CS 320L – Applied Discrete Mathematics – Spring 2018
Instructor: Marc Pomplun

Assignment #1 – Sample Solutions

Question 1: Hair Splitting with Set Expressions

Let us define the *successor* of the set A to be the set A ∪ {A}. Find the successors of the following sets:

a) A = {x}
A ∪ {A} = {x, {x}}

b) B = {x, y}
B ∪ {B} = {x, y, {x, y}}

c) C = Ø
C ∪ {C} = { Ø }

d) D = { Ø, { Ø }}
D ∪ {D} = { Ø, { Ø }, {Ø, { Ø }}}

Question 2: Tautologies and Contradictions

Find out for each of the following propositions whether it is a tautology, a contradiction, or neither (a contingency). Prove your answer.

a) [(p → q) ∧ (¬p → r)] → (q ∨ r)

p	**q**	**r**	**p → q**	**¬p → r**	**(p → q) ∧ (¬p → r)**	**q ∨ r**	**[(p → q) ∧ (¬p → r)] → (q ∨ r)**
T	T	T	T	T	T	T	T
T	T	F	T	T	T	T	T
T	F	T	F	T	F	T	T
T	F	F	F	T	F	F	T
F	T	T	T	T	T	T	T
F	T	F	T	F	F	T	T
F	F	T	T	T	T	T	T
F	F	F	T	F	F	F	T

It is a tautology!

b) $(p \wedge q \wedge \neg r) \rightarrow [(q \rightarrow r) \vee (p \rightarrow r)]$

p	**q**	**r**	$\mathbf{p \wedge q \wedge \neg r}$	$\mathbf{q \rightarrow r}$	$\mathbf{p \rightarrow r}$	$\mathbf{(q \rightarrow r) \vee (p \rightarrow r)}$	$\mathbf{(p \wedge q \wedge \neg r) \rightarrow [(q \rightarrow r) \vee (p \rightarrow r)]}$
T	T	T	F	T	T	T	T
T	T	F	T	F	F	F	F
T	F	T	F	T	T	T	T
T	F	F	F	T	F	T	T
F	T	T	F	T	T	T	T
F	T	F	F	F	T	T	T
F	F	T	F	T	T	T	T
F	F	F	F	T	T	T	T

This one is a contingency!

c) $(p \rightarrow q) \rightarrow (q \rightarrow p)$

p	**Q**	$\mathbf{p \rightarrow q}$	$\mathbf{q \rightarrow p}$	$\mathbf{(p \rightarrow q) \rightarrow (q \rightarrow p)}$
T	T	T	T	T
T	F	T	F	F
F	T	F	T	T
F	F	T	T	T

And here is another contingency!

Question 3: Set Operations

Let us take a look at the sets A = {x, y, z}, B = {1, 2}, C = {y, z}. List the elements of the following sets D, E, F, G, H, and I:

a) $D = (B \times A) - (B \times C) = \{(1, x), (2, x)\}$

b) $E = 2^A - 2^C = \{\{x\}, \{x, y\}, \{x, z\}, \{x, y, z\}\}$

c) $F = 2^{(2^B)} = 2^{\{\varnothing, \{1\}, \{2\}, \{1, 2\}\}} = \{\varnothing, \{\varnothing\}, \{\{1\}\}, \{\{2\}\}, \{\{1, 2\}\}, \{\varnothing, \{1\}\}, \{\varnothing, \{2\}\}, \{\varnothing, \{1, 2\}\}, \{\{1\}, \{2\}\}, \{\{1\}, \{1, 2\}\}, \{\{2\}, \{1, 2\}\}, \{\varnothing, \{1\}, \{2\}\}, \{\varnothing, \{1\}, \{1, 2\}\}, \{\varnothing, \{2\}, \{1, 2\}\}, \{\{1\}, \{2\}, \{1, 2\}\}, \{\varnothing, \{1\}, \{2\}, \{1, 2\}\}\}$

d) $G = (A \times B \times C) \cap (C \times B \times A) = \{(y, 1, y), (y, 1, z), (z, 1, y), (z, 1, z), (y, 2, y), (y, 2, z), (z, 2, y), (z, 2, z)\}$

e) $H = \{(a, b, c) \mid a, b, c \in B \wedge b \neq c \wedge a = b\} = \{(1, 1, 2), (2, 2, 1)\}$

f) $I = \{(a, b, c) \mid a \in A \wedge b \in B \wedge c \in C \wedge a = c\} = \{(y, 1, y), (z, 1, z), (y, 2, y), (z, 2, z)\}$

Question 4: Cardinality

Are the following statements true for all sets A, B and C? Prove your answers.

a) $|A \cup B \cup C| = |A - B - C|$

No. Counterexample: A = {1}, B = {1}, C = {1} gives us 1 = 0

b) $|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C|$

To prove this, let us consider a membership table as discussed in class. Any element in our universe of discourse is either in set A or not in set A, either in set B or not in set B, and either in set C or not in set C. Consequently, as shown in the table, this creates 8 types of set memberships that every element falls into. If it turns out that each type of element increases the cardinalities on the left and right sides of the equation by the same amount, it means that the equation is always true because it is impossible to create an inequality, regardless of the sets A, B, and C that we choose:

A	**B**	**C**	**$\|A \cup B \cup C\|$**	**$\|A\| + \|B\| + \|C\|$**	**$\|A \cap B\|$**	**$\|A \cap C\|$**	**$\|B \cap C\|$**	**$\|A \cap B \cap C\|$**	**Right side total**
0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	0	0	0	0	1
0	1	0	1	1	0	0	0	0	1
0	1	1	1	2	0	0	1	0	1
1	0	0	1	1	0	0	0	0	1
1	0	1	1	2	0	1	0	0	1
1	1	0	1	2	1	0	0	0	1
1	1	1	1	3	1	1	1	1	1

Clearly, the left and right sides of the equation are effected identically by each element, and therefore they must always be equal.

Question 5: Functions

Find out whether the following functions from **R** to **R** are injective, surjective, and/or bijective (no proof necessary).

a) $f(z) = -2z$

Injective, surjective, and bijective.

b) $f(z) = 3z^5 + 4$

Injective, surjective, and bijective.

c) $f(z) = z \cdot \cos z$

Not injective: For example, $f(0.5\pi) = f(1.5\pi) = 0$

Surjective, because with growing z we can reach any value for f(z).

Not bijective.

d) $f(z) = z^2/(z^2 + 1)$

Not injective. For example, $f(1) = f(-1) = 0.5$

Not surjective. For example, there is no z such that $f(z) = -1$

Question 6: Big-O Estimates

Give as good a big-O estimate as possible for the following complexity functions:

a) $f(n) = (n^2 \cdot \log n)\ (n + 3n^2) = n^3 \cdot \log n + 3n^4 \cdot \log n$

The term with greater exponent will dominate, therefore it is $O(n^4 \cdot \log n)$.

b) $f(n) = (3n! + 4n^2) + (2^n \cdot n^2)$

The factorial term will dominate, so it is $O(n!)$.

c) $f(n) = n^5 + 2 \log n + n^3\ (n^3 + 2n) = n^5 + 2 \log n + n^6 + 2n^4$

Again, the greatest exponent decides: $O(n^6)$.

Question 7: Algorithms and Their Complexity

a) Write a simple program in pseudocode (or in Python, C, C++, or Java, but only use basic commands so that comparisons can be counted) that receives a sequence of positive integers $a_1, \ldots, a_n$ as its input and determines if the sequence contains three distinct terms x, y, and z such that $x^2 + y^2 = z^2$. Once it finds such terms, it prints them and terminates; it does not continue searching after the first find. If the program does not find any such terms, it prints out that it was unsuccessful and then terminates.

Example program with test in plain C:

```
#include <stdio.h>
```

```
void findTriple(int seq[], int n)
{
    int x, y, z;

    for (x = 0; x < n; x++)
        for (y = 0; y < n; y++)
            for (z = 0; z < n; z++)
            if (seq[x]*seq[x] + seq[y]*seq[y] == seq[z]*seq[z])
            {
               printf("Awesome! I found %d, %d, and %d!\n", seq[x],
                      seq[y], seq[z]);
               return;
            }
    printf("What the heck? I didn't find any such triple!\n");
}

int main()
{
    int testSeq[10] = { 8, 3, 4, 2, 10, 9, 7, 20, 5, 7 };

    findSquarePair(testSeq, 10);
    return 0;
}

Awesome! I found 3, 4, and 5!
```

b) Describe the kind of input that causes worst-case time complexity for your algorithm (only count comparisons), and explain why this is the case.

The most comparisons for a given sequence length are necessary if there is no $x^2 + y^2 = z^2$ match among its elements at all. In that case, in each iteration of the loop one comparison (plus the ones in the “for” construct) have to be performed.

c) Provide an equation for your algorithm that describes the number of required comparisons as a function of input length n in the worst case. For some algorithms, it may be a good idea to first use a sum notation, but at the end you should provide a closed-form equation, i.e., one that no longer uses the sum symbol but only operations such as multiplication or addition of individual numbers or variables.

If you count the comparisons in the “for” constructs (and if you did not, you will not lose points), then for a sequence of n elements, the outermost loop (the x-loop) will cause (n + 1) comparisons in its “for” statement because of n loop iterations and the decision to end the loop when it is reached for the (n + 1)-th time. In addition, of course, the x-loop includes n executions of the y-loop, which each time causes (n + 1) comparisons and calls the z-loop n times. The z-loop also involves (n + 1) comparisons in the “for” statement and n comparisons in the “if” statement. In total this gives us f(n) comparisons:

$$f(n) = n + 1 + \sum_{x=1}^{n} \left[n + 1 + \sum_{y=1}^{n} \left(n + 1 + \sum_{z=1}^{n} 1 \right) \right]$$

$$\Leftrightarrow f(n) = n + 1 + \sum_{x=1}^{n}\left[n + 1 + \sum_{y=1}^{n}(2n + 1)\right]$$

$$\Leftrightarrow f(n) = n + 1 + \sum_{x=1}^{n}[n + 1 + 2n^2 + n]$$

$$\Leftrightarrow f(n) = n + 1 + \sum_{x=1}^{n}[2n^2 + 2n + 1]$$

$$\Leftrightarrow f(n) = n + 1 + 2n^3 + 2n^2 + n$$

$$\Leftrightarrow f(n) = 2n^3 + 2n^2 + 2n + 1$$

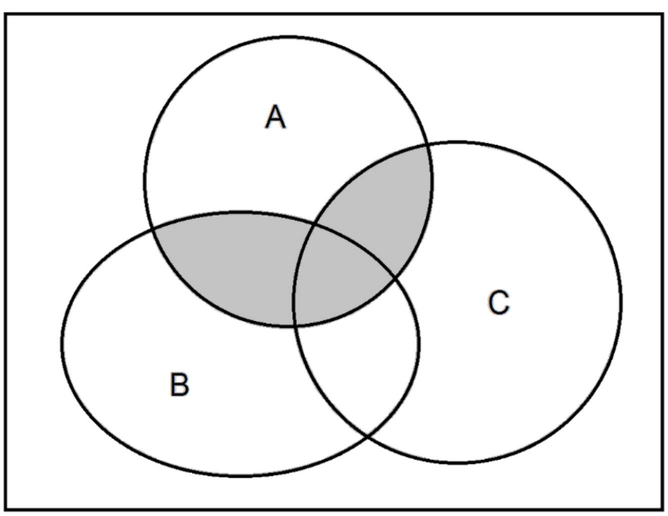
d) Use the big-O-notation to describe the worst-case time complexity of your algorithm.

$O(n^3)$

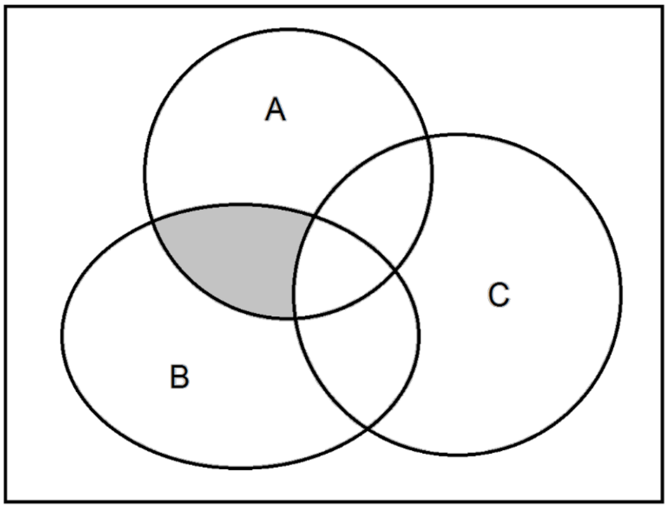
Question 8 (Bonus Question): Venn Diagrams

Draw the Venn diagrams for the following sets:

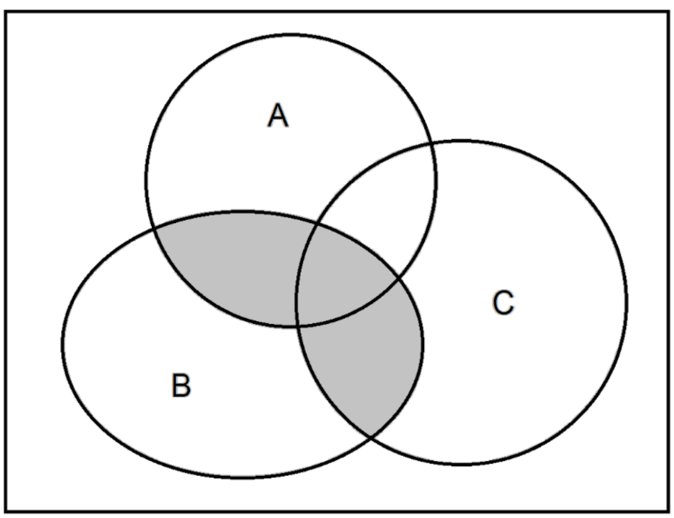
a) $A \cap (B \cup C)$



b) $A \cap (B - C)$



c) $(A \cap B) \cup (B \cap C)$



d) $(A \cap \overline{B}) \cup (B \cap \overline{C})$