Question Number 6

Describe a differential attack, analogous to the one described in class for Sypher004, that will find eight subkey bits in the last round. What is the probability of the differential characteristic you devised.

[Hint]: Your differential characteristic must involve two Sbox-es in the 4th round.

Solution. 8-bit Attack

a) First We need to decide the path which suppose to be followed by the Sypher004, up to round 4 output. In our case we are taking path as follows

$$(0,0,2,2) -- > (0,0,0,3) -- > (0,0,1,1) -- > (0,0,3,0) -- > (0,0,2,2)$$

total cost of this path is $(\frac{6}{14})^6 = 0.0027$ (probability that our key guess is correct)

Note: We are analyzing all difference path to get most optimum path for our attack. For analyzing we are using following function form `analyze.py`

- `find_all_paths`
- `dfs`

b) As we know the initial message pair difference and difference at the end of four round which is $(0,0,2,2)$. So we have to create 2^{16} message pair to iterate over to get correct key.

This is done by the `generate_message_pairs` function from `utils.py`

c) Using `message_filter` function from `utils.py` file. Which will reduce number of messages around 7200

d) Then we have to iterate over all keys guess that is from `(0x0110,0x0ff0)` for k5

e) In each iteration We have to initialize one counter and then for each message pair among filtered message pairs

- get ciphertext for each message form pair (c_1, c_2)
- then reverse the last round to get output for fourth round (out_1, out_2)
- Then check that output difference $(out_1 \oplus out_2)$ should be equal to `0x0220`
- if condition satisfy then increase counter

f) When we get counter for each key guess then take one with the highest counter (*bestkeys*) (maybe multiple candidates possible)

g) Cross-check the candidates are with actual key for correct key

h) All This code is automated in `attack.py` file (made faster my filter useful messages)

Note: In this experiment to get path for difference that will have 2 SBox active in last round we have written code in `analyze.py`. This file will try all possible transition and get maximum probability path at last. Observing output we can analyze that by satisfying all above condition we get maximum probability 0.0027 on multiple path. We have taken one of this path.

All the output of this file is already dumped in path.txt. It is recommended to not run analyze.py file as it will take more time to complete execution(as it is checking all path)

Output Image