Problem Sheet #3

Aayush Sharma Acharya

October 3, 2018

Problem 3.1: *Distributive laws for sets*

Let A, B and C be sets. Proof that the following two distributive laws hold:

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

and,

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

Solution,
Let P and Q be any two sets:
Now,

$$P \cup Q := \{x : x \in P \; or \; x \in Q\}$$

And,

$$P \cap Q := \{x : x \in P \; and \; x \in Q\}$$

Let T be the event when an element x is present in the set.

and

F be the event when an element x is NOT present in the set.
Recalling the definition of $P \cap Q$ and $P \cup Q$, we can construct a table in order to prove the relation:

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

A	B	C	$A \cup B$	$A \cup C$	$(\boldsymbol{A} \cup \boldsymbol{B}) \cap (\boldsymbol{A} \cup \boldsymbol{C})$	$B \cap C$	$\boldsymbol{A} \cup (\boldsymbol{B} \cap \boldsymbol{C})$
T	T	T	T	T	**T**	T	**T**
T	T	F	T	T	**T**	F	**T**
T	F	T	T	T	**T**	F	**T**
T	F	F	T	T	**T**	F	**T**
F	T	T	T	T	**T**	T	**T**
F	T	F	T	F	**F**	F	**F**
F	F	T	F	T	**F**	F	**F**
F	F	F	F	F	**F**	F	**F**

Here in this table we have generated a full list of possible events when x is present and not present in both set A and B and we have found that the events in $(A \cup B) \cap (A \cup C)$ and $A \cup (B \cap C)$ are equivalent. This says that the distributive laws hold for the following:

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

Using the same method we can construct another table for the following:

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

A	B	C	$A \cap B$	$A \cap C$	$(\boldsymbol{A} \cap \boldsymbol{B}) \cup (\boldsymbol{A} \cap \boldsymbol{C})$	$B \cup C$	$\boldsymbol{A} \cap (\boldsymbol{B} \cup \boldsymbol{C})$
T	T	T	T	T	**T**	T	**T**
T	T	F	T	F	**T**	T	**T**
T	F	T	F	T	**T**	T	**T**
T	F	F	F	F	**F**	F	**F**
F	T	T	F	F	**F**	T	**F**
F	T	F	F	F	**F**	T	**F**
F	F	T	F	F	**F**	T	**F**
F	F	F	F	F	**F**	F	**F**

In this table we have generated a full list of possible events when x is present and not present in both set A and B and we have found that the events in $(A \cap B) \cup (A \cap C)$ and $A \cap (B \cup C)$ are equivalent. This, also, says that the distributive laws hold for the following:

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

Problem 3.2: *reflexive, symmetric and transitive*
For each of the following relations, determine whether they are reflexive, symmetric, or transitive.Provide a reasoning.

a) $\mathbb{R} = \{(a, b) | a, b \in \mathbb{Z} \wedge a \neq b\}$

$a \neq a$ is not true hence it is not reflexive.
If $a \neq b$ is true then $b \neq a$ is also true. Hence it is symmetric.
If $a \neq b$ is true and assume $b \neq c$ is true, then $a \neq c$ may or may not be true. So it is not transitive.

b) $\mathbb{R} = \{(a, b) | a, b \in \mathbb{Z} \wedge |a - b| \leq 3\}$

$|a - a| \leq 3$ and $|b - b| \leq 3$ is true, hence it is reflexive.
If $|a - b| \leq 3$ is true, then $|b - a| \leq 3$ is also true. So it is also symmetric.
If $|a - b| \leq 3$ and lets assume that $|b - c| \leq 3$, then $|a - c| \leq 3$ may be greater than or less than 3 or sometimes equals to 3. Hence it is not transitive.

c) $\mathbb{R} = \{(a, b) | a, b \in \mathbb{Z} \wedge (a \; mod \; 10) = (b \; mod \; 10)\}$

$(a \; mod \; 10) = (a \; mod \; 10)$ is always true and $(b \; mod \; 10) = (b \; mod \; 10)$ is also always true. Therefore, it is reflexive.
If $(a \; mod \; 10) = (b \; mod \; 10)$ is True, then $(b \; mod \; 10) = (a \; mod \; 10)$ is also True. Hence, it is also symmetric.
It also transitive because if $(a \; mod \; 10) = (b \; mod \; 10)$ and $(b \; mod \; 10) = (c \; mod \; 10)$ then, the relation $(a \; mod \; 10) = (c \; mod \; 10)$ is also true.

Problem 3.3: *Circular prime numbers (haskell)*

Source code is also included in the zip file.

a) Implement a function **prime :: Integer -> Bool** that returns a Bool value indicating whether the Integer argument is a prime number or not.

```
prime :: Integer -> Bool
prime n = length [x | x <- [1..n], n ‘mod‘ x == 0] == 2
```

This function takes a number n and finds its remainder by dividing it from *1 to n*. For example $n = 92$, It would look like:

```
91 ‘mod‘ 1  = 0
91 ‘mod‘ 2  = 1
91 ‘mod‘ 3  = 1
91 ‘mod‘ 4  = 3
.   .    .    .
.   .    .    .
.   .    .    .
91 ‘mod‘ 91 = 0
```

This would generate all the possible factors of n. Then it would filter the list of numbers which has only 2 factors (which is why I used the length function). The number which has only 2 factors (1 and number itself) is called the prime number.
We can test the validity of our function by:

```
     > prime 2
     True
    > filter prime [2..100]
[2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61,67,71,73,79,83,89,97]
```

b) Using the prime function, implement a function **circprime :: Integer ->Bool** that returns a **Bool** value indicating whether the Integer value is a circular prime number or not.

```
circprime :: Integer -> Bool
circprime n =
        length [z | z <- [prime y |
                    y <- [read x :: Integer |
                    x <- circle (show n)]],
            z == True]
            == length (circle (show n))
```

First of all the function **circprime** accepts a number ***n*** and then converts it to string using **show** function. After it is converted it is then passed to **circle** function which I had written from the previous assignment.The circle function provides me with all the cyclic orders of the string. After all the orders of the string are returned in an array, I convert each combination string in the array to numbers and pass it to function **prime** which I had written earlier. It returns a list of True and False which denotes that combination of number is a prime number or not. If the list contains True values that is equal to the length of **(circle (show *n*))**, that prime number is circular.