CPSC 418 / MATH 318 Introduction to Cryptography

ASSIGNMENT 1 — SOLUTION KEY

Written Problems for CPSC 418 and MATH 318

Problem 1 — Linear feedback shift register key streams (11 marks)

Stream ciphers such as the one-time pad require a secret key stream of pseudorandom bits.In this problem, you will cryptanalyze one possible approach for generating such a key stream.

Let m be a positive integer and $c_0, c_1, \ldots, c_{m-1} \in \{0, 1\}$ a sequence of m fixed bits. Let $z_0, z_1, \ldots, z_{m-1}$ be any sequence of m bits and define $z_m, z_{m+1}, z_{m+1}, \ldots$ via the linear recurrence

$$z_{n+m} \equiv c_{m-1}z_{n+m-1} + c_{m-2}z_{n+m-2} + \cdots + c_1 z_{n+1} + c_0 z_n \pmod 2 , \tag{1}$$

with the usual arithmetic modulo 2. The fixed bits $c_0, c_1, \ldots c_{m-1}$ are the *coefficients* of the linear recurrence (1) and the initial values $z_0, z_1, \ldots, z_{m-1}$ are its *seed*. If the seed and the coefficients are appropriately chosen, then (1) generates a sequence of 2^m pseudorandom bits[1] $(z_i)_{i\geq 0}$ from a seed of length m. This type of construction is popular since it can be implemented very efficiently in hardware using a *linear feedback shift register*; see pp. 36-37 of the Stinson-Paterson book.

a. (2 marks) Consider the recurrence

$$z_{n+5} \equiv z_{n+2} + z_n \pmod 2 ,$$

This represents the special case of (1) with $m = 5$ and $(c_4, c_3, c_3, c_1, c_0) = (0, 0, 1, 0, 1)$. Write down the first 31 bits $z_0, z_1, \ldots, z_{30}$ generated by this recurrence with seed $(z_0, z_1, z_2, z_3, z_4) = (0, 0, 1, 0, 1)$.

Solution.

$$0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1 \ .$$

Remark. As the footnote states, one expects a repeated pattern after $2^{32} - 1 = 31$ bits, and if you keep computing more bits, you will see that. However, there is no repeated pattern within the first 31 bits.

b. (6 marks) Suppose you are given the 10-bit sequence

$$(z_3, z_4, z_5, z_6, z_7, z_8, z_9, z_{10}, z_{11}, z_{12}) = (1, 0, 0, 1, 1, 0, 1, 0, 0, 1)$$

which was generated using an unknown linear recurrence of the form (1) with $m = 5$, i.e.

$$z_{n+5} \equiv c_4 z_{n+4} + c_3 z_{n+3} + c_2 z_{n+2} + c_1 z_{n+1} + c_0 z_n \pmod 2 \ .$$

Find the unknown coefficients c_0, c_1, c_2, c_3, c_4 of this recurrence. Your answer *must use a systematic approach* and should make no assumptions or distinguish cases; e.g. solutions that include

[1] Since there are only $2^m - 1$ distinct non-zero bit patterns of length m, there must be repetition after at most $2^m - 1$ bits. So in practice, m must be large.

a phrase such as "suppose $c_3 = 0$" and then either derive a contradiction or obtain the values of c_0, c_1, c_2, c_4 will get zero credit. Show your work and remember your linear algebra!

Suggestion: check your answer, i.e. ensure that the coefficients you obtain define a recurrence that produces the second five given bits $(z_8, z_9, z_{10}, z_{11}, z_{12}) = (0, 1, 0, 0, 1)$ from the first four given bits $(z_3, z_4, z_5, z_6, z_7) = (1, 0, 0, 1, 1)$.

Note: the hardest part of this problem is probably typesetting it properly; hence the high mark score. So this is a good problem on which to hone your LaTeX skills. Use math display environments like `align*` to nicely line up your congruences and learn how to typeset matrices.

Solution. The last five bits are related to the first five as follows

$$\begin{aligned}
z_8 &\equiv c_4 z_7 + c_3 z_6 + c_2 z_5 + c_1 z_4 + c_0 z_3 \pmod 2 \ , \\
z_9 &\equiv c_4 z_8 + c_3 z_7 + c_2 z_6 + c_1 z_5 + c_0 z_4 \pmod 2 \ , \\
z_{10} &\equiv c_4 z_9 + c_3 z_8 + c_2 z_7 + c_1 z_6 + c_0 z_5 \pmod 2 \ , \\
z_{11} &\equiv c_4 z_{10} + c_3 z_9 + c_2 z_8 + c_1 z_7 + c_0 z_6 \pmod 2 \ , \\
z_{12} &\equiv c_4 z_{11} + c_3 z_{10} + c_2 z_9 + c_1 z_8 + c_0 z_7 \pmod 2 \ .
\end{aligned}$$

Substituting the values for $z_3, \ldots, z_{12}$ yields the system of congruences

$$\begin{aligned}
0 &\equiv c_4 + c_3 + c_0 \pmod 2 \ , \\
1 &\equiv c_3 + c_2 \pmod 2 \ , \\
0 &\equiv c_4 + c_2 + c_1 \pmod 2 \ , \\
0 &\equiv c_3 + c_1 + c_0 \pmod 2 \ , \\
1 &\equiv c_2 + c_0 \pmod 2 \ .
\end{aligned}$$

This system can be solved via standard linear algebra techniques using arithmetic modulo 2. We write the system as a matrix-vector congruence

$$\begin{pmatrix} 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} c_4 \\ c_3 \\ c_2 \\ c_1 \\ c_0 \end{pmatrix} \equiv \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \pmod 2 \ .$$

and and convert the augmented matrix to row echelon form via row operations like you learned in MATH 211:

$$\left(\begin{array}{ccccc|c} 1&1&0&0&1&0\\ 0&1&1&0&0&1\\ 1&0&1&1&0&0\\ 0&1&0&1&1&0\\ 0&0&1&0&1&1 \end{array}\right) \xrightarrow{\text{add row 1 to row 3}} \left(\begin{array}{ccccc|c} 1&1&0&0&1&0\\ 0&1&1&0&0&1\\ 0&1&1&1&1&0\\ 0&1&0&1&1&0\\ 0&0&1&0&1&1 \end{array}\right) \xrightarrow{\text{add row 2 to rows 3 and 4}}$$

$$\left(\begin{array}{ccccc|c} 1&1&0&0&1&0\\ 0&1&1&0&0&1\\ 0&0&0&1&1&1\\ 0&0&1&1&1&1\\ 0&0&1&0&1&1 \end{array}\right) \xrightarrow{\text{add row 4 to row 5}} \left(\begin{array}{ccccc|c} 1&1&0&0&1&0\\ 0&1&1&0&0&1\\ 0&0&0&1&1&1\\ 0&0&1&1&1&1\\ 0&0&0&1&0&0 \end{array}\right) \xrightarrow{\text{add row 3 to row 5}}$$

$$\left(\begin{array}{ccccc|c} 1&1&0&0&1&0\\ 0&1&1&0&0&1\\ 0&0&0&1&1&1\\ 0&0&1&1&1&1\\ 0&0&0&0&1&1 \end{array}\right) \xrightarrow{\text{swap rows 3 and 4}} \left(\begin{array}{ccccc|c} 1&1&0&0&1&0\\ 0&1&1&0&0&1\\ 0&0&1&1&1&1\\ 0&0&0&1&1&1\\ 0&0&0&0&1&1 \end{array}\right) .$$

At this point, you can either continue with row operations to convert the matrix to reduced row echelon form, or convert the system back to congruences and solve via back substitution. Here, I will opt for the latter:

$$\begin{aligned} c_4 + c_3 + c_0 &\equiv 0 \pmod{2} , \\ c_3 + c_2 &\equiv 1 \pmod{2} , \\ c_2 + c_1 + c_0 &\equiv 1 \pmod{2} , \\ c_1 + c_0 &\equiv 1 \pmod{2} , \\ c_0 &\equiv 1 \pmod{2} . \end{aligned}$$

Substituting the fifth congruence into the fourth, the fourth one into the third, the third into the second and finally the second into the first yields successively

$$c_0 = 1 , \quad c_1 = 0 , \quad c_2 = 0 , \quad c_3 = 1 , \quad c_4 = 0 ,$$

which produces the recurrence

$$z_{n+5} \equiv z_{n+3} + z_n \pmod{2} .$$

Generating $(z_8, z_9, z_{10}, z_{11}, z_{12})$ from the seed $(z_3, z_4, z_5, z_6, z_7) = (1, 0, 0, 1, 1)$ via this recurrence produces $(z_8, z_9, z_{10}, z_{11}, z_{12}) = (0, 1, 0, 0, 1)$, which agrees with the values for $z_8, z_9, z_{10}, z_{11}, z_{12}$ given above. So this is the correct recurrence relation.

Common errors:

- Not using matrix algebra. You can combine the equations, but if it gets too messy, it really doesn't match the requirement of the systematic approach anymore.
- Not doing the arithmetic modulo 2. The solution to this question should not contain a single fraction!

c. (3 marks) Suppose you are intercepting a bit stream which you know as generated via a recurrence relation of the form (1) with $m = 5$. Through the method illustrated in parts (b) and (c), you were able to find out that the recurrence is

$$z_{n+5} \equiv z_{n+4} + z_{n+3} \pmod{2} \qquad (n \geq 0)\ . \tag{2}$$

Now suppose this bit stream is used to generate keys for encryption with the one-time pad. You intercept the 10-bit ciphertext fragment $C = 1001101000$ that was one-time pad encrypted using as key a 10-bit sequence $(z_0, z_1, \ldots, z_9)$, where $(z_0, z_1, z_2, z_3, z_4)$ is an unknown seed and the 5-tuple $(z_5, z_6, z_7, z_8, z_9)$ was obtained from this seed using the recurrence (2). Suppose also that you discover that the last two bits of the seed (i.e. z_3 and z_4) are 1 and 0, respectively. Decrypt as many bits of C as possible.

Solution. The corresponding 10-bit plaintext M is the bitwise x-or, or equivalently, the bitwise addition modulo 2, of the ciphertext with the key, i.e.

$$M = (m_0, m_1, \ldots, m_9) = (1001101000) \oplus (z_0, z_1, \ldots, z_9)\ .$$

Starting with the two known seed bits and substituting iteratively into (2), we find

$$\begin{aligned}
z_3 &= 1\ , \\
z_4 &= 0\ , \\
z_5 &\equiv z_4 + z_3 \equiv 0 + 1 \equiv 1 \pmod{2}\ , \\
z_6 &\equiv z_5 + z_4 \equiv 1 + 0 \equiv 1 \pmod{2}\ , \\
z_7 &\equiv z_6 + z_5 \equiv 1 + 1 \equiv 0 \pmod{2}\ , \\
z_8 &\equiv z_7 + z_6 \equiv 0 + 1 \equiv 1 \pmod{2}\ , \\
z_9 &\equiv z_8 + z_7 \equiv 1 + 0 \equiv 1 \pmod{2}\ .
\end{aligned}$$

Hence

$$(m_0, m_1, \ldots, m_9) = (1001101000) \oplus (z_0, z_1, z_2, 1, 0, 1, 1, 0, 1, 1) \equiv (1 + z_0, z_1, z_2, 0, 1, 1, 0, 0, 1, 1)\ .$$

So the last 7 bits of M are 0110011. The first 3 bits of M cannot be determined; depending on the specific values of z_0, z_1 and z_2, they can take on any values.

Problem 2 — Password counts (15 marks)

In each question below, provide a brief explanation, a formula and the numerical value for your answer (exact if the answer is an integer, approximate otherwise).

There are 94 *printable characters* on a standard North American keyboard, comprised of the 26 upper case letters A-Z, the 26 lower case letters a-z, the 10 numerical digits 0-9 and the 32 special characters

‘ ’ " . , ; : ! ? ~ @ # $ % ^ & * _ - + = ( ) { } [ ] < > \ / |

For our purposes, *passwords* are strings consisting of printable characters. The *length* of a password is the number of characters in the password.

a. (1 mark) What is the total number of passwords of length 8?

   Solution. Each of the 8 characters in the password can be one of the 94 printable characters. So the total number of passwords of length 8 is

$$94^8 = 6{,}095{,}689{,}385{,}410{,}816;$$

   about six quadrillion.

b. (3 marks) Suppose a user has a password of length 8 whose first four characters are the first four letters of their oldest child's name (all in either lower or upper case) and the second four characters are the child's birthday in the format[2] DDMM. Intelligence gathering on your part reveals that the child's first name starts with ‘L’ and that the child was born in 2008. Making no further assumption about a "reasonable" first name (e.g., for all you know, the kid's name could be ‘Lxqscdx’), what is the minimal number of candidates that our lazy user could potentially be using as their password?

   Solution. The first character of the password is ‘l’ or ‘L’, and the choice dictates whether the next three characters are lower or upper case. Either way, there are 26 choices for each of the second, third and fourth characters. 2008 is a leap year and thus has 366 days, so there are 366 date combinations for the last four characters. So the total number of candidates for the password is

$$2 \cdot 26^3 \cdot 366 = 12{,}865{,}632\ .$$

   A computer can brute-force search a space of less than thirteen million objects in no time.

   Common errors:

   - Overly verbose answers. You really only need to say that 2008 is a leap year to obtain the count of 366 days.
   - Not taking into account that the first four letters may or may not be capitalized (and we don't know which is the case), which results in the solution being off by a factor of 2.
   - Overcounting. "Minimal" in this question means that based on the information we have, we should only count the potential candidates for passwords, no more.

[2]For example, if a user's oldest child's name was Robin and Robin's birthday was on July 1, then the user's password would be either ROBI0107 or robi0107, but not Robi0107 or rObI0107.

- Counting combinations rather than permutations; order is important here.

c. (4 marks) To guard against *dictionary attacks* (where a password cracker checks if a password is a common word or phrase), passwords are typically required to contain at least one numerical digit and at least one special character. What is the total number of passwords of length 8 that satisfy this requirement?

(*Hint:* It is easier to characterize the number of passwords that violate this rule and subtract that count from the total number of 8-character passwords. But be careful that you don't subtract out some passwords twice.)

Solution. Following the hint, the rule forbids passwords that contain no numerical digit or no special characters (note that the negation changes the "and" in the question to an "or".

In passwords that contain no digit, each character is one of 52 upper or lower case letters or one of 32 special characters. Since $52 + 32 = 84$, that count of such 8-character passwords comes to 84^8.

Similarly, the number of length 8 passwords without a special character is $(52 + 10)^8 = 62^8$.

Now comes the tricky part. The final answer to this question is *not* $94^8 - 84^8 - 62^8$ because passwords that contain neither a a digit nor a special character are included in both the counts of 84^8 and 62^8. So subtracting $84^8 + 62^8$ from the total of 94^8 would remove these types of passwords twice. So we need to add these back in. Passwords that contain neither a digit nor a special character consist of letters only, so there are 52^8 such passwords of length 8. Thus, the final answer is

$$94^8 - 84^8 - 62^8 + 52^8 = 3{,}452{,}050{,}097{,}274{,}880 .$$

d. (2 marks) What is the percentage of 8-character passwords that satisfy the rule of part (c)?

Solution. This is simply the quotient of the answers to parts (c) and (a) converted to a percentage. We have

$$\frac{3{,}452{,}050{,}097{,}274{,}880?}{6{,}095{,}689{,}385{,}410{,}816} \approx 0.5663 .$$

So just under 57% of all length 8 passwords satisfy the rule of part (b).

e. (2 marks) Assuming that each permissable character in a password is chosen equally likely[3], what is the entropy of the password space of

- part (a)?
- part (c)?

Solution. Choosing each character with equal likelihood means that each password is equally likely. Thus, the entropy is $\log_2(n)$ bits where n is the number of possible passwords. This comes to

$$\log_2(94^8) = 8\log_2(94) \approx 52.44 \text{ bits for part (a)} ,$$
$$\log_2(3{,}452{,}050{,}097{,}274{,}880) \approx 51.62 \text{ bits for part (c)}$$

[3]This assumption may be appropriate for passwords chosen by computer systems with good random number generators, but it is utterly false for passwords chosen by humans. So in practice, the minimum password length computed in part (f) is a significant underestimate.

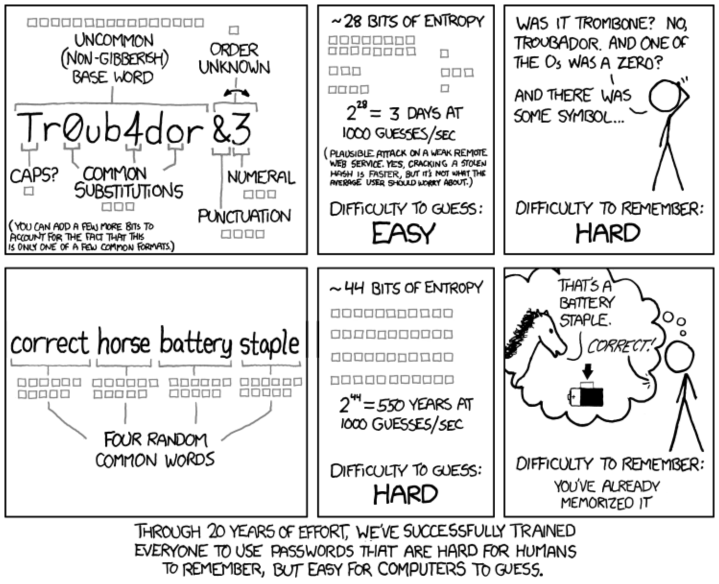
f. (3 marks) Suppose we want a password space with entropy 128 bits, assuming that keys are chosen equally likely, i.e. a total of 2^{128} passwords (this number is typical for key space sizes of modern cryptosystems). Assuming no restrictions on the characters appearing in passwords (i.e. the scenario of part (a)), what is the minimum password length that guarantees a password space with entropy 128?

   Solution. Let ℓ be the (unknown) number of characters in a password. Then there are 94^ℓ passwords of length ℓ. That number is supposed to be equal to 2^{128}, so

$$\begin{aligned} 94^\ell &= 2^{128} \\ \ell \log_2(94) &= \log_2(2^{128}) = 128 \\ \ell &= 128/\log_2(94) \approx 19.53 \end{aligned}$$

   Since ℓ is an integer, we need passwords with at least 20 characters, with each character chosen with equal likelihood from the set of printable characters, to guarantee entropy 128.

Concluding remark: As of March 2019, NIST has completely moved away from password complexity rules and the imposition of password expiration periods that force users to change their passwords regularly and frequently (see NIST SP 800-63). The real world is slow to follow that wisdom, including our very own U of C. The following XKCD comic, taken from https://xkcd.com/936/, illustrates why:


UNCOMMON (NON-GIBBERISH) BASE WORD
ORDER UNKNOWN
Tr0ub4dor&3
CAPS?
COMMON SUBSTITUTIONS
NUMERAL
PUNCTUATION
(YOU CAN ADD A FEW MORE BITS TO ACCOUNT FOR THE FACT THAT THIS IS ONLY ONE OF A FEW COMMON FORMATS.)
~28 BITS OF ENTROPY
2^28 = 3 DAYS AT 1000 GUESSES/SEC
(PLAUSIBLE ATTACK ON A WEAK REMOTE WEB SERVICE. YES, CRACKING A STOLEN HASH IS FASTER, BUT IT'S NOT WHAT THE AVERAGE USER SHOULD WORRY ABOUT.)
DIFFICULTY TO GUESS: EASY
WAS IT TROMBONE? NO, TROUBADOR. AND ONE OF THE Os WAS A ZERO?
AND THERE WAS SOME SYMBOL...
DIFFICULTY TO REMEMBER: HARD
correct horse battery staple
FOUR RANDOM COMMON WORDS
~44 BITS OF ENTROPY
2^44 = 550 YEARS AT 1000 GUESSES/SEC
DIFFICULTY TO GUESS: HARD
THAT'S A BATTERY STAPLE.
CORRECT!
DIFFICULTY TO REMEMBER: YOU'VE ALREADY MEMORIZED IT
THROUGH 20 YEARS OF EFFORT, WE'VE SUCCESSFULLY TRAINED EVERYONE TO USE PASSWORDS THAT ARE HARD FOR HUMANS TO REMEMBER, BUT EASY FOR COMPUTERS TO GUESS.

Problem 3 — Weak collisions (16 marks)

The cryptographic relevance of this problem will become evident when we cover hash functions in class.

For each question below, provide a brief explanation and a compact formula for your answer.

Let n be positive integer. Consider an experiment involving a group of participants, where we assign each participant a number that is randomly chosen from the set $\{1, 2, \ldots, n\}$ (so all these assignments are independent events). Note that we allow for the possibility of assigning the same number to two different participants.

Now pick your favourite number N between 1 and n. When any one of the participants is assigned the number N, we refer to this as a *weak collision* (with N). In this problem, we determine how to ensure at least a 50% chance of a weak collision in our experiment.

a. (2 marks) What is the probability that a given participant is assigned your favourite number N?

   Solution. There are n equally likely outcomes (number assignments), so this probability is $1/n$ (regardless of the choice of participant and of the number N).

b. (2 marks) What is the probability that a given participant is not assigned the number N?

   Solution. This event is the complement of the event of part (a). So the answer is one minus the probability of having received N as your assignment, i.e. $1 - 1/n$ by part (a).

c. (3 marks) Suppose k people participate in the experiment (for some positive integer k). What is the probability that none of them is assigned the number N, i.e. that there is no weak collision?

   Solution. Since the assignments represent independent events, the probability of this joint event is equal to the product of the probabilities of each individual event, where an individual event represents a person not being assigned the number 1. By part (b), the probability of no one being assigned the number N is thus

$$P = \left(1 - \frac{1}{n}\right)^k .$$

d. (4 marks) Intuitively, the more people participate, the likelier we encounter a weak collision. We wish to find the minimum number K of participants required to ensure at least a 50% chance of a weak collision.

   Suppose $n = 10$. What is the threshold K in this case? Give a numerical value that is an integer.

   Solution. By part (d), we need to find the integer K such that

$$1 - \left(1 - \frac{1}{10}\right)^K \geq 0.5 > 1 - \left(1 - \frac{1}{10}\right)^{K-1} .$$

   There are now two approaches to finding K. One is by trial and error, simply computing the powers of 0.9 (here, I have computed them to two decimal places):

k	0	1	2	3	4	5	6	7
$1-(0.9)^k$	0	0.1	0.19	0.27	0.34	0.41	0.47	0.52

The quantities $1-(0.9)^k$ stay below 0.5 up to exponent $k \leq 6$ and move above 0.5 starting at exponent $k \geq 7$. So $K = 7$.

A better way is to solve for K analytically, but be sure to handle the inequalities correctly. We have

$$\begin{aligned} 1-(0.9)^K &\geq 0.5 > 1-(0.9)^{K-1} \\ -(0.9)^K &\geq -0.5 > -(0.9)^{K-1} \\ (0.9)^K &\leq 0.5 < (0.9)^{K-1} \\ K\log(0.9) &\leq \log(0.5) < (K-1)\log(0.9) \ . \end{aligned}$$

We now divide by $\log(0.9)$. Note that this quantity is negative (regardless of the base of the log). Dividing by a negative number switches inequalities, i.e. '$\geq$' will change to '$\leq$' and '$>$' will become '$<$'":

$$K \geq \frac{\log(0.5)}{\log(0.9)} > K-1 \ .$$

Now $\log(0.5)/\log(0.9) \approx 6.5788\ldots$ (again, regardless of what base you are using for your log), so we find again that $K = 7$.

This means that allow at least 7 participants, there is at least a 50% chance that one of them is assigned N; for 6 or fewer, we can no longer guarantee such a 50% chance.

e. (5 marks) Generalizing part (d) from $n = 10$ to arbitrary n, prove that if the number of participants is above $\log(2)n \approx 0.69n$, then there is at least a 50% chance of a weak collision. Use (without proof) the inequality[4]

$$1 - x < \exp(-x) \quad \text{for } x > 0 \ . \tag{3}$$

Solution. Let K be the number of participants, with $K \geq \log(2)n$. Then we have following chain of equivalences:

$$\begin{aligned} K &\geq \log(2)n \\ K/n &\geq \log(2) \\ -K/n &\leq -\log(2) = \log(0.5) \\ \exp(-K/n) &\leq 0.5 \\ (\exp(-1/n))^K &\leq 0.5 \ . \end{aligned}$$

Since $1/n > 0$, (3) implies that $\exp(-1/n) < 1 - 1/n$. It follows that

$$\left(1 - \frac{1}{n}\right)^K \leq 0.5 \ .$$

[4]This inequality comes from the Taylor series $\exp(-x) = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \frac{x^4}{4!} - \cdots + \cdots$, and the sum of the terms from $x^2/2$ onwards is positive.

The left hand side of this inequality is precisely the probability of part (c), i.e. the probability P that there is no weak collision with K participants. Thus,

$$\begin{aligned} P &\leq 0.5 \\ -P &\geq -0.5 \\ 1 - P &\geq 1 - 0.5 = 0.5 \ . \end{aligned}$$

Since $1 - P$ is the probability that a weak collision occurs, the claim is proved.

Common errors: mostly to do with proof writing and working with inequalities.

- Starting a proof with your assumption and then working to the bottom to get a tautology (i.e. a statement that is always true) does not constitute a proof.
- Assuming the inequality $P \leq 0.5$ that is to be proved and then working backwards to prove $K \geq \log(n)$ is also not a proof. That is proving the converse of the statement (you are supposed to prove $A \Rightarrow B$, but you are instead proving $B \Rightarrow A$ which is logically not an equivalent statement).
- Introducing "convenient" assumptions partway through a proof is also not allowed.
- Working with inequalities can be tricky. For example, multiplying an inequality by -1 flips the inequality, i.e. $a < b$ if and only if $-a > -b$. In equalities are transitive, i.e. $a < b$ and $b < c$ implies $a < c$; however, if you know that $a < b$ and $c < b$, you cannot infer any relationship between a and c. Substituting one equality into another also needs to be done with care. e.g. if $a < b$ and $c < d - a$, then you cannot infer $c < d - b$.

Concluding Remark. In (3), when x is small, $\exp(-x)$ is very close to $1 - x$; for example, $\exp(0.01)$ and $1 - 0.01$ are within 4 decimal places of each other. This implies that for large n, the threshold $0.69n$ of part (e) is close to optimal, i.e. we expect the necessary and sufficient number of participants for ensuring a weak collision with at least 50% probability to be on the order of n. Compare this to the corresponding result for (strong) collisions derived in the next problem.

Problem 4 — (Strong) collisions (20 marks)

The cryptographic relevance of this problem will become evident when we cover hash functions in class.

For each question below, provide a brief explanation and a compact formula for your answer.

We consider the same experiment as in the previous problem, although here, you don't need to pick a favourite number N. When at least two of the participants are assigned identical numbers, we refer to this as a *strong collision* or just a *collision*. In this problem, we determine how to ensure at least a 50% chance of a collision in our experiment.

a. (4 marks) What is the probability that among k participants, no collision occurs?

   Solution. The requirement is that all participants are assigned different numbers. For ease of description, we label the participants $0, 1, 2, \ldots, k-1$. Participant 0 can be assigned any number between 1 and n. For Participant 1 to be assigned a number that is different from the one assigned to participant 0, there are $n-1$ choices left out of the n numbers as possible assignments. For participant 2, there are $n-2$ choices left (ruling out the two numbers that participants 0 and 1 were assigned), and so on. Finally, there are $n-(k-1) = n-k+1$ choices out of n numbers left for participant $k-1$. Since probabilities of independent events multiply, the probability that all k participants are assigned different numbers is

$$P = \frac{n}{n} \cdot \frac{n-1}{n} \cdot \frac{n-2}{n} \cdots \frac{n-k+1}{n} = \frac{(n-1)(n-2)\cdots(n-k+1)}{n^k} = \frac{n!}{n^k(n-k)!} .$$

   An alternative expression (see part (g)) is

$$P = \prod_{i=1}^{k-1} \left(1 - \frac{i}{n}\right) .$$

b. (2 marks) What is the probability of a collision among k participants?

   Solution. This event is the complement of the event in part (a), so the answer is

$$1 - P = 1 - \frac{n!}{n^k(n-k)!} .$$

c. (4 marks) Once again, intuitively, the more people participate, the likelier we encounter a collision. We wish to find the minimum number K of participants required to ensure at least a 50% chance of a collision.

   Suppose $n = 10$. What is the the threshold K in this case? Give a numerical value that is an integer.

   Solution. By part (g), we want to find the integer K such that

$$1 - \frac{10!}{10^K(10-K)!} \geq 0.5 > 1 - \frac{10!}{10^{K-1}(10-(K-1))!} .$$

   Unlike part (d) of the previous problem, we cannot solve this analytically since the unknown quantity K appears inside factorials and in exponents. So trial and error it is. We compute sufficiently good approximations to the quantities $1 - 10!/(10^k(10-k)!)$ for $k = 0, 1, 2, \ldots$ and find:

k	0	1	2	3	4	5
$1 - 10!/(10^k(10-k)!)$	0	0	0.1	0.28	0.496	0.7

The expressions $1 - 10!/(10^k(10-k)!)$ stay below 0.5 up to exponent $k \le 4$ and move above 0.5 starting at exponent $k \ge 5$. So $K = 5$. This means that if we allow 5 or more participants, there is at least a 50% chance of a collision; if we allow 4 or fewer, we can no longer guarantee such a 50% chance.

d. (5 marks) Let P be the probability of no collisions as computed in part (a). Prove that

$$P \le \exp\left(-\frac{k(k-1)}{2n}\right) .$$

First, if your expression obtained for P in part (a) is not already in this form, rewrite P as

$$P = \prod_{i=1}^{k-1}(1 - z_i) ,$$

where $0 < z_i < 1$ for $1 \le i \le k-1$ (you'll have to figure out the appropriate expression for z_i). Then use inequality (3) from the previous problem.

Solution. The quantities z_i referred to above are $z_i = i/n$ for $1 \le i \le k-1$. From part (a), we have

$$\begin{aligned} P &= \frac{n-1}{n} \cdot \frac{n-2}{n} \cdots \frac{n-(k-1)}{n} \\ &= \left(1 - \frac{1}{n}\right)\left(1 - \frac{2}{n}\right) \cdots \left(1 - \frac{k-1}{n}\right) \\ &\le \exp\left(-\frac{1}{n}\right) \exp\left(-\frac{2}{n}\right) \cdots \exp\left(-\frac{k-1}{n}\right) \\ &= \exp\left(-\frac{1}{n} - \frac{2}{n} - \cdots - \frac{k-1}{n}\right) \\ &= \exp\left(-\frac{1}{n}\sum_{i=1}^{k-1} i\right) \\ &= \exp\left(-\frac{1}{n}\frac{k(k-1)}{2}\right) = \exp\left(-\frac{k(k-1)}{2n}\right) . \end{aligned}$$

Equality holds if and only if $k = 1$, in which case $P = \exp(-1(1-1)/2n) = 1$.

e. (5 marks) Similarly to the previous problem, when n is much larger than the number k of participants, the upper bound on P in part (d) is a very close approximation to P. When in addition k is not too small (but still much smaller than n), k is very close to $k-1$. This means that the quantity $\exp(-k^2/2n)$ is a very close approximation to the probability P of part (a).

Generalizing part (c) from $n = 10$ to arbitrary n, prove that if the number of participants is above $\sqrt{\log(4)n} \approx 1.177\sqrt{n}$, then there is at least a 50% chance of a collision. You may replace the actual expression for P derived in part (a) by $\exp(-k^2/2n)$.

Note: you can solve this question even if you didn't attempt parts (a)-(d).

Solution. Let K be the number of participants, with $K \geq \sqrt{\log(4)n}$. Then we have following chain of equivalences:

$$\begin{aligned}
K &\geq \sqrt{\log(4)n} \\
K^2 &\geq \log(4)n = 2\log(2)n \\
K^2/2n &\geq \log(2) \\
-K^2/2n &\leq -\log(2) = \log(0.5) \\
\exp(-K^2/2n) &\leq 0.5 \ .
\end{aligned}$$

Identifying the expression on the left hand side of the last inequality with the probability P of no collisions from part (a), we see that when $K \geq \sqrt{\log(4)n}$, the probability P of no collisions is at most 0.5:

$$\begin{aligned}
P &\leq 0.5 \\
-P &\geq -0.5 \\
1 - P &\geq 1 - 0.5 = 0.5 \ .
\end{aligned}$$

Since $1 - P$ is the probability that a collision occurs, the claim is proved.

Common errors: very similar errors to Problem 3 (e).

Remark. For $n = 365$, the answer to part (e) is known as the *birthday paradox* which is actually not a paradox, but a potentially counter-intuitive fact. 365 is the number of possible birthdays, disregarding leap years. Here, we obtain $K \approx 1.177 \cdot \sqrt{365} \approx 22.49$. So in a group of as few as 23 people, there is a 50-50 chance that two of them have the same birthday. Most people would guess the required number of people required to be much higher than 23.

Concluding Remark. As in the previous problem, for large n, the threshold $1.177\sqrt{n}$ of part (e) is close to optimal, i.e. we expect the necessary and sufficient number of participants for ensuring a collision with at least 50% probability to be on the order of $\sqrt{n}$. Compare this to the corresponding threshold of $0.69n$ (i.e. of order n) for weak collisions derived in the previous problem. Strong collisions are expected to be much more likely. E.g. drawing just 12 numbers between 1 and 100 is more likely than not to produce a strong collision, whereas it takes as many as 69 draws to make a weak collision more likely than not.

Written Problem for MATH 318 only

Submit your answers to this problem in a separate PDF file. **Do *not* include the solution to this problem in the PDF file containing your solutions to Problems 1-4.**

Problem 5 — A time-memory trade-off for key search (38 marks + 4 bonus marks)

This problem investigates a time-memory trade-off for exhaustive key search of a cipher. Normally, searching a key space of size n takes up to n test decryptions but requires no memory as every wrong decryption (not yielding the valid plaintext) can immediately be discarded. The strategy in this problem requires only t such tests, at the expense of storing on the order of n/t keys (though frequently fewer keys than that). A drawback is that the search strategy requires a very expensive pre-computation of tn tests. However, this pre-computation need only be done once and its results can be re-used for an arbitrary number of subsequent key searches. Here, t is a *trade-off parameter*, which is a number between 2 and n chosen by the attacker. The choice of t needs to balance the feasibility of the pre-computation, the amount of available memory, and the amount of time to be spent on key search.

We assume that the message space, ciphertext space and key space for this cryptosystem are all identical, i.e. $\mathcal{M} = \mathcal{C} = \mathcal{K}$. For brevity, put $n = |\mathcal{K}|$. We also assume that for every plaintext/ciphertext pair (M, C), there is a unique key K such that $E_K(M) = C$. Examples of ciphers that satisfy these properties include the shift cipher, the one-time pad, the Advanced Encryption Standard (AES) and many others.

Our attack scenario is a known plaintext attack, where the adversary has a plaintext/ciphertext pair (M, C) and the task is to search for the unique unknown key K such that $E_K(M) = C$. We define a function

$$g : \mathcal{K} \to \mathcal{K} \quad \text{via} \quad g(K) = E_K(M) ,$$

where we note that the co-domain of g is $\mathcal{C} = \mathcal{K}$. So we are searching for the key $K \in \mathcal{K}$ such that $g(K) = C$.

In order to understand and illustrate the attack, we will use as an example a cryptosystem that represents a modification of the shift cipher on an alphabet of $n = 51$ characters. Specifically, $\mathcal{M} = \mathcal{C} = \mathcal{K} = \mathbb{Z}_{51} = \{0, 1, \ldots, 50\}$ (the integers modulo 51) and

$$E_K(M) \equiv M + 8K \pmod{51} , \qquad D_K(C) \equiv C - 8K \equiv C + 43K \pmod{51} \tag{4}$$

for all $M, C, K \in \mathbb{Z}_{51}$.

Some parts of this problem ask you to compute quantities specific to the this sample cipher. All parts not stating explicitly that you should use the cipher defined in (4) should be solved in general, i.e. for any cipher as described above. *Even if you are having difficulties answering the general questions, you are urged to attempt the parts that refer specifically to the cipher of (4).*

a. (2 marks) Prove that the map g is a bijection on $\mathcal{K}$.

Solution. We prove injectivity. Let $K_1, K_2 \in \mathcal{K}$ with $g(K_1) = g(K_2)$. Then $E_{K_1}(M) = E_{K_2}(M)$. By assumption, for any ciphertext C, there is excactly one key K with $E_K(M) = C$. So $E_{K_1}(M) = E_{K_2}(M)$ implies $K_1 = K_2$; hence g is injective.

Since the domain and co-domain of g are finite sets of the same cardinality (in fact, they are even the same set), g must also be surjective. Another way to see this is to consider the set $g(\mathcal{K})$ of all images under g. Since g is injective, all the images $g(K), K \in \mathcal{K}$ are distinct, so there are n of them. Since this set is a subset of $\mathcal{K}$, these images must make up all of $\mathcal{K}$, so g is surjective.

Common errors:

- Forgetting to state that the domain and codomain are *finite* when inferring surjectivity from injectivity or vice versa.
- Assuming that an arbitrary element in the codomain is of the form $C = E_K(M)$. This is what you are asked to prove. E.g. for the sample cipher, an arbitrary element in the codomain is just an element in $\mathbb{Z}_{51}$ and you need to prove that it is indeed of the form $C = E_K(M)$.

b. As usual, for $K \in \mathcal{K}$ and any $i \in \mathbb{Z}$, let $g^i(K)$ denote

- g applied i times to K when $i > 0$,
- K when $i = 0$ (i.e. g^0 is the identity on $\mathcal{K}$),
- the inverse g^{-1} applied $|i|$ times to K when $i < 0$.

For any $K \in \mathcal{K}$, the *orbit* of K under g is the subset

$$\mathcal{O} = \{g^i(K) \mid i \in \mathbb{Z}\} .$$

Since $\mathcal{K}$ is finite, orbits are actually finite sets containing at most n elements.

(i) (5 marks) Prove that for every orbit $\mathcal{O}$, there exists a positive integer $n_{\mathcal{O}}$ such that $\mathcal{O}$ can be written in the form

$$\mathcal{O} = \{K, g(K), g^2(K), \ldots, g^{n_{\mathcal{O}}-1}(K)\} = \{g^k(K) \mid 0 \leq k \leq n_{\mathcal{O}} - 1\} , \qquad (5)$$

where K is any key in $\mathcal{O}$ and the elements $g^i(K)$ with $0 \leq k \leq n_{\mathcal{O}} - 1$ are all distinct. The number $n_{\mathcal{O}}$ of elements in $\mathcal{O}$ is called the *length* of $\mathcal{O}$.

Solution. Let $\mathcal{O}$ be any orbit and let $K \in \mathcal{O}$. Since $\mathcal{O}$ is finite, the elements $g^i(K)$ with $i \geq 0$ cannot all be distinct. So there exist minimal integers $j < k$, dependent on K, such that $g^j(K) = g^k(K)$. Here, "minimal' means that all the elements $g^i(K)$ with $0 \leq i < k$ are distinct and the first repetition appears at $g^k(K)$ which matches an earlier element $g^j(K)$. Then $g^{k-j}(K) = K$. Putting $m = k - j$, we have $g^m(K) = K$ and the elements $K, g(K), \ldots g^{m-1}(K)$ are all distinct.

The integer m depends on K. So we need to show that m is also the minimal positive integer such that $g^m(K') = K'$ for all the other keys $K' \in \mathcal{O}$. To that end, let $K' \in \mathcal{O}$. We first establish that $g^m(K') = K'$, then we show minimality. By definition of orbits, $K' = g^i(K)$ for some $i \in \mathbb{Z}$. It follows that

$$g^m(K') = g^m(g^i(K)) = g^{m+i}(K) = g^i(g^m(K)) = g^i(K) = K'$$

as asserted. Now assume by way of contradiction that $g^r(K') = K'$ for some integer r with $0 < r < m$. Then

$$K = g^{-i}(K') = g^{-i}(g^r(K')) = g^{r-i}(K') = g^r(g^{-i}(K')) = g^r(K) .$$

But $g^r(K) = K$ with $0 < r < m$ contradicts the minimality of m. So m is the minimal positive integer such that $g^m(K) = K$ for all $K \in \mathcal{O}$. We can now simply define $n_{\mathcal{O}}$ as $n_{\mathcal{O}} = m$.

(ii) (3 marks) Prove that $\mathcal{K}$ is the union of disjoint orbits.

Solution. Every key $K \in \mathcal{K}$ belongs to some orbit; for example, the orbit $\mathcal{O} = \{g^i(K) \mid i \in \mathbb{Z}\}$ (since $K = g^0(K)$). It follows that $\mathcal{K}$ is the union of orbits.

To prove that distinct orbits are disjoint, let $\mathcal{O}, \mathcal{O}'$ be two orbits and write $\mathcal{O} = \{g^i(K) \mid i \in \mathbb{Z}\}$, $\mathcal{O}' = \{g^i(K') \mid i \in \mathbb{Z}\}$ for two keys $K, K' \in \mathcal{K}$. Suppose $\mathcal{O}$ and $\mathcal{O}'$ have a common element $X \in \mathcal{O} \cap \mathcal{O}'$. Then $X = g^j(K) = g^k(K')$ for some integers j, k.

Now $K = g^{k-j}(K') \in \mathcal{O}'$, so $g^i(K) = g^{i+k-j}(K')$ also belongs to $\mathcal{O}'$ for all integers i. Thus, every element in $\mathcal{O}$ belongs to $\mathcal{O}'$; in other words, $\mathcal{O} \subseteq \mathcal{O}'$. A symmetric argument shows that $\mathcal{O}' \subseteq \mathcal{O}$: we have $K' = g^{j-k}(K)$ and hence $g^i(K') = g^{i+j-k}(K) \in \mathcal{O}$ for all $i \in \mathbb{Z}$. Hence $\mathcal{O} = \mathcal{O}'$. It follows that any two orbits with a common element are identical. Hence any two distinct orbits are disjoint, so $\mathcal{K}$ is the union of disjoint orbits.

Another way to prove this assertion is to define a relation $\sim$ on $\mathcal{K}$ according to which $K \sim K'$ if and only if $K = g^i(K')$ for some $i \in \mathbb{Z}$. This is easily shown to be an equivalence relation on $\mathcal{K}$ whose equivalence classes are precisely the orbits. Since every set with an equivalence relation is the disjoint union of equivalence classes, the result follows.

(iii) (3 marks) Compute the orbits for the cipher given in (4) using $M = 1$. For every orbit $\mathcal{O}$, start with the smallest element $K \in \mathbb{Z}_{51}$ and write down the distinct elements in $\mathcal{O}$ in the order $K, g(K), g^2(K), \ldots$

Solution. By part (b) (i), every orbit has the form $\mathcal{O} = \{K, g(K), g^2(K), \ldots, g^{n_{\mathcal{O}}}(K)\}$ where we can choose K to be the smallest key in $\mathcal{O}$. Also, distinct orbits are disjoint by part (b) (ii), so we only need to compute the orbits of keys that are not already contained in a previously computed orbit.

It is easy to symbolically compute orbits for any M and any K:

$$\begin{aligned}
g(K) &= E_K(M) \equiv M + 8K \pmod{51} ,\\
g^2(K) &= g(g(K)) \equiv g(M + 8K) \equiv E_{M+8K}(M) \equiv M + 8(M + 8K) \equiv 9M + 13K \pmod{41} ,\\
g^3(K) &= g(g^2(K)) \equiv g(9M + 13K) \equiv E_{9M+13K}(M) \equiv M + 8(9M + 13K) \equiv 22M + 2K \pmod{51} ,\\
g^4(K) &= g(g^3(K)) \equiv \cdots \equiv 24M + 16K \pmod{51} ,\\
g^5(K) &\equiv 20M + 16K \pmod{51} ,\\
g^6(K) &\equiv 15M + 4K \pmod{51} \pmod{51} ,\\
g^7(K) &\equiv 19M + 32K \pmod{51} ,\\
g^8(K) &\equiv M + 8(19M + 32K) \equiv 0 \cdot M + 1 \cdot K \equiv K \pmod{51} ,
\end{aligned}$$

So every orbit has length at most 8, but some will turn out to have length less than 8. Substituting $M = 1$ and starting every orbit at the smallest key not already appearing in a previous orbit, we obtain

$$\begin{aligned}
\mathcal{O}_0 &= \{0, 1, 9, 22, 24, 40, 15, 19\} \ , \\
\mathcal{O}_2 &= \{2, 17, 35, 26, 5, 41, 23, 32\} \ , \\
\mathcal{O}_3 &= \{3, 25, 48, 28, 21, 16, 27, 13\} \ , \\
\mathcal{O}_4 &= \{4, 33, 10, 30, 37, 42, 31, 45\} \ , \\
\mathcal{O}_6 &= \{6, 49, 36, 34, 18, 43, 39, 7\} \ , \\
\mathcal{O}_8 &= \{8, 14, 11, 38, 50, 44, 47, 20\} \ , \\
\mathcal{O}_{12} &= \{12, 46\} \ , \\
\mathcal{O}_{29} &= \{29\} \ .
\end{aligned}$$

c. Let t be the trade-off parameter. Call an orbit $\mathcal{O}$ *short* if its length is at most t (i.e. $n_{\mathcal{O}} \leq t$) and *long* otherwise (i.e. $n_{\mathcal{O}} \geq t + 1$).

(i) (1 mark) For $t = 3$, identify the short and long orbits for the cipher given in (4).

Solution. $\mathcal{O}_{12}$ and $\mathcal{O}_{29}$ are short, while $\mathcal{O}_0$, $\mathcal{O}_2$, $\mathcal{O}_3$, $\mathcal{O}_4$, $\mathcal{O}_6$ and $\mathcal{O}_8$ are long.

(ii) (2 marks) Prove that the number of long orbits[5] is at most $n/(t + 1)$.

Solution. Let $\mathcal{O}_1, \mathcal{O}_2, \ldots, \mathcal{O}_r$ be all the distinct long orbits with respective lengths $n_{\mathcal{O}_1}, n_{\mathcal{O}_2} \ldots, n_{\mathcal{O}_r}$. Then

$$n \geq \sum_{i=1}^{r} n_{\mathcal{O}_i} \geq \sum_{i=1}^{r} (t + 1) = r(t + 1) \ ,$$

so $r \leq n/(t + 1)$.

d. For any key $K \in \mathcal{K}$, define the set

$$\mathcal{R}(K) = \{g^t(K), g^{2t}(K), \ldots, g^{q_{\mathcal{O}}t}(K)\} = \{g^{it}(K) \mid 1 \leq i \leq q_{\mathcal{O}}\} \ ,$$

where $n_{\mathcal{O}}$ is the length of the orbit $\mathcal{O}$ containing K and $q_{\mathcal{O}}$ is the unique integer such that $(q_{\mathcal{O}} - 1)t < n_{\mathcal{O}} \leq q_{\mathcal{O}}t$ (or equivalently, $(q_{\mathcal{O}} - 1)t + 1 \leq n_{\mathcal{O}} \leq q_0 t$ as $n_{\mathcal{O}}$ and $(q_{\mathcal{O}} - 1)t$ are integers).

(i) (1 mark) Prove that for every key $K \in \mathcal{K}$, $\mathcal{R}(K)$ has cardinality at most $(n_{\mathcal{O}} - 1)/t + 1$, where $n_{\mathcal{O}}$ is the length of $\mathcal{O}$.

Solution. The number of elements in each $\mathcal{R}(K)$ is q where $n_{\mathcal{O}} \geq (q_{\mathcal{O}} - 1)t + 1$. From this inequality, we obtain $q_{\mathcal{O}} \leq (n_{\mathcal{O}} - 1)/t + 1$ after rearranging.

(ii) (3 marks) For $t = 3$, compute the collection of sets $\mathcal{R}(K)$ of the cipher given in (4), Specifically, for each $\mathcal{O}$ computed in part (b) (iii), write down the elements of $\mathcal{R}(K)$ in the order $g^3(K), g^6(K), \ldots$, where K is the smallest element of $\mathcal{O}$.

Solution. All long orbits $\mathcal{O}$ have length 8. The unique integer q such that $3(q-1)+1 \leq 8 < 3q$ is $q = 3$. So for the keys K contained in long orbits, $\mathcal{R}(K)$ contains three elements,

[5]This bound can be very crude in practice if all the long orbits are much longer than $t + 1$. For our sample cipher (4) for example, we have $n/(t + 1) = 51/4 = 12.75$, but the number of long orbits is significantly less than 12.

namely the keys $g^3(K)$, $g^6(K)$ and $g^9(K) = g(K)$, where K is the smallest key in $\mathcal{O}$. We can read these elements off of $\mathcal{O}$ as the fourth, seventh and second element in $\mathcal{O}$.

In general, for short orbits, we have $0 \leq n_{\mathcal{O}} < t$ and hence $q_{\mathcal{O}} = 1$. So for keys K in contained in short orbits, $R(K)$ is a singleton set containing only the element $g^t(K)$. We obtain

$$\begin{aligned}
\mathcal{R}(0) &= \{22, 15, 1\} \ , \\
\mathcal{R}(2) &= \{26, 23, 17\} \ , \\
\mathcal{R}(3) &= \{28, 27, 25\} \ , \\
\mathcal{R}(4) &= \{30, 31, 33\} \ , \\
\mathcal{R}(6) &= \{34, 39, 49\} \ , \\
\mathcal{R}(8) &= \{38, 47, 14\} \ , \\
\mathcal{R}(12) &= \{46\} \ , \\
\mathcal{R}(29) &= \{29\} \ .
\end{aligned}$$

e. Let $\mathcal{O}_1, \mathcal{O}_2, \ldots, \mathcal{O}_r$ be all the distinct *long* orbits. For each i with $1 \leq i \leq r$, fix a key $k_i \in \mathcal{O}_i$ and put

$$\mathcal{R} = \mathcal{R}(k_1) \cup \mathcal{R}(k_2) \cup \cdots \cup \mathcal{R}(k_r) \ .$$

(i) (2 marks) Compute the set $\mathcal{R}$ for the cipher given in (4) using the keys k_i where, as before, each k_i is the smallest element in its (long) orbit. List the elements of $\mathcal{R}$ in ascending order.

Solution. We take the union of the first six sets computed in part (e) and sort them in ascending order to obtain

$$\mathcal{R} = \{1, 14, 15, 17, 22, 23, 25, 26, 27, 28, 30, 31, 33, 34, 38, 39, 47, 49\}.$$

(ii) (3 marks) Let $K \in \mathcal{K}$ be a key contained in some long orbit. Prove that there exists $R \in \mathcal{R}$ with $R \neq K$ such that $R \in \{g(K), g^2(K), \ldots, g^t(K)\}$. Intuitively, this means that every key $K \in \mathcal{K}$ is either contained in an orbit of length at most t or is at most t applications of g away from from an element in $\mathcal{R}$.

Solution. Let $\mathcal{O}$ be the orbit containing K (this orbit is unique as distinct orbits are disjoint). Suppose $k \in \mathcal{O}$ is the key used to construct the set $\mathcal{R}(k) \subset \mathcal{R}$, arising from the representation of $\mathcal{O}$ as $\mathcal{O} = \{k, g(k), \ldots, g^{n_{\mathcal{O}}-1}(k)\}$ according to part (b) (i). Since $K \in \mathcal{O}$, we have $K = g^i(k)$ for some $i \in \mathbb{Z}$ with $0 \leq i \leq n_{\mathcal{O}} - 1$.

Write $i = qt + r$ with $q, r \in \mathbb{Z}$ and $0 \leq r < t$. Then $q \geq 0$ (as $i \geq 0$), so $(q + 1)t \geq t$, and $qt \leq i < n_{\mathcal{O}} \leq q_{\mathcal{O}}t$, so $q < q_{\mathcal{O}}$ and hence $q + 1 \leq q_{\mathcal{O}}$. Now put $R = g^{(q+1)t}(k)$. Then $R \in \mathcal{R}(k)$ as $1 \leq q + 1 \leq q_{\mathcal{O}}$. Also

$$R = g^{qt+t}(k) = g^{i-r+t}(k) = g^{t-r}(g^i(k)) = g^{t-r}(K) \ ,$$

and $1 \leq t - r \leq t$. So $R \in \{g(K), g^2(K), \ldots, g^t(K)\}$ as claimed.

(iii) (4 marks) Prove that $\mathcal{R}$ has cardinality[6] at most $2n/(t + 1)$.

[6] As pointed out in the previous footnote, much of the time the set $\mathcal{R}$ is much smaller because the bound $2n/(t+1)$ here uses a bound on the number of long orbits that is frequently a significant overestimate.

Solution. As in part (c) (ii), let $\mathcal{O}_1, \mathcal{O}_2, \ldots, \mathcal{O}_r$ be all the distinct *long* orbits, of respective lengths $n_{\mathcal{O}_1}, n_{\mathcal{O}_2}, \ldots, n_{\mathcal{O}_r}$. Then

$$\sum_{i=1}^{r} n_{\mathcal{O}_i} \leq n \ .$$

Now the number r of long orbits is at most $n/(t+1)$ by part (c) (ii), and each set $R(k_i)$ corresponding to a long orbit $\mathcal{O}_i$ has cardinality at most $(n_{\mathcal{O}_i} - 1)/t + 1$ by part (d) (i). Hence

$$\begin{aligned}|\mathcal{R}| &\leq \sum_{i=1}^{r} \left(\frac{n_{\mathcal{O}_i} - 1}{t} + 1 \right) = \sum_{i=1}^{r} \frac{n_{\mathcal{O}_i}}{t} + \sum_{i=1}^{r} \left(-\frac{1}{t} + 1 \right) \\ &\leq \frac{n}{t} + r\left(1 - \frac{1}{t}\right) \leq \frac{n}{t} + \frac{n}{t+1}\left(1 - \frac{1}{t}\right) = n\left(\frac{1}{t} + \frac{1}{t+1}\left(1 - \frac{1}{t}\right)\right) \ .\end{aligned}$$

Now

$$\frac{1}{t} + \frac{1}{t+1}\left(1 - \frac{1}{t}\right) = \frac{1}{t} + \frac{1}{t+1} - \frac{1}{t(t+1)} = \frac{(t+1) + t - 1}{t(t+1)} = \frac{2}{t+1} \ .$$

Hence $\mathcal{R}$ has cardinality at most $2n/(t+1)$.

f. Define the set

$$\mathcal{T} = \{(K, g^{-t}(K)) \mid K \in \mathcal{R}\} \ .$$

(i) (2 marks) Compute the set $\mathcal{T}$ for $t = 3$ for the cipher defined in (4) using the set $\mathcal{R}$ computed in part (e) (i). List the elements of $\mathcal{T}$ in ascending order of first coordinate, i.e. in the same order as the elements in $\mathcal{R}$.

Solution. We have $\mathcal{T} = \{K, g^{-3}(K) \mid K \in \mathcal{R}\} = \{K, g^{5}(K) \mid K \in \mathcal{R}\}$. For each key $K \in \mathcal{R}$, is is easy to find $g^5(K)$ by a cyclic forward shift of 5 in the corresponding orbit, when the elements of the orbit are ordered $K, g(K), g^2(K), \ldots$ as was done in part (b) (iii). Then we just need to order the pairs according to their first coordinate K. We obtain

$$\begin{aligned}\mathcal{T} = \{&(1,15),(14,47),(15,22),(17,23),(22,0),(23,26),(25,27),(26,2),(27,28),\\ &(28,3),(30,4),(31,30),(33,31),(34,6),(38,8),(39,34),(47,38),(49,39)\}\end{aligned}$$

(ii) **Bonus** (4 marks) Describe a procedure that computes the sets $\mathcal{R}$ and $\mathcal{T}$ using no more than nt application of the function g and without having to store all n keys (or any other array of size n) in memory. Prove that your procedure computes $\mathcal{R}$ and $\mathcal{T}$ correctly and complies with the specified time and memory restrictions.

g. Consider the following search procedure that takes as input the pair (M, C) and the sets $\mathcal{R}, \mathcal{T}$ and finds the key K such that $C = E_K(M)$:

Starting with $K_0 = C$, compute the elements $K_1 = g(K_0)$, $K_2 = g(K_1)$, …until one one of the following conditions is satisfied:

A) If an index $i > 0$ is found such that $K_i = C$, output $K = K_{i-1}$.

B) If an index $i > 0$ is found such that $K_i \in \mathcal{R}$, look up the entry with first coordinate K_i in $\mathcal{T}$ and put $X = g^{-t}(K_i)$ (the second coordinate of that entry in $\mathcal{T}$). Starting with $K'_0 = X$, compute the elements $K'_1 = g(K'_0)$, $K'_2 = g(K'_1)$, …until a key K'_j is found such that $K'_j = C$. Output $K = K'_{j-1}$.

(i) (3 marks) Execute the search procedure for the cipher given in (4) with $t = 3$ using the ciphertexts $C = 12$, $C = 29$ and $C = 37$. For each C, indicate which of the conditions (A) or (B) is saisfied, and write down all the keys K_i computed during the search, as well as the output key K.

Suggestion: check your output keys against (4) to make sure that you have found the correct key that encrypts $M = 1$ to C.

Solution. For $C = 12$, the algorithm computes $K_0 = 12$, $K_1 = 46$, $K_2 = 12$ and finds that $K_2 = C$, which is condition (A) with $i = 2$. So it outputs $K = K_1 = 46$. Note that $E_{46}(1) \equiv 1 + 8 \cdot 46 \equiv 369 \equiv 12 \pmod{51}$.

For $C = 29$, the algorithm computes $K_0 = 29$, $K_1 = 29$ and finds that $K_1 = C$, which is condition (A) with $i = 1$. So it outputs $K = K_0 = 29$. Note that $E_{29}(1) \equiv 1 + 8 \cdot 29 \equiv 233 \equiv 29 \pmod{51}$.

For $C = 37$, the algorithm computes $K_0 = 37$, $K_1 = 42$, $K_2 = 31$, and finds that $K_2 = 31 \in \mathcal{R}$, which is condition (B) with $i = 2$. Looking up the entry whose first coordinate is 31 in $\mathcal{T}$ yields $X = 30$. Now the algorithm starts with $K'_0 = 30$ and computes $K'_1 = 37 = C$. So the procedure outputs $K'_0 = 30$. Note that $E_{30}(1) \equiv 241 \equiv 37 \pmod{51}$.

(ii) (4 marks) Prove that the search procedure outputs the correct key and performs no more than t evaluations of the map g.

Note: the computational effort of comparing keys to C in condition (A), checking whether $C \in \mathcal{R}$ in condition (B) and looking up the entry in $\mathcal{T}$ that has first coordinate C as per condition (B) is generally negligible compared to the evaluations of g. These kinds of operations on finite sets can be implemented very efficiently via hash tables, for example. So we ignore that cost in our analysis.

Solution. Suppose first that C belongs to a short orbit $\mathcal{O}$. Then

$$\mathcal{O} = \{C, g(C), g^2(C), \ldots, g^{n_{\mathcal{O}}-1}(C)\} ,$$

by part (b) (i), where $n_{\mathcal{O}} \le t$. The algorithm thus finds a match $K_i = C$ with $i = n_{\mathcal{O}}$ in condition (A) above. Note that $E_{K_{i-1}}(M) = g(K_{i-1}) = K_i = C$, so the output K_{i-1} is the correct key that encrypts M to C. Moreover, the algorithm computes the $n_{\mathcal{O}}$ elements $g(C), g^2(C), \ldots g^{n_{\mathcal{O}}}(C)$. Since $n_{\mathcal{O}} \le t - 1$, the algorithm performs at most $t - 1$ evaluations of g.

Suppose now that C belongs to a long orbit $\mathcal{O}$. By part (e) (ii), there exists $R \in \mathcal{R}$ with $R \ne C$ such that $R \in \{g(C), g^2(C), \ldots, g^t(C)\} = \{K_1, K_2, \ldots, K_t\}$. So a key among $K_1, K_2, \ldots, K_t$ belongs $\mathcal{R}$ and thus satifies condition (B) above. Suppose the i-th key belongs to $\mathcal{R}$, i.e. $K_i = g^i(C) \in \mathcal{R}$ with $1 \le i \le t$. Then $(K_i, g^{-t}(K_i))$ is the corresponding entry in $\mathcal{T}$. So $K'_0 = X = g^{-t}(K_i) = g^{-t+i}(C)$, and hence $C = g^j(K'_0) = K_j$ where $j = t - i$. Note that $0 \le j \le t - 1$. So a match K_j to C is found among K'_0, K'_1, $\ldots$, K'_{t-1}. Note that $E_{K'_{j-1}}(M) = g(K'_{j-1}) = K'_j = C$, so the output K'_{j-1} is the correct key that encrypts M to C. Moreover, the algorithm evaluates $g(C)$, $g^2(C)$, $\ldots$, $g^i(C)$ and $g(K'_0), g^2(K'_0), \ldots, g^{t-i}(K'_0)$, for a total of $i + (t - i) = t$ evaluations of g.

Programming Problem for CPSC 418 only

Problem 6 — Create Your Own Encryption Algorithm (38 marks + 4 bonus marks)

Working code for these problems is posted separately on the Piazza resources page `https://piazza.com/ucalgary.ca/fall2021/cpsc418math318/resources`.