

Palindromes and semordilaps

Method

To find the palindromes and semordilaps I use two lists. The one that contains all the words to check if they are palindromes or semordilaps and the whole dictionary in the other one, to even with a smaller dictionary subset, be able to check against the whole thing.

The program goes through all the words to check through in a parallelized for loop, and checks if the reverse of the word is contained in the dictionary with binary search. If it is the word is added to a private vector and after the loop the private vector is added to the end of a shared vector in a critical section.

I use binary search for efficiency, with a naive checking algorithm the performance gains from parallelization are even more obvious.

Testing

To test the program I chose to run it for 1, 4, 8, and 16 cores. For each of these I also ran it on a different data set 100, 1000, 10 000 and 100 000 words. Each variation was run 5 times and the median time is shown in this table.

Size	1 thread	4 threads	8 threads	16 threads
100	25	32	36	37
1 000	181	216	341	677
10 000	1 408	932	823	4262
100 000	12 217	3730	2634	2479

Note: Time in μ s

At size 100 the program has a condition to not run in parallel, as I saw a negative speedup when using more cores, probably due to openmp thread manging overhead and quite big critical section. This can still be seen on the dictionary of 1 0000.

With the 10 000 size dictionary we see a speedup and then a sudden drop in performance. I think that this could be attributed to the more time spent in the critical section and processes blocking each other there.

Then in the 100 000 size we see pretty much ideal speedup going from 1 to 4 cores, but then diminishing results. I assume again, the problem with writing all the data to one vector in a critical section.

All the data screenshots:

```
mint@mint-Legion-7-16ACHg6:~/Documents/concurrent_programing/HM2$ ./palindromes_par 100 1
it took 2.4724e-05 seconds and 100
```

```
mint@mint-Legion-7-16ACHg6:~/Documents/concurrent_programing/HM2$ ./palindromes_par 100 4
it took 3.1987e-05 seconds and 100
```

```
mint@mint-Legion-7-16ACHg6:~/Documents/concurrent_programing/HM2$ ./palindromes_par 100 8
it took 3.5759e-05 seconds and 100
```

```
mint@mint-Legion-7-16ACHg6:~/Documents/concurrent_programing/HM2$ ./palindromes_par 100 16
it took 3.7155e-05 seconds and 100
```

```
mint@mint-Legion-7-16ACHg6:~/Documents/concurrent_programing/HM2$ ./palindromes_par 1000 1
it took 0.000181097 seconds and 1000
```

```
mint@mint-Legion-7-16ACHg6:~/Documents/concurrent_programing/HM2$ ./palindromes_par 1000 4
it took 0.000216576 seconds and 1000
```

```
mint@mint-Legion-7-16ACHg6:~/Documents/concurrent_programing/HM2$ ./palindromes_par 1000 8
it took 0.0003418 seconds and 1000
```

```
mint@mint-Legion-7-16ACHg6:~/Documents/concurrent_programing/HM2$ ./palindromes_par 1000 16
it took 0.000677245 seconds and 1000
```

```
mint@mint-Legion-7-16ACHg6:~/Documents/concurrent_programing/HM2$ ./palindromes_par 10000 1
it took 0.00140855 seconds and 10000
```

```
mint@mint-Legion-7-16ACHg6:~/Documents/concurrent_programing/HM2$ ./palindromes_par 10000 4
it took 0.000932931 seconds and 10000
```

```
mint@mint-Legion-7-16ACHg6:~/Documents/concurrent_programing/HM2$ ./palindromes_par 10000 8
it took 0.00082398 seconds and 10000
```

```
mint@mint-Legion-7-16ACHg6:~/Documents/concurrent_programing/HM2$ ./palindromes_par 10000 16
it took 0.00426223 seconds and 10000
```

```
mint@mint-Legion-7-16ACHg6:~/Documents/concurrent_programing/HM2$ ./palindromes_par 100000 1
it took 0.0122172 seconds and 100000
```

```
mint@mint-Legion-7-16ACHg6:~/Documents/concurrent_programing/HM2$ ./palindromes_par 100000 4
it took 0.00373082 seconds and 100000
```

```
mint@mint-Legion-7-16ACHg6:~/Documents/concurrent_programing/HM2$ ./palindromes_par 100000 8
it took 0.00263432 seconds and 100000
```

```
mint@mint-Legion-7-16ACHg6:~/Documents/concurrent_programing/HM2$ ./palindromes_par 100000
it took 0.00247948 seconds and 100000
```