Adina Hessen - 336165139
Ora Wetzler - 212058689

Computational biology - Final project - Part 1

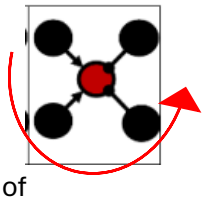
In this part, we were to write a program that finds all the monotonic regulation conditions of the reasoning engine. Specifically, we needed to show that there are only 18 such functions.

The first part was to create as input all nine given conditions of the activators and depressors:

```
# All the possible values of activators\depressors as given in the project
conditions = {
    1: (0, 0, 0, 0),
    2: (1, 0, 0, 0),
    3: (1, 1, 0, 0),
    4: (0, 0, 1, 0),
    5: (1, 0, 1, 0),
    6: (1, 1, 1, 0),
    7: (0, 0, 1, 1),
    8: (1, 0, 1, 1),
    9: (1, 1, 1, 1)
}

# According to the conditions, match condition number to number of active activators\depressors
num_activators_on = {1: 0, 2: 1, 3: 2, 4: 0, 5: 1, 6: 2, 7: 0, 8: 1, 9: 2}
num_depressors_on = {1: 0, 2: 0, 3: 0, 4: 1, 5: 1, 6: 1, 7: 2, 8: 2, 9: 2}
```

These conditions were written as a dictionary, where every condition has a name. The state of every condition was given as a tuple of four bits, where an inactive activator/depressor is 0, and an active one is 1. The tuple bits correspond to the activators and depressors starting from the upper left activator and goes clockwise. As a conclusion, the first two bits represent activators, and the last two represent depressors.



As a straight conclusion, num_activaors_on represents for every condition number the number of active activators, and similarly for num_depressors_on represents the number of active depressors.

```
from itertools import product
from tabulate import tabulate
from colorama import Fore, Style, init
```

Here we import the function "product" for creating all functions, and the rest are for printing the output table.

Adina Hessen - 336165139
Ora Wetzler - 212058689

```python
def main():
    all_functions = get_all_possible_functions()
    monotonic_functions = find_monotonic_functions(all_functions)
    print_output_table(monotonic_functions)


if __name__ == '__main__':
    main()
```

Our project consists of three parts:

1. find all possible Boolean functions of length 9
2. find all monotonic functions
3. print the output table

```python
# Get all possible functions with #conditions --> 2^#conditions possible functions
def get_all_possible_functions():
    return list(product([0, 1], repeat=len(conditions)))
```

Returns a tuple of length 9 with all possible functions

```python
# Out of all functions, find the monotonic ones
def find_monotonic_functions(all_functions):
    monotonic_functions = []
    for function in all_functions:
        if is_monotonic(function):
            monotonic_functions.append(function)
    return monotonic_functions
```

For every function, check if it monotonic – if so, add to monotonic_functions array

Adina Hessen - 336165139
Ora Wetzler - 212058689

```python
# Check if received function is monotonic
def is_monotonic(function):
    _,_,all_activators,_,_,_,all_depressors,_,_ = function
    if not all_activators or all_depressors:
        return False

    # Go over function activity
    for i in conditions.keys():
        if function[i - 1]:
            # If gene in this condition is active, check monotonic attributes
            for j in conditions.keys():
                if num_activators_on[i] < num_activators_on[j] and num_depressors_on[i] == num_depressors_on[j]:
                    if not function[j - 1]:
                        # Gene with more activators and not active - non-monotonic
                        return False
                if num_depressors_on[i] > num_depressors_on[j] and num_activators_on[i] == num_activators_on[j]:
                    if not function[j - 1]:
                        # Gene with fewer depressors and not active - non-monotonic
                        return False
    return True
```

In this function, we check for every tuple of length 9 that represents a Boolean function, if it's monotonic. As we learned in class, a monotonic function maintains the following conditions:

1) If for some condition the gene is active, then adding another activator should still have the gene activated.
2) If for some condition the gene is activated, decreasing the number of depressors should keep the gene active.
3) If all activators are on, and no depressors are on (that is the 3rd condition in the table), the gene must be active
4) If all depressors are on, and no activators are on, the gene must not be active

This function checks these conditions through negation.

```python
# Print table of monotonic functions
def print_output_table(monotonic_functions):
    init(autoreset=True)
    # Prepare table headers
    headers = ['Function #'] + [str(cond) for cond in conditions.values()]

    # Prepare table data
    table_data = []
    for func_count, function in enumerate(monotonic_functions):
        # Unpack function to single bits, and match to correct cell
        func = {}
        func[1], func[2], func[3], func[4], func[5], func[6], func[7], func[8], func[9] = function
        row = [func_count]

        for index in conditions.keys():
            # Color active gene in ren, non-active gene in green
            color = Fore.RED if func[index] else Fore.GREEN
            row.append(f"{color}{func[index]}{Style.RESET_ALL}")

        table_data.append(row)

    # Print the table
    print(tabulate(table_data, headers=headers, tablefmt='fancy_grid'))
```

Adina Hessen - 336165139
Ora Wetzler - 212058689

This function prints out all monotonic functions in a table, with colours for easy analysis.

The output came out as following:

Function #	(0, 0, 0, 0)	(1, 0, 0, 0)	(1, 1, 0, 0)	(0, 0, 1, 0)	(1, 0, 1, 0)	(1, 1, 1, 0)	(0, 0, 1, 1)	(1, 0, 1, 1)	(1, 1, 1, 1)
0	0	0	1	0	0	0	0	0	0
1	0	0	1	0	0	1	0	0	0
2	0	0	1	0	0	1	0	0	1
3	0	1	1	0	0	0	0	0	0
4	0	1	1	0	0	1	0	0	0
5	0	1	1	0	0	1	0	0	1
6	0	1	1	0	1	1	0	0	0
7	0	1	1	0	1	1	0	0	1
8	0	1	1	0	1	1	0	1	1
9	1	1	1	0	0	0	0	0	0
10	1	1	1	0	0	1	0	0	0
11	1	1	1	0	0	1	0	0	1
12	1	1	1	0	1	1	0	0	0
13	1	1	1	0	1	1	0	0	1
14	1	1	1	0	1	1	0	1	1
15	1	1	1	1	1	1	0	0	0
16	1	1	1	1	1	1	0	0	1
17	1	1	1	1	1	1	0	1	1

And we can see that there are 18 monotonic functions that fit the conditions, and they are the same as the functions given in the assignment paper.

d