CS201: Discrete Mathematics (Fall 2024)
Written Assignment #2 - Solutions
(100 points maximum but 110 points in total)
Deadline: 11:59pm on Oct 25 (please submit to Blackboard)
PLAGIARISM WILL BE PUNISHED SEVERELY

Q.1 (**5p**) Suppose that A, B and C are three finite sets. For each of the following, determine whether or not it is true. Explain your answers.

(a) (**1p**) $(A \cap B \neq \varnothing) \rightarrow ((A - B) \subset A)$

(b) (**1p**) $(A \subseteq B) \rightarrow (|A \cup B| \geq 2|A|)$

(c) (**3p**) $\overline{(A - B)} \cap \overline{(B - A)} = \overline{A \cup B}$

Solution:

(a) True. $A \cap B \neq \varnothing$ means that there exists at least one element in the intersection of A and B, so $A - B$ misses at least one element of A.

(b) False. Counterexample: Let $A = B = \{1\}$. Then, $A \subseteq B$ is true, but $|A \cup B| = 1 < 2 = 2|A|$.

(c) False. Counterexample: Let $A = \{1, 2\}$, $B = \{2, 3\}$ and the universe $U = \{1, 2, 3, 4\}$. Then, $\overline{(A - B)} = \{2, 3, 4\}$, $\overline{(B - A)} = \{1, 2, 4\}$, $\overline{(A - B)} \cap \overline{(B - A)} = \{2, 4\}$ and $\overline{A \cup B} = \{4\}$. Therefore, $\overline{(A - B)} \cap \overline{(B - A)} \neq \overline{A \cup B}$.

□

Q.2 (**10p**) Let A, B and C be sets. Prove the following using set identities. Please write out the names of identities used at each step (see the lecture slides for examples).

(a) (**4p**) $\overline{A \cap (B \cup C)} = (\overline{C} \cap \overline{B}) \cup \overline{A}$

(b) (**6p**) $(A - B) \cap (B - A) = \varnothing$

Solution:

(a) We have

$$\begin{aligned}
\overline{A \cap (B \cup C)} &= \overline{A} \cup \overline{(B \cup C)} && \text{De Morgan's} \\
&= \overline{A} \cup (\overline{B} \cap \overline{C}) && \text{De Morgan's} \\
&= (\overline{B} \cap \overline{C}) \cup \overline{A} && \text{Communative} \\
&= (\overline{C} \cap \overline{B}) \cup \overline{A} && \text{Communative}
\end{aligned}$$

(b) We have

$$\begin{aligned}
(A - B) \cap (B - A) &= (A \cap \overline{B}) \cap (B \cap \overline{A}) && \text{Definition} \\
&= ((A \cap \overline{B}) \cap B) \cap \overline{A} && \text{Associative} \\
&= (A \cap (\overline{B} \cap B)) \cap \overline{A} && \text{Associative} \\
&= (A \cap \varnothing) \cap \overline{A} && \text{Complement} \\
&= \varnothing \cap \overline{A} && \text{Domination} \\
&= \varnothing && \text{Domination}
\end{aligned}$$

□

Q.3 (**10p**) Prove the following statements:

(a) (**5p**) Any subset of a countable set A is still countable. (Note that the contrapositive statement is also useful: If A is uncountable, then any set having A as a subset is also uncountable.)

(b) (**5p**) If A is a countable and there is an onto function from A to B, then B is also countable.

Solution:

(a) If a set A is countable, then we can list its elements as $a_1, a_2, \ldots, a_n, \ldots$ (possibly ending after a finite number of steps). Every subset of A consists of some (or none or all) of the items in this sequence, so we can list them in the same order in which they appear in the above sequence. This gives us a sequence listing all the elements of the subset. Therefore, the subset of A is still countable.

(b) First, it is easy to see that the statement is true if set A is finite, because the cardinality of B can only be smaller than or equal to that of A. Then, if set A is infinite, then there exists a bijection f from $\mathbf{Z}^+$ to A. Let g denote the onto function from A to B, then the composition $g \circ f$ can list all elements of B, but some elements in B may be listed more than once. Removing those duplicated elements yields a listing of all elements in B, i.e., B is countable.

□

Q.4 (**15p**) The *symmetric difference* of A and B, denoted by $A \Delta B$, is the set containing those elements in either A or B (not in both A and B).

(a) (**5p**) Determine whether the symmetric difference is associative; that is, if A, B and C are sets, does it follow that $A \Delta (B \Delta C) = (A \Delta B) \Delta C$? Explain your answer.

(b) (**5p**) Suppose that A, B, C are sets such that $A \Delta C = B \Delta C$. Prove or disprove $A = B$.

(c) (**5p**) Give an example of two uncountable sets A and B such that $A \Delta B$ is infinite and countable. (Hint: we learned from class that $[0, 1]$ is uncountable.)

Solution:

(a) Using a membership table, one can show that each side consists of the elements that are in an odd number of the sets A, B and C. (The table is required but omitted here.)

(b) Yes. We prove that for every element $x \in A$, we have $x \in B$ and vice versa.

We prove $A \subseteq B$ by considering two cases: "$x \in A$ and $x \notin C$" and "$x \in A$ and $x \in C$". First, for elements $x \in A$ and $x \notin C$, since $A \Delta C = B \Delta C$, we know that $x \in A \Delta C$ and hence $x \in B \Delta C$. Since $x \notin C$, we must have $x \in B$. Then, for elements $x \in A$ and $x \in C$, we have $x \notin A \Delta C$ and hence $x \notin B \Delta C$. Since $x \in C$, we must have $x \in B$.

Similarly, we have $B \subseteq A$, and hence $A = B$ holds.

(c) Let $A = [0, 1]$, $B = [0, 1] \cup \mathbb{N}$. We know that A is uncountable and from Q.3 (a) we have B is also uncountable. Then, $A \Delta B = \mathbb{N} \setminus \{0, 1\}$ is obviously infinite and countable.

□

Q.5 (**5p**) For finite sets A, B and C, explain why the following inclusion-exclusion formula is true:

$$|A \cup B \cup C| \;=\; |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C|.$$

Solution: To count the number of elements in $A \cup B \cup C$, we proceed as follows. First we count the elements in each of the sets and add. This certainly gives us all the elements in the union, but we have overcounted. Each element in $A \cap B$, $A \cap C$, and $B \cap C$ has been counted twice. Therefore we subtract the cardinalities of these intersections to make up for the overcount. Finally, we have compensated a bit too much, since the elements of $A \cap B \cap C$ have now been counted three times and subtracted three times. We adjust by adding back the cardinality of $A \cap B \cap C$. □

Q.6 (**5p**) Show that if A, B, C and D are (probably infinite) sets with $|A| = |B|$ and $|C| = |D|$, then $|A \times C| = |B \times D|$.

Solution: By definition, there exist bijective functions $f : A \to B$ and $g : C \to D$. Then, clearly the function from $A \times C$ to $B \times D$ that maps (a, c) to $(f(a), g(c))$ is also a bijection. By definition, this shows $|A \times C| = |B \times D|$. □

Q.7 (**10p**) Consider two functions $g : A \to B$ and $f : B \to C$ and its composition function $f \circ g$. Answer the following questions and explain.

(a) (**2p**) If $f \circ g$ is one-to-one and g is one-to-one, must f be one-to-one?

(b) (**2p**) If $f \circ g$ is one-to-one and f is one-to-one, must g be one-to-one?

(c) (**2p**) If $f \circ g$ is one-to-one, must g be one-to-one?

(d) (**2p**) If $f \circ g$ is onto, must f be onto?

(e) (**2p**) If $f \circ g$ is onto, must g be onto?

Solution:

(a) No. We prove this by giving a counterexample. Let $A = \{1, 2\}$, $B = \{a, b, c\}$, and $C = A$. Define the function g by $g(1) = a$ and $g(2) = b$, and define the function f by $f(a) = 1$, and $f(b) = f(c) = 2$. Then it is easily verified that $f \circ g$ is one-to-one and g is one-to-one. But f is not one-to-one.

(b) Yes. For any two elements $x, y \in A$ with $x \neq y$, assume to the contrary that $g(x) = g(y)$. On one hand, since $f \circ g$ is one-to-one, we have $f \circ g(x) \neq f \circ g(y)$. One the other hand, $f \circ g(x) = f(g(x)) = f(g(y)) = f \circ g(y)$. This leads to a contradiction. Thus, $g(x) \neq g(y)$, which means that g must be one-to-one.

(c) Yes. The proof is the same as (b), as the condition "f is one-to-one" is actually not used.

(d) Yes. Since $f \circ g$ is onto, we know that $f \circ g(A) = C$, which means that $f(g(A)) = C$. Note that $g(A)$ is a subset of B, thus, $f(B)$ must also be C. This means that f is also onto.

(e) No. A counterexample is the same as that in (a).

□

Q.8 (**5p**) Let x be a real number. Prove that $\lfloor 3x \rfloor = \lfloor x \rfloor + \lfloor x + \frac{1}{3} \rfloor + \lfloor x + \frac{2}{3} \rfloor$.

Solution: Note that every real number x lies in an interval $[n, n+1)$ for an integer $n = \lfloor x \rfloor$. We can prove the statement by considering three cases below.

- Case 1: $x \in [n, n+\frac{1}{3})$. We have $3x$ lies in the interval $[3n, 3n+1)$, so $\lfloor 3x \rfloor = 3n$. Moreover, in this case both $x+\frac{1}{3}$ and $x+\frac{2}{3}$ are still less than $n+1$, so the right side $\lfloor x \rfloor + \lfloor x+\frac{1}{3} \rfloor + \lfloor x+\frac{2}{3} \rfloor = n+n+n = 3n$, which equals the left side.

- Case 2: $x \in [n+\frac{1}{3}, n+\frac{2}{3})$. We have $3x \in [3n+1, 3n+2)$, so $\lfloor 3x \rfloor = 3n+1$. Moreover, in this case $x+\frac{1}{3}$ is in $[n+\frac{2}{3}, n+1)$, and $x+\frac{2}{3}$ is in $[n+1, n+\frac{4}{3})$, so $\lfloor x \rfloor + \lfloor x+\frac{1}{3} \rfloor + \lfloor x+\frac{2}{3} \rfloor = n+n+(n+1) = 3n+1$, which is the same as the left side.

- Case 3: $x \in [n+\frac{2}{3}, n+1)$. The proof is similar and both sides equal $3n+2$.

□

Q.9 (**10p**) Derive the *closed formula* for $\sum_{k=0}^{m} \lfloor \sqrt{k} \rfloor$, where m is a positive integer. (Hint: express the summation as a function $f(m, n)$, where $n = \lfloor \sqrt{m} \rfloor - 1$.)

Solution: First, note that $\sum_{k=0}^{m} \lfloor \sqrt{k} \rfloor = \sum_{k=0}^{(n+1)^2-1} \lfloor \sqrt{k} \rfloor + \sum_{k=(n+1)^2}^{m} \lfloor \sqrt{k} \rfloor$, where $n = \lfloor \sqrt{m} \rfloor - 1$.

It is not hard to see that there are $(2i+1)$ is in $\sum_{k=0}^{(n+1)^2-1} \lfloor \sqrt{k} \rfloor$. That is, $\sum_{k=0}^{(n+1)^2-1} \lfloor \sqrt{k} \rfloor = (1+1+1) + \cdots + (i+\cdots+i) + \cdots + (n+\cdots+n) = 3 \cdot 1 + \cdots + (2i+1) \cdot i + \cdots + (2n+1) \cdot n$.

Clearly, the second summation $\sum_{k=(n+1)^2}^{m} \lfloor \sqrt{k} \rfloor$ contains $(m - (n+1)^2 + 1)$ $(n+1)$s, i.e., $\sum_{k=(n+1)^2}^{m} \lfloor \sqrt{k} \rfloor = (m - (n+1)^2 + 1)(n+1)$.

Therefore, we have:

$$\begin{aligned}
\sum_{k=0}^{m} \lfloor \sqrt{k} \rfloor &= \sum_{i=1}^{n} (2i^2 + i) + (n+1)(m - (n+1)^2 + 1) \\
&= 2\sum_{i=1}^{n} i^2 + \sum_{i=1}^{n} i + (n+1)(m - (n+1)^2 + 1) \\
&= \frac{n(n+1)(2n+1)}{3} + \frac{n(n+1)}{2} + (n+1)(m - (n+1)^2 + 1)
\end{aligned}$$

□

Q.10 (**5p**) Apply the Schröder-Bernstein theorem to prove $(0, 1)$ and $[0, 2]$ have the same cardinality.

Solution: By the Schröder-Bernstein theorem, it suffices to find one-to-one functions $f : (0, 1) \to [0, 2]$ and $g : [0, 2] \to (0, 1)$. Let $f(x) = x$ and $g(x) = (x+1)/4$. It is then straightforward to prove that f and g are both one-to-one.

□

Q.11 (**5p**) Show that when the Hilbert's Grand Hotel (see lecture slides "03 Sets and Functions") is fully occupied one can still accommodate countably infinite new guests in it.

Solution: First, fix a list that counts all new guests: $n_1, n_2, \ldots$ and let e_i denote the existing guest in room i. Then, alternately accommodate new guests and existing guests, i.e., put guest n_i in room $2i-1$ and put guest e_i in room $2i$. This way one can accommodate countably infinite new guests in a fully occupied Hilbert's Grand Hotel.

□

Q.12 (**10p**) In order to show that there exist uncomputable functions, it suffices to prove the following two parts:

(a) (**5p**) The set of all computer programs in all existing programming languages is countable.

(b) (**5p**) The set of all functions from $\mathbf{Z}^+$ to the set of digits $\{0, 1, \ldots, 9\}$ is uncountable.

Then, one can conclude that there exists a function $f^* : \mathbf{Z}^+ \to \{0, 1, \ldots, 9\}$ that is uncomputable, i.e., no computer program in any programming language can find the values of this function f^*. Prove the above two statements from scratch, i.e., do not use proved theorems taught in class. (Hint: refer to the proof of the theorem "the set of all Java programs are uncountable" for (a) and refer to the proof of the theorem "the set of real numbers is uncountable" for (b).)

Solution:

(a) First, note that any computer program in any particular programming language can be viewed as a finite string of symbols from a finite alphabet. Similar to the proof of showing the set of all Java programs is countable, one can prove that "the set of all computer programs in any particular programming language is countable". (The proof is omitted here for similarity, but to get points you should write the formal procedures.) Then, since the number of existing programming languages is finite, one can count all computer programs in all existing programming languages as follows: for each language, fix a list that counts all programs in this particular language; in the first round, count the first programs in all those (finite many) lists; next, in the second round, count the second programs in all those lists; etc.

(b) Note that any function $f : \mathbf{Z}^+ \to \{0, 1, \ldots, 9\}$ can be viewed as an infinite string $a_1 a_2 \ldots a_n \ldots$ where $a_i \in \{0, 1, \ldots, 9\}$, because it defines f as: $f(1) = a_1, f(2) = a_2, \ldots, f(n) = a_n, \ldots$. Then, one can prove the set of all such functions is uncountable in the same way as proving "the set of real numbers is uncountable" with Cantor's diagonization argument. (The proof is omitted here for similarity, but to get points you should write the formal procedures).

□

Q.13 (**5p**) Prove that for any $a > 1$, $\Theta(\log_a n) = \Theta(\log_2 n)$. (This means the base of logarithm does not matter for measuring the complexity so we can simply write $\Theta(\log n)$.)

Solution: By definition, it suffices to show that there exist positive constants C_1, C_2 and k_0 such that $\log_a n \leq C_1 \log_2 n$ and $\log_2 n \leq C_2 \log_a n$ for all $n > k_0$. By the change of bases formula we have

$$\log_a n = \frac{\log_2 n}{\log_2 a}.$$

Now, note that $\log_2 a > 0$ for $a > 1$. We can choose $C_1 = \frac{1}{\log_2 a}$, $C_2 = \log_2 a$, $k_0 = 1$. □

Q.14 (**10p**) Consider Algorithm 1 for the *binary search* algorithm, which searches an integer x in any increasingly ordered sequence of n distinct integers $a_1, a_2, \ldots, a_n$.
Answer the following questions:

(a) (**3p**) What is the time complexity of Algorithm 1 in terms of n? Explain your answer. (Just need to count the number of comparison operations and express it with the $\Theta(\cdot)$ notation.)

(b) (**3p**) Improve Algorithm 1 such that its best-case time complexity is $\Theta(1)$ and its worst-case time complexity is unchanged (when measured with $\Theta(\cdot)$). Explain your answer.

(c) (**2p**) What is the space complexity of Algorithm 1? Explain. (Use the $\Theta(\cdot)$ notation.)

Algorithm 1 Binary Search (x: target integer, $a_1, a_2, \ldots, a_n$: increasingly ordered integers)

1: $i := 1$
2: $j := n$
3: **while** $i < j$ **do**
4: $\quad m := \lfloor (i + j)/2 \rfloor$
5: $\quad$ **if** $x > a_m$ **then** $i := m + 1$
6: $\quad$ **else** $j := m$
7: **if** $x = a_i$ **then** *location* $:= i$
8: **else** *location* $:= 0$
9: **return** *location* {*location* is the subscript of a_i such that $a_i = x$, or 0 if no such a_i is found}

(d) (**2p**) Considering binary representation of integers on computers, what is the input size of the above binary search problem using fixed-length encoding?

Solution:

(a) Time complexity: $\Theta(\log_2 n)$ (or simply $\Theta(\log n)$). The key observation is as follows: in the beginning the "distance" between i, j equals $n - 1$ (i.e., $i = 1, j = n$), then the while loop terminates only if i "meets" j (i.e., $i \geq j$), and in each iteration the distance between i, j is halved (i.e., line $3 \sim 6$).

(b) In the best case, x equals the first element in the sequence, i.e., $x = a_{\lfloor (n+1)/2 \rfloor}$. We only need to terminate the while loop and returns m when this happens. Therefore, one can just add such a check-then-break clause to the while loop. For instance, we can add "if $x = a_m$ then return m" between line 4 and line 5. It is easy to see that this new algorithm has best-case time complexity $\Theta(1)$ and worst-case time complexity $\Theta(\log n)$.

(c) It is not hard to see that the storage of the n integers $a_1, a_2, \ldots, a_n$ dominates the space complexity, which takes $\Theta(n)$ memory.

(d) Each input has the same fixed length $m = \lceil \log_2(\max\{|x|, |a_1|, \ldots, |a_n|\} + 1) \rceil + 1$. The input size is $(n + 1)m$.

□