COL215P: Software Assignment - 1
Karnaugh Maps

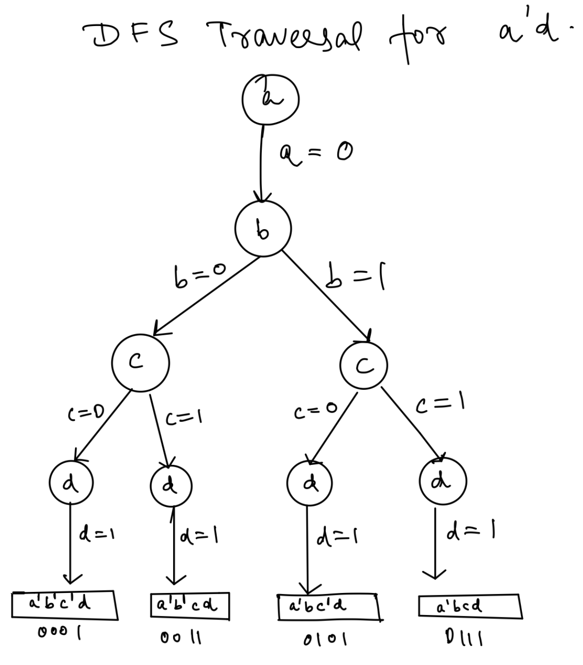
Chinmay Mittal(2020CS10336) and Dhruv Tyagi(2020CS10341)

The main function is *is_legal_region* which is implemented in the file submitted according to the assignment specifications.

Implementing Legality

The first part is to check whether a given term is legal or not. I have implemented this as a generic meaning it works for any number of variables. The column contains *ceil(n/2)* variables and the rows contain the *floor(n/2)* variables. The terms in the column and the rows appear in the order specified by the gray code (each consecutive term differs by 1 bit). I have implemented the gray code in the function ***n_bit_gray_code***. This function is recursive and uses the gray code for n-1 variables to generate the gray code for n variables.

To check legality I generate all cells in the region specified by the term and check whether they have 1s or xs. If any cell has 0 then the region is not legal. To do this for any number of variables I use a dfs traversal where each node represents one variable and the edges represent the assignment to the variable. If the variable in *None* then there are two edges for that variable (for both 0 and 1) otherwise there is only one edge corresponding to the variable which is either 0 or 1. There is a terminal state in which all variables are assigned depending on which edge was traversed to reach that terminal state. This way all the terminal states generated by this dfs corresponds to all the cells specified by the term. If the value in the k_map is 0 in any one of them then the term is not legal otherwise it is legal. The gray-code function discussed before is used to find the index into the k_map at every terminal state.

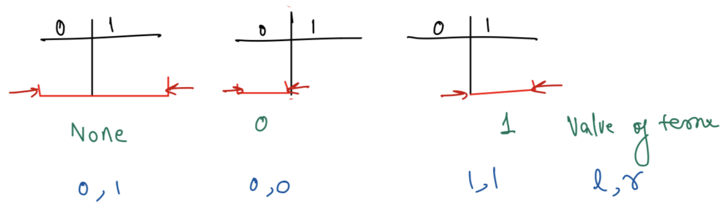


COL215P: Software Assignment - 1
Karnaugh Maps

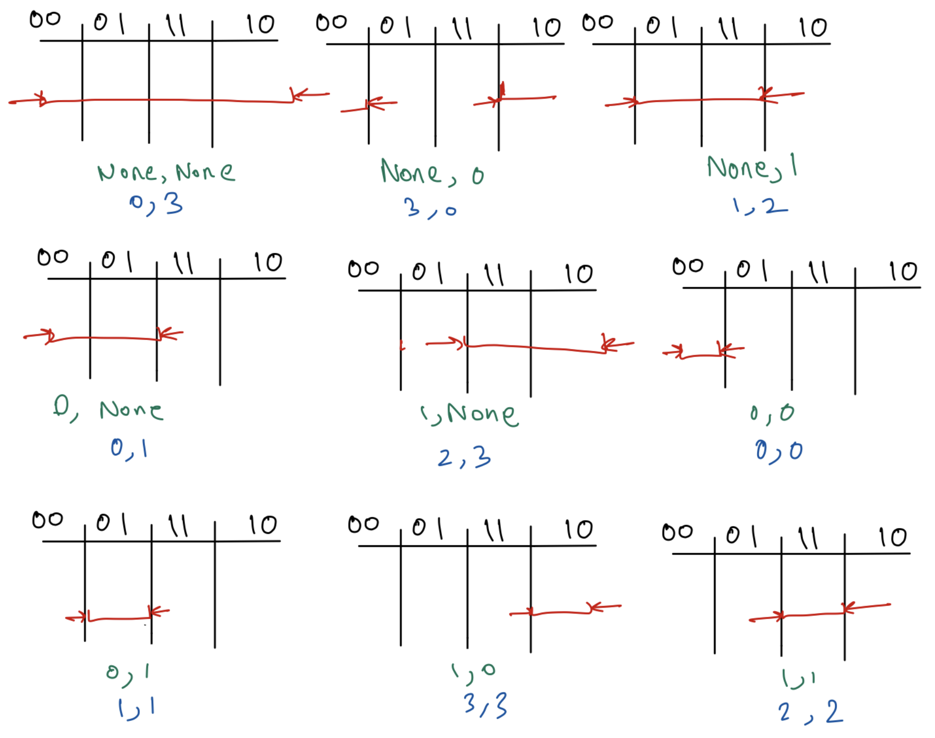
Marking the Region in the K-map

We have implemented this part for only 2/3/4 variables. We need to find four values the top left x and y values and the bottom right x and y values. We have assumed the x-axis to be vertical and the y-axis to be horizontal. The main property we have used is that to decide the value of the x co-ordinates we only have to look at the variables along the rows and to find the value of the y-coordinates we only have to look at the variables along the columns. The way we determine these co-ordinates depends on the number of variables along the axis and not anything else. We have made two helper functions which determine these values (can be reused for x and y values and also for different number of variables), one of these is for the case when there is one variable along the axis and the other is for the case when there are 2 variables along the axis. Both these functions return 2 values the left (top-left co-ordinate) and right (bottom-right co-ordinate) value. These functions are somewhat hard-coded since there are limited possibilities we iterate over them to decide the values of the coordinates.

one-term_helper.

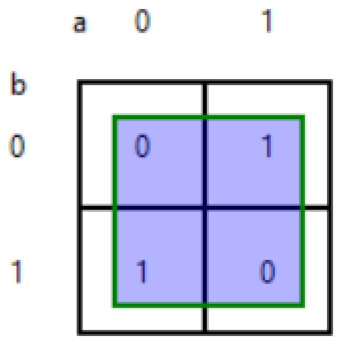


two_term_helper

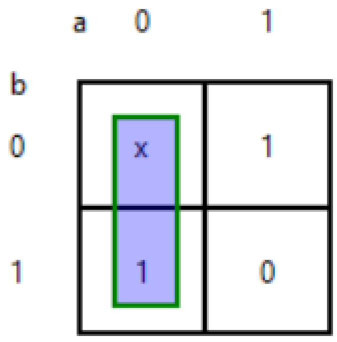


COL215P: Software Assignment - 1
Karnaugh Maps

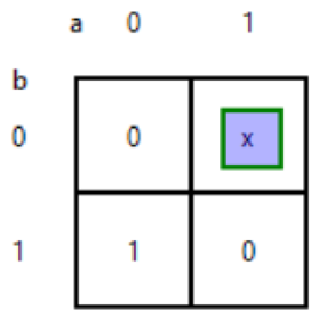
Test Cases



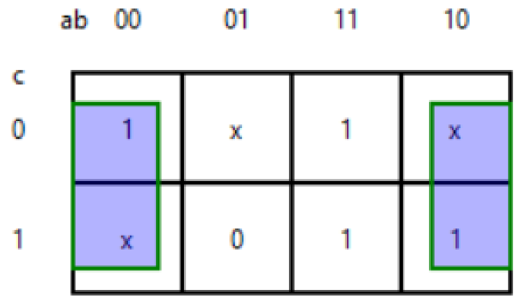
Term: [None, None]
Legal: False
Region: (0, 0) (1, 1)



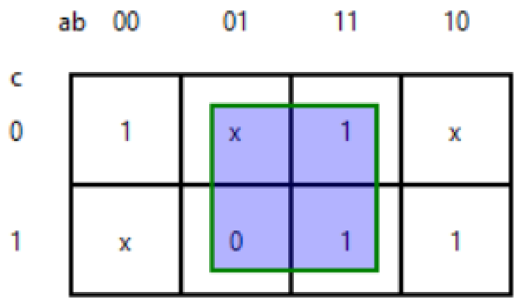
Term: [0, None]
Legal: True
Region: (0, 0) (1, 0)



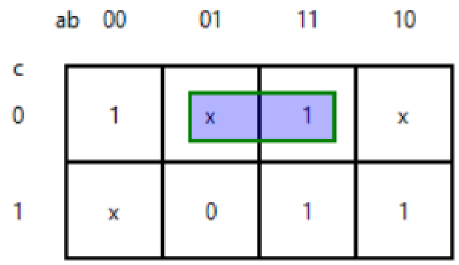
Term: [1, 0]
Legal: True
Region: (0, 1) (0, 1)



Term: [None, 0, None]
Legal: True
Region: (0, 3) (1, 0)

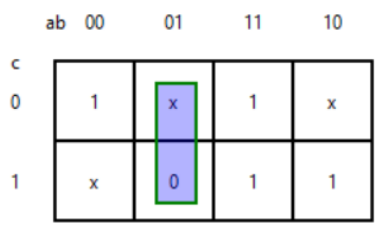


Term: [None, 1, None]
Legal: False
Region: (0, 1) (1, 2)

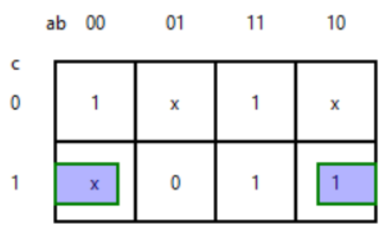


Term: [None, 1, 0]
Legal: True
Region: (0, 1) (0, 2)

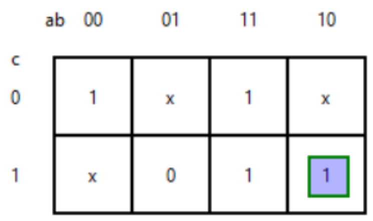
COL215P: Software Assignment - 1
Karnaugh Maps



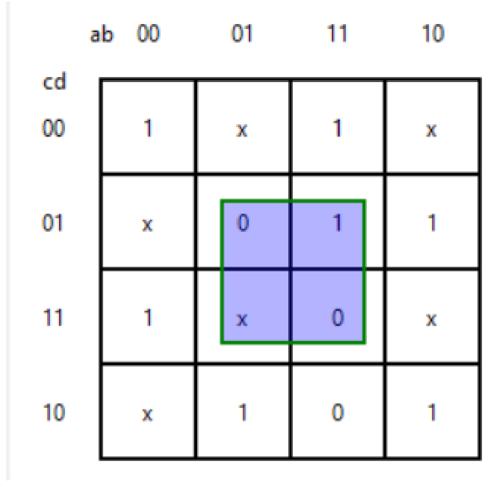
Term: [0, 1, None]
Legal: False
Region: (0, 1) (1, 1)



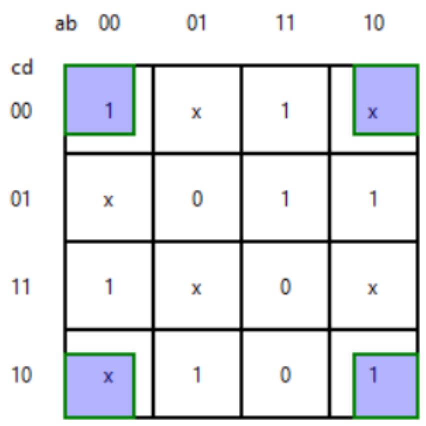
Term: [None, 0, 1]
Legal: True
Region: (1, 3) (1, 0)



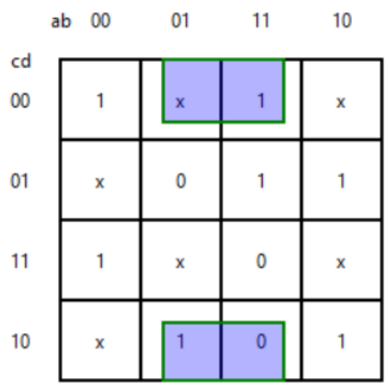
Term: [1,0,1]
Legal: True
Region: (1, 3) (1, 3)



Term: [None, 1, None, 1]
Legal: False
Region: (1, 1) (2, 2)

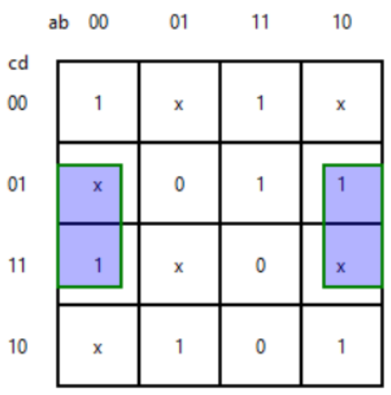


Term: [None, 0, None, 0]
Legal: True
Region: (3, 3) (0, 0)



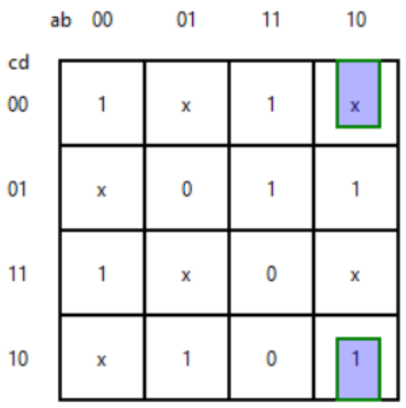
Term: [None, 1, None, 0]
Legal: False
Region: (3, 1) (0, 2)

COL215P: Software Assignment - 1
Karnaugh Maps



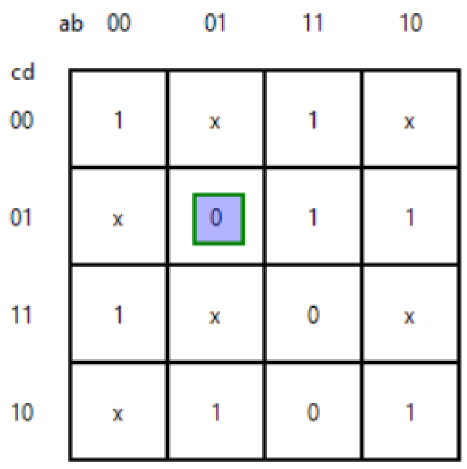
cd \ ab	00	01	11	10
00	1	x	1	x
01	x	0	1	1
11	1	x	0	x
10	x	1	0	1

Term: [None, 0, None, 1]
Legal: True
Region: (1, 3) (2, 0)



cd \ ab	00	01	11	10
00	1	x	1	x
01	x	0	1	1
11	1	x	0	x
10	x	1	0	1

Term: [1, 0, None, 0]
Legal: True
Region: (3, 3) (0, 3)



cd \ ab	00	01	11	10
00	1	x	1	x
01	x	0	1	1
11	1	x	0	x
10	x	1	0	1

Term: [0, 1, 0, 1]
Legal: False
Region: (1, 1) (1, 1)