COMP 2711 Discrete Mathematical Tools for Computer Science
2021 Spring Semester – Final Exam (Part 1)

Note 1: For all non-proof questions, showing your steps is not necessary unless required otherwise. However, in case your answer is wrong, showing your steps may earn you some partial credits.
Note 2: You can express your answers using binomial coefficients, factorials, and D_n (derangement number). However, you should not have summation $\sum$ in your final answers. For example, $\binom{10}{3}D_9+4!$ and $1!+2!+3!+4!$ are valid, but $\sum_{i=0}^{n}\binom{n}{i}$ or $\binom{n}{0}+\binom{n}{1}+\cdots+\binom{n}{n}$ is not. The latter has to be simplified to 2^n.

Question 1: [10 pts] You are constructing a string of n digits by the following procedure.

1. Set all the n digits to 0.
2. For every x_1-th digit, replace it by 1.
3. For every x_2-th digit, replace it by 2.
4. For every x_3-th digit, replace it by 3.

Below is an example with $n = 15$, $x_1 = 3$, $x_2 = 5$ and $x_3 = 7$.

"001021301201032"

Suppose $x_1 = 31$, $x_2 = 43$ and $x_3 = 53$. What is the minimum value n so that the string ends with "31020"?

Solution: This is equivalent to finding the smallest n that satisfies the following system of congruences.

$$\begin{cases} n \equiv 3 \pmod{31} \\ n \equiv 1 \pmod{43} \\ n \equiv 4 \pmod{53} \end{cases}$$

Let $m = 31 \cdot 43 \cdot 53 = 70649$, $M_1 = 43 \cdot 53 = 2279$, $M_2 = 31 \cdot 53 = 1643$ and $M_3 = 31 \cdot 43 = 1333$. By the extended Euclidean algorithm, we have 2 is an inverse of $M_1 \pmod{31}$, 24 is an inverse of $M_2 \pmod{43}$ and 20 is an inverse of M_3. By the Chinese Remainder Theorem, $n = 3 \cdot 2 \cdot M_1 + 1 \cdot 24 \cdot M_2 + 4 \cdot 20 \cdot M_3 = 159746 \equiv 18448 \pmod{70649}$. So, the minimum length is 18448.

Question 2: [10 pts] There are n coins, all of equal shape and size, but only $n-1$ of them are of equal weight. You are given a balance scale which can compare the weight of any two sets of coins. Assume n is a power of 3. We use the following algorithm to find the coin of different weight. Show that the number of weightings is at most $2\log_3 n$.

Procedure $findCoin(S$: a set of coins)

if $|S| = 1$:

return the only coin in S.

Partition S equally into 3 sets of coins, A, B and C.

if A and B are of equal weight:

return $findCoin(C)$

else if B and C are of equal weight:

return $findCoin(A)$

else:

return $findCoin(B)$.

Solution: In each recursive call, it takes at most 2 comparisons and the input size are reduced to one-third of the parent. So, the number of comparisons is at most $2\log_3 n = O(\log n)$.

Question 3: [10 pts] Consider a company with $n > 0$ workers. A worker is a *famous-solo-worker* if she or he is known by every other worker, but knows none of them. You are the boss and you want to check if there is a famous-solo-worker in your company. You are only allowed to ask the following question to a worker: "Do you know worker j?" The workers will answer your questions truthfully. Show that you can find the famous-solo-worker, or declare that such a worker doesn't exist, with at most $3(n-1)$ questions.

Note that there can be at most one famous-solo-worker.

Solution: We prove this by induction.

Basis step:
When $n = 1$, the only worker must be a famous-solo-worker, and no questions are needed, which equals to $3(1-1) = 0$.

Inductive step:
The inductive hypothesis is that if there are $k \geq 1$ workers, then we can determine whether there is a famous-solo-worker with at most $3(k-1)$ questions.
Suppose there are $k+1$ workers. Let i and j be any two workers among them. We ask whether i knows j, this takes one question. If i knows j, then i is not a famous-solo-worker. If i doesn't know j, then j is not a famous-solo-worker. Without loss of generality, assume that i is not a famous-solo-worker. Excluding i from the $k+1$ workers, there are k workers (including j) remaining. By the inductive hypothesis, we can use $3(k-1)$ questions for our checking.

If there is no famous-solo-worker among these k workers, then we know that there is no famous-solo-worker, and the number of questions used is $1 + 3(k-1) = 3k - 2 \leq 3k$.

If there is a famous-solo-worker s among these k workers, then we ask whether s knows i and whether i knows s. This takes two more questions. If i knows s and s doesn't know i, then s is still a famous-solo-worker, otherwise s is not a famous-solo-worker (thus no famous-solo-worker). In this case, we used a total of $1 + 3(k-1) + 2 = 3k$ questions.

Question 4: [8 pts] Suppose that $P(n)$ is a predicate. Under each of the following conditions, find the integers n such that $P(n)$ must be true:

(a) $P(0)$ is true; for all positive integers n, if $P(n)$ is true, then $P(n+2)$ is true.

(b) $P(4)$ is true; for all positive integers n, if $P(n)$ is true, then $P(2n)$ is true.

(c) $P(1)$ is true; for all positive integers n, if $P(n)$ and $P(n+1)$ are true, then $P(n+2)$ is true.

(d) $P(2)$ is true; for all positive integers n, if $P(k)$ is true for $2 \leq k \leq n$, then $P(n+1)$ is true.

Solution: (a) 0 only.

(b) For all integers n that is a power of 2 and greater than or equal to 4.

(c) 1 only.

(d) For all integers $n \geq 2$.

Grading scheme: 2 points for each.

Question 5: [6 pts] Give a recursive defintion of the set of bit strings that have more 0s than 1s.

Solution: Let S be the set of bit strings that have more 0s than 1s.

Base case: $0 \in S$.

Recursive case: If x and y are bit strings in S, then $xy \in S$, $xy1 \in S$, $x1y \in S$ and $1xy \in S$.

Question 6: [10 pts] A string is called **E-String** if the string consists of only digits in $\{0, 2, 4, 7, 8, 9\}$ and the sum of the digits is even. E.g. "079" is an E-String because $0+7+9 = 16$ is even. "708" is not an E-String because $7+0+8 = 15$ is odd. "107" is not an E-String because the digit 1 is not allowed. The empty string is also considered an E-string.

How many E-Strings of length n are there? Give a closed-form solution. [*Hint: use a recurrence.*]

Solution: Let $E(n)$ denote the number of E-Strings of length n.

Base case: $E(0) = 1$

Recursive case: For $n > 0$, $E(n) = 2 \cdot (6^{n-1} - E(n-1)) + 4 \cdot E(n-1) = 2 \cdot 6^{n-1} + 2 \cdot E(n-1)$.

$$\begin{aligned}
E(n) &= 2 \cdot 6^{n-1} + 2 \cdot E(n-1) \\
&= 2 \cdot 6^{n-1} + 2 \cdot (2 \cdot 6^{n-2} + 2 \cdot E(n-2)) \\
&= 2 \cdot 6^{n-1} + 2^2 \cdot 6^{n-2} + 2^2 \cdot E(n-2) \\
&\vdots \\
&= \sum_{i=1}^{n} 2^i \cdot 6^{n-i} + 2^n \cdot E(0) \\
&= \sum_{i=1}^{n} 2^n \cdot 3^{n-i} + 2^n \\
&= 2^n \sum_{i=0}^{n-1} 3^i + 2^n \\
&= 2^n \frac{3^n - 1}{3 - 1} + 2^n \\
&= 2^{n-1}(3^n - 1) + 2^n
\end{aligned}$$