Multicore Homework 2

David Chun (dc37875)
Grace Zhuang (gpz68)

Question 1

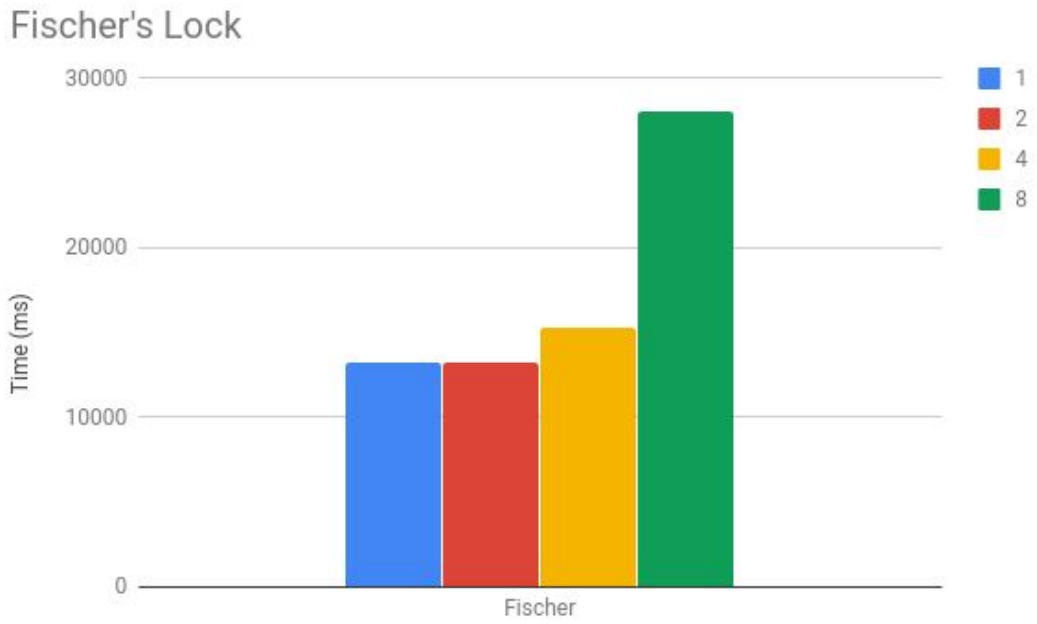
Build a triangular nxn 2d array of splitters (such that there are *n(n+1)/2 splitters*) and assign each splitter a *unique* number from 0 up to n(n+1)/2. When a thread calls ***int rename(int j)***, it starts at the root splitter at [0, 0] in the matrix. If it stops there, ie if there is no thread that already went DOWN, it returns the unique number assigned to the splitter in the initialization. Otherwise, it traverses the matrix according to the direction specified by the splitter outcome, LEFT or RIGHT until it stops at a splitter (if the thread is at [0,0], LEFT would move to [1,0] and RIGHT would be [0,1]), returning the value assigned to that splitter as the new index.

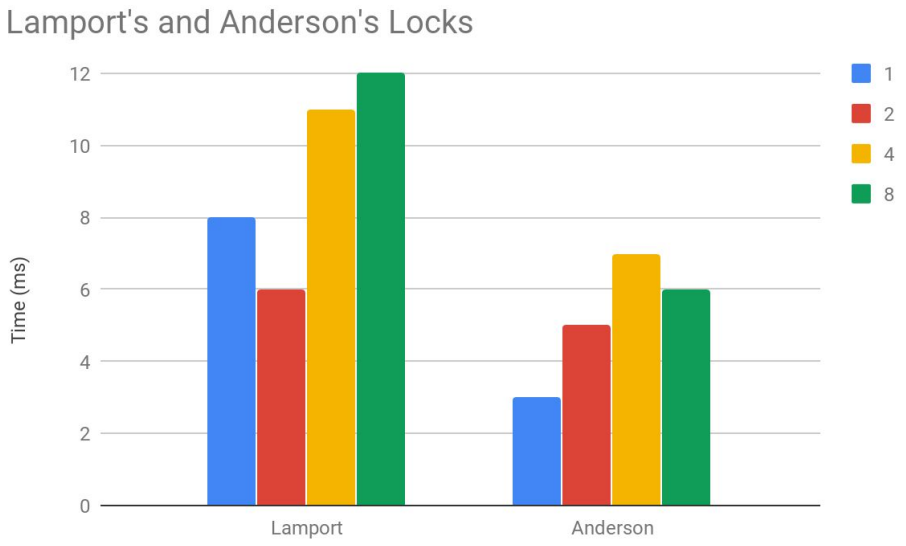
This solution satisfies the index independence property since the algorithm never uses a thread's original index to make any decisions on which direction the thread moves in the matrix and the indexes are independently assigned to the matrix splitter. Thus any thread could potentially be assigned any new index value. Additionally, no two threads can obtain the same new index as only one thread can be stopped at a splitter.

Question 2

**Calculated using n=12000

Lock	1	2	4	8
Lamport	8	6	11	12
Anderson	3	5	7	6
Fischer	13278	13260	15284	28043





Question 4a

Threads	Time (ms)
1	7.2602
2	2.8003
3	3.683
4	3.1335
5	3.5406
6	3.0169
7	3.0102
8	3.8997