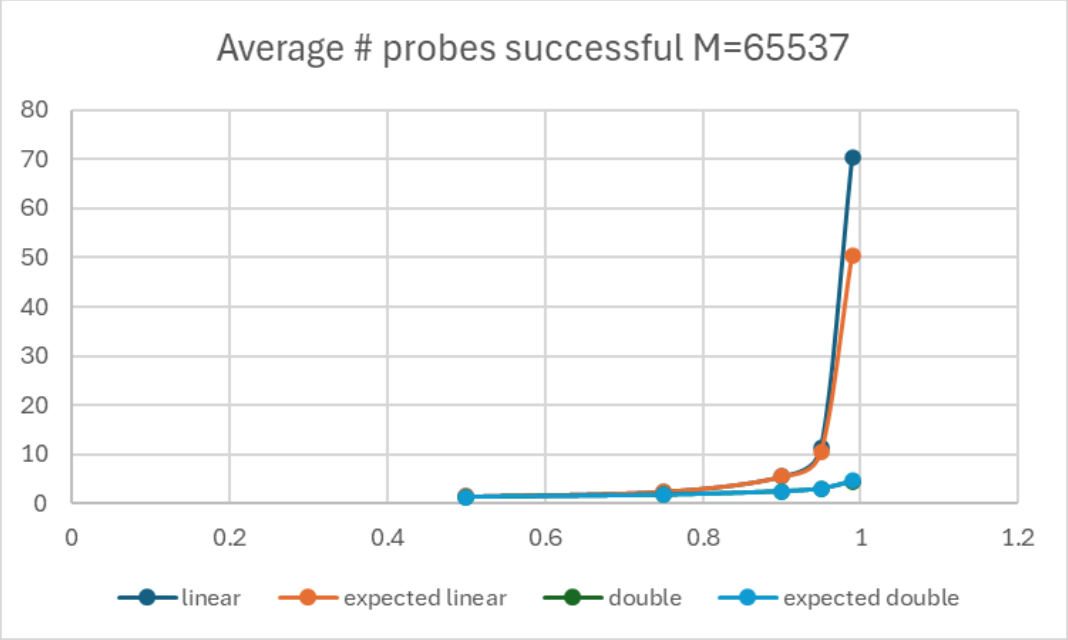
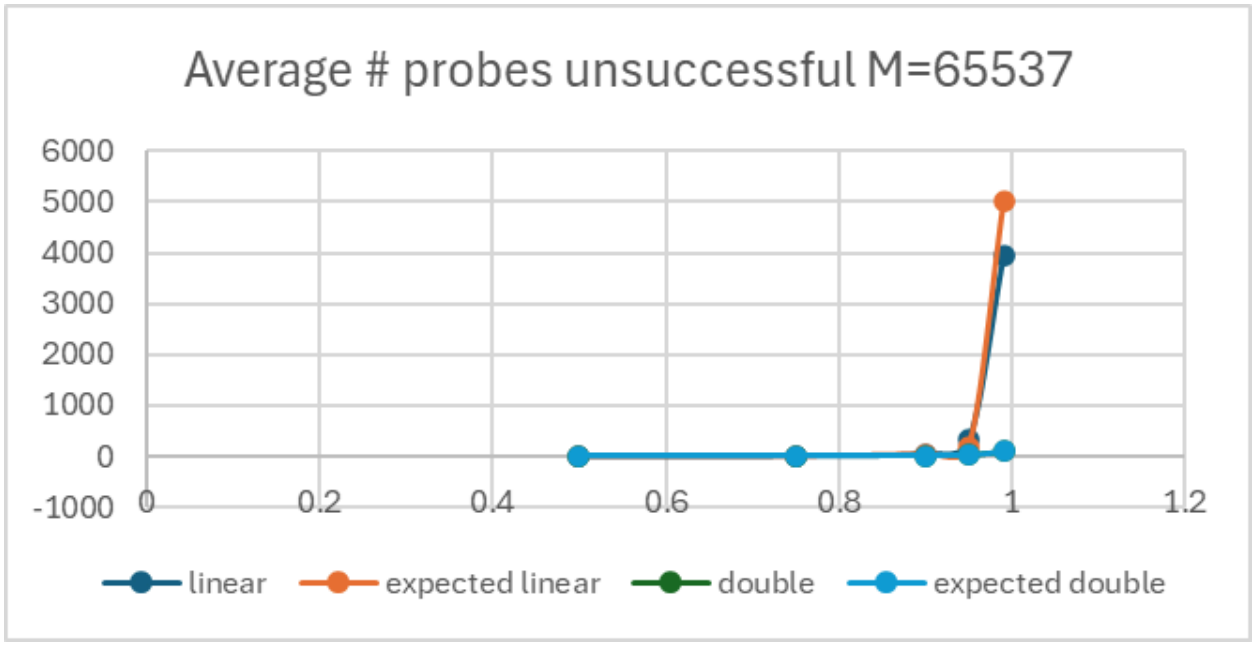
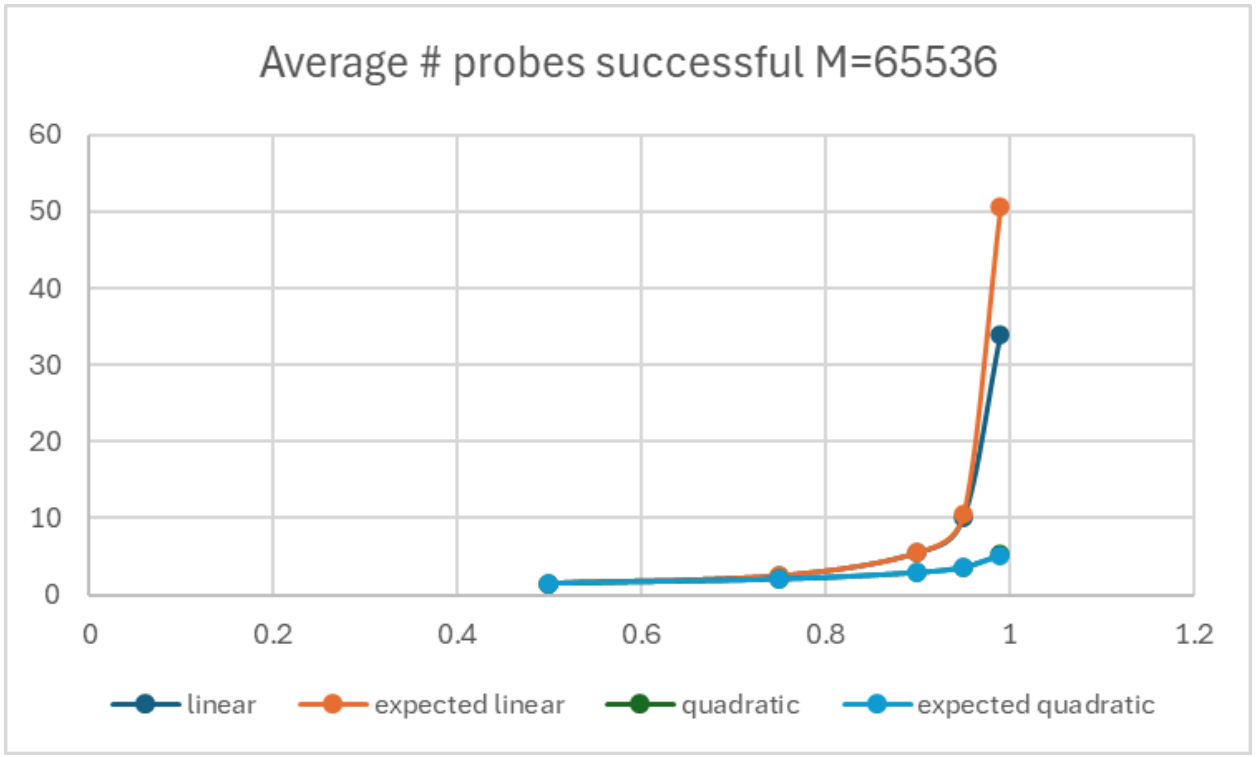
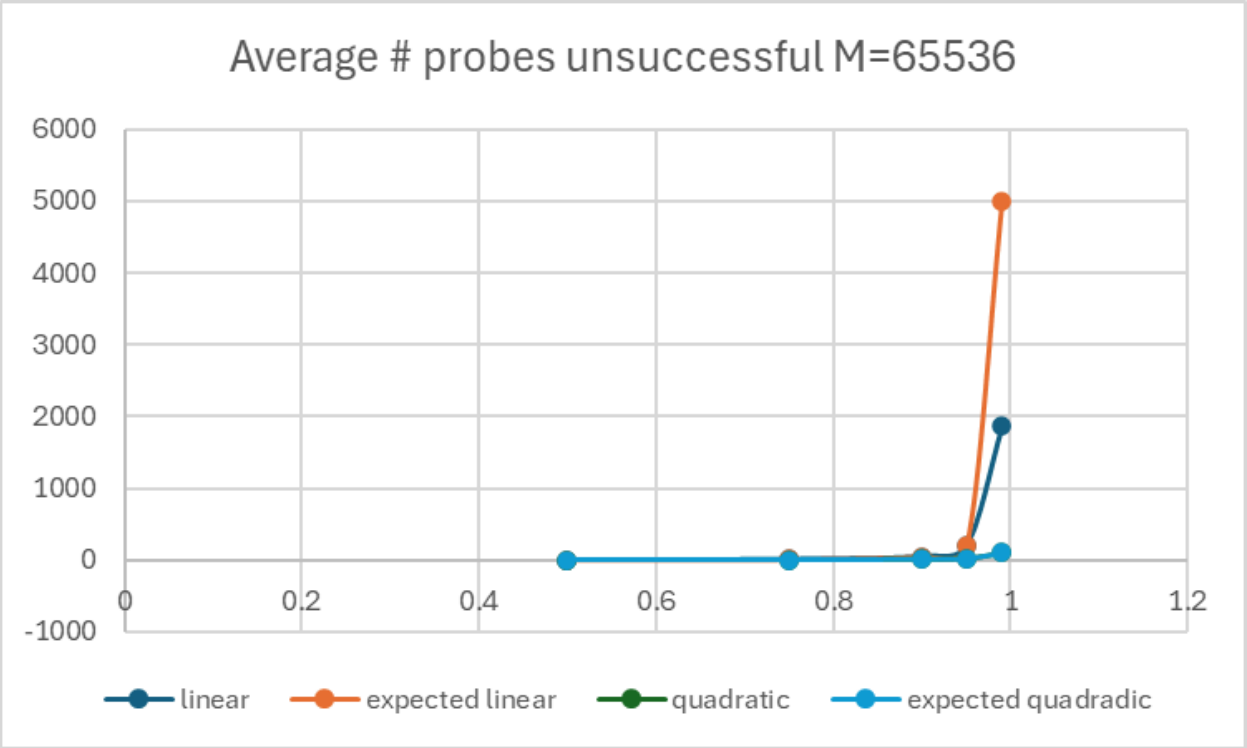
Drew Buley
C20407096
12/5/2024

1) Initially I tested the response of successful and unsuccessful searches for random key values at different load levels. As double requires a prime table size, and quadratic requires a power of two, these could not be shown on the same graph. From observation of the data directly, performed slightly better than quad for all load levels (but within 5% difference to quad). Additionally, I noticed that for high load levels, linear unsuccessful searches performed much better (often under half) than the expected values. I am not sure what the largest contributing factor to this was.




Average # probes unsuccessful M=65537
6000
5000
4000
3000
2000
1000
0
-1000
0
0.2
0.4
0.6
0.8
1
1.2
linear
expected linear
double
expected double


Average # probes successful M=65536
60
50
40
30
20
10
0
0
0.2
0.4
0.6
0.8
1
1.2
linear
expected linear
quadratic
expected quadratic



2)

Successful searches

	random	sequential	folded	worst
quadratic	2.4	1	112	11669
double	2.22	1	4.22	3.38
linear	3.83	1	13927	24521

Unsuccessful searches

	random	sequential	folded	worst
quadratic	7.94	190	191	7.39
double	6.67	15.02	7.64	6.6
linear	23	23778	23605	23771

Analysis of time complexity:

From this spread, we can see that non-random keys had a huge effect on linear probing (and has demoted it to O(n) due to the large amount of primary clustering). Quadratic probing also suffered slightly from the worst case and shifted in complexity to something slower than constant time. For random insertion, all 3 certainly made a good case of being constant time. Double ended up being the most consistent at dodging worst cases and maintaining time

complexity O(1). Interestingly, for sequential keys, it took 1 probe for the algorithms to successfully find their target. I would extrapolate that for most hash probe calculations, if the table had no collisions during insertions, it would also experience this behavior.

3) First output is 50000 trials, second is 200000 trials

```
drewcoding@drewcoding-VirtualBox:~/ece2720/mp6-masterarcher300$ ./lab6 -e -m 65537 -t 50000
Open addressing with linear probe sequence
Using hash algorithm abs_hash
Seed: 11172024

----- Equilibrium test driver -----
Table size (65537), load factor (0.9)
  Trials: 50000
  Build table with 58983 random keys
during random build generated duplicate key (90672838) on trial (41709)
this should be unlikely: if see more than a few you have a problem
    The average number of probes for a successful search = 5.57969
  Keys added (25189), removed (24809) new size should be (59363) and is (59363)
  After exercise, time=15.263
  successful searches during exercise=10.1935, trials=24811
  unsuccessful searches during exercise=205.04, trials=25189
  After retrieve experiment, time=38.276
  New load factor = 0.905794
  Percent empty locations marked deleted = 89.1156
   Measured avg probes for successful search=14.2299, trials=59363
   Measured avg probes for unsuccessful search=394.163, trials=49999
    Do deletions increase avg number of probes?
--- Linear probe sequence performance formulas ---
    Expected probes for successful search 5.8075
    Expected probes for unsuccessful search 56.8391
  Rehash table
  After rehash, time=9.102
   Measured avg probes for successful search=6.70438, trials=59363
   Measured avg probes for unsuccessful search=81.3546, trials=49998
----- End of equilibrium test -----

drewcoding@drewcoding-VirtualBox:~/ece2720/mp6-masterarcher300$
```

```
drewcoding@drewcoding-VirtualBox:~/ece2720/mp6-masterarcher300$ ./lab6 -e -m 65537 -t 200000
Open addressing with linear probe sequence
Using hash algorithm abs_hash
Seed: 11172024

----- Equilibrium test driver -----
Table size (65537), load factor (0.9)
  Trials: 200000
  Build table with 58983 random keys
during random build generated duplicate key (90672838) on trial (41709)
this should be unlikely: if see more than a few you have a problem
    The average number of probes for a successful search = 5.57969
  Keys added (99872), removed (100123) new size should be (58732) and is (58732)
  After exercise, time=1525.91
  successful searches during exercise=16.2222, trials=100128
  unsuccessful searches during exercise=5557.73, trials=99872

  After retrieve experiment, time=7975.13
  New load factor = 0.896166
  Percent empty locations marked deleted = 99.8677
   Measured avg probes for successful search=19.7991, trials=58732
   Measured avg probes for unsuccessful search=7187.59, trials=199991
    Do deletions increase avg number of probes?
--- Linear probe sequence performance formulas ---
    Expected probes for successful search 5.31536
    Expected probes for unsuccessful search 46.8753
  Rehash table
  After rehash, time=18.567
   Measured avg probes for successful search=5.04534, trials=58732
   Measured avg probes for unsuccessful search=44.2795, trials=199995
----- End of equilibrium test -----

drewcoding@drewcoding-VirtualBox:~/ece2720/mp6-masterarcher300$
```

With the massive increase to trials, we went from 90% of empty locations being marked as empty to 99%. For the first experiment, rehashing took the time from 15.26 to 9.1. For the second, reshahing was much more impactful and took the time from 1525 to 18.5. This likely was a result of faster times to find empty cells (and thus stop inserts/deletes/searches earlier).

4) As you can see from this screenshot of my terminal, my two sums solution passes all tests from the previous page.

```
drewcoding@drewcoding-VirtualBox:~/ece2720/mp6-masterarcher300$ time ./lab6 -p4 -m 1000000 -t 20 -h double
Open addressing with double hashing
Using hash algorithm abs_hash
Seed: 11172024
Two Sum Driver -p 4 for -t 20 trials array size -m 1000000
    All trials passed!

real    0m2.531s
user    0m2.497s
sys     0m0.032s
drewcoding@drewcoding-VirtualBox:~/ece2720/mp6-masterarcher300$ time ./lab6 -p4 -m 10000 -t 5000 -h double
Open addressing with double hashing
Using hash algorithm abs_hash
Seed: 11172024
Two Sum Driver -p 4 for -t 5000 trials array size -m 10000
    All trials passed!

real    0m1.949s
user    0m1.949s
sys     0m0.000s
drewcoding@drewcoding-VirtualBox:~/ece2720/mp6-masterarcher300$ time ./lab6 -p4 -m 100 -t 500000 -h double
Open addressing with double hashing
Using hash algorithm abs_hash
Seed: 11172024
Two Sum Driver -p 4 for -t 500000 trials array size -m 100
    All trials passed!

real    0m2.102s
user    0m2.095s
sys     0m0.005s
drewcoding@drewcoding-VirtualBox:~/ece2720/mp6-masterarcher300$
```