total permutations of a 6-element set when picking 5, including all orderings.

d P(7,5) = 2520
Rationale: We would apply the same logic as in section c, but with a target with seven elements, instead of six, still choosing only five elements to match the domain.