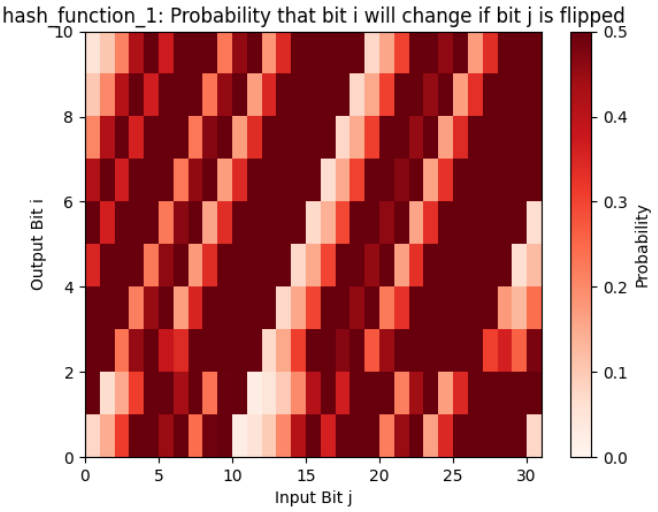
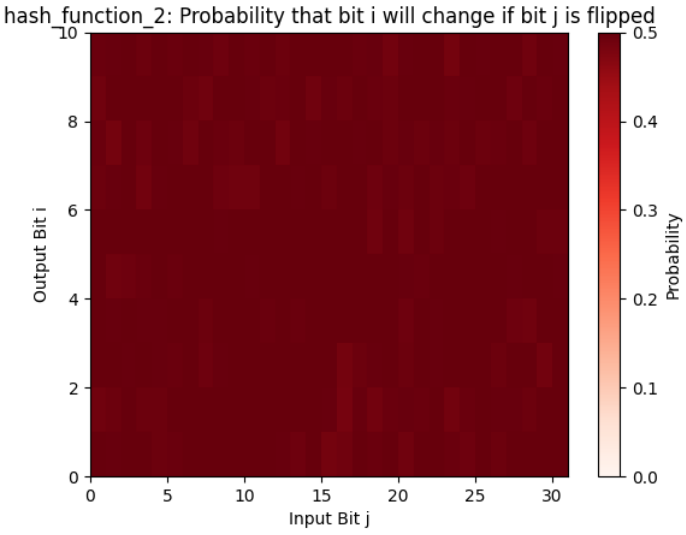
Hash function params and murmur hash seed:
Code: https://github.com/timothyoh777/comp480-hw1, also submitted on canvas

```
MURMUR_HASH_SEED = 1264528
# [a, b, c, d]
HASH_FUNCTION_PARAMS = [79898, 818327, 149554, 269194]
```

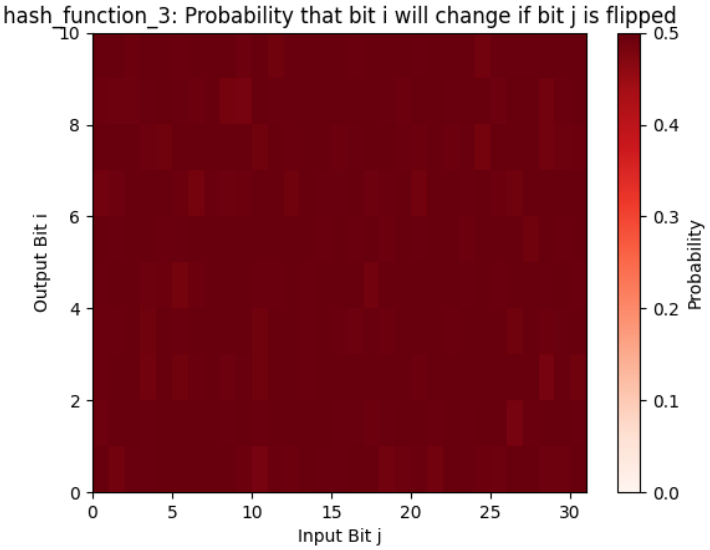
Heat map from `hash_function_1`



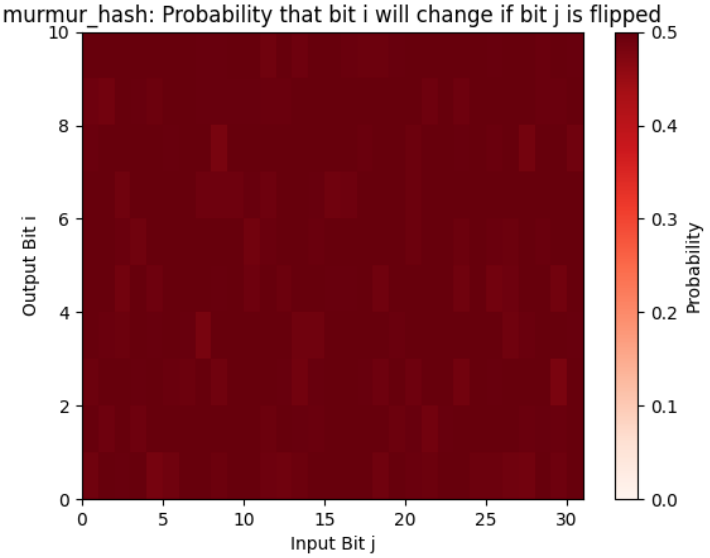
Heat map from `hash_function_2`



Heat map from `hash_function_3`



Heat map from `murmur_hash`



This experiment performs avalanche analysis on 4 different hash functions, a simple 2-universal, 3-universal, 4-universal, and Murmur hash. With 5000 randomly generated keys, we hashed each key, then flipped each of the key's bits and hashed them again. Then, we checked for each bit out the output of our hash function, to see if a bit flip of input bit j resulted in a flipped but in output but i. We then mapped the probability in a heat map with 0.5 probability being the darkest, as an ideal hash function achieves P(Output bit i changes | Input bit j changes) = 0.5 $\forall$ i, j.

We can notice from the plots that our 2-universal hash function performs poorly in achieving avalanche effect. The probability that an output bit i changes given that input j changes is not near 0.5 for all i, j. We can also notice linear patterns in our first heatmap; this suggests that a certain bit j being flipped suggests that a certain bit i might be flipped or not, decreasing the security and effectiveness of our first simple congruential linear hash function. However, in contrast to our 2-universal hash function's performance, our other three hash functions perform very well in our avalanche analysis. For all i and j, we can see that the probability that output bit i is flipped given that input bit j is flipped is all around 0.5. This suggests that at least in our small scale experiment, our 3 and 4 universal hash function and murmur hash displays avalanche effect strongly; we can "predict" the state with around 0.5 probability, which suggests that a small change to the input of our hash function leads to a random significant, change our output hash.