Sayemum Hassan

Dr. Kevin Molloy

CS412 – Applied Algorithms

29 August 2024

CS412 - Lab 2 Report

List

```
37 function calls in 11.762 seconds

   Ordered by: cumulative time

   ncalls  tottime  percall  cumtime  percall filename:lineno(function)
        1    0.000    0.000   11.762   11.762 {built-in method builtins.exec}
        1    0.000    0.000   11.762   11.762 cs412_foxsays_list.py:1(<module>)
        1    0.000    0.000   11.762   11.762 cs412_foxsays_list.py:13(main)
        7   11.762    1.680   11.762    1.680 {built-in method builtins.input}
        6    0.000    0.000    0.000    0.000 {method 'split' of 'str' objects}
        2    0.000    0.000    0.000    0.000 {built-in method builtins.print}
       16    0.000    0.000    0.000    0.000 {method 'append' of 'list' objects}
        2    0.000    0.000    0.000    0.000 {method 'join' of 'str' objects}
        1    0.000    0.000    0.000    0.000 {method 'disable' of '_lsprof.Profiler' objects}
```

The snippet above shows a profiling analysis of all function calls and how long it took the program to run. The ‘list’ version had 37 function calls and took 11.762 seconds to run.

Dictionary

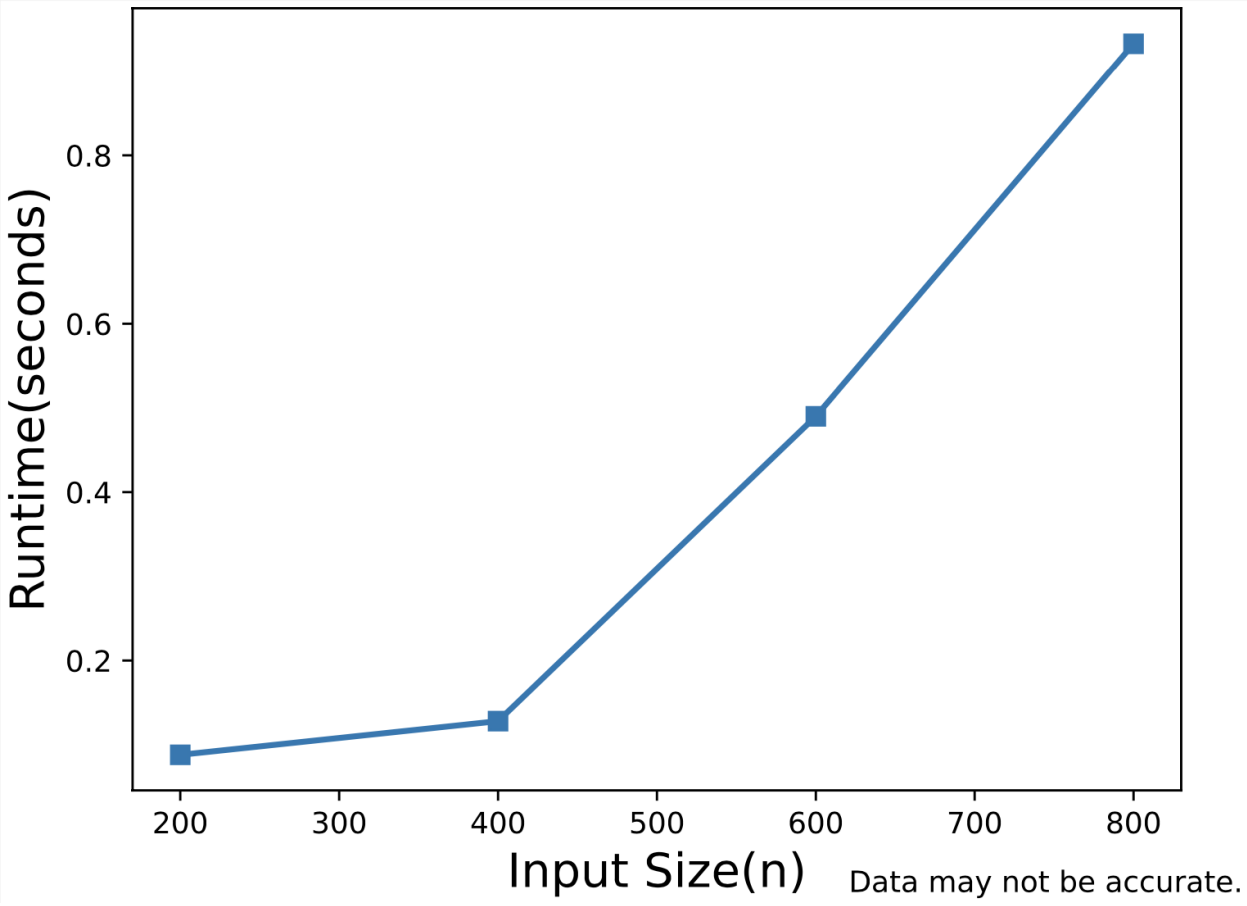
```
55 function calls in 13.271 seconds

   Ordered by: cumulative time

   ncalls  tottime  percall  cumtime  percall filename:lineno(function)
        1    0.000    0.000   13.271   13.271 {built-in method builtins.exec}
        1    0.000    0.000   13.271   13.271 cs412_foxsays_dict.py:1(<module>)
        1    0.000    0.000   13.271   13.271 cs412_foxsays_dict.py:13(main)
        7   13.271    1.896   13.271    1.896 {built-in method builtins.input}
        2    0.000    0.000    0.000    0.000 {built-in method builtins.print}
        6    0.000    0.000    0.000    0.000 {method 'split' of 'str' objects}
       18    0.000    0.000    0.000    0.000 {method 'get' of 'dict' objects}
       16    0.000    0.000    0.000    0.000 {method 'append' of 'list' objects}
        2    0.000    0.000    0.000    0.000 {method 'join' of 'str' objects}
        1    0.000    0.000    0.000    0.000 {method 'disable' of '_lsprof.Profiler' objects}
```

The snippet above shows the original ‘dict’ version of the program with 55 function calls done in 13.271 seconds. This surprised me because using dictionaries is supposed to be faster than Python lists. But in this case, dictionaries are slower because I combined lists and dictionaries and made too many calls.

Dictionary Performance Analysis

As mentioned above, the original ‘dict’ version was very slow in comparison to the ‘list’ version. The input function was the only function call impacting the cumulative time so I didn’t make modifications to the dictionary python file.

Plot Analysis



I was stumped at the part where I had to use split, wc, and head UNIX commands to make smaller input files. Because of this, I used my own input data to assume the runtime for all

the input sizes (200k, 400k, 600k, 800k). From analyzing my dict python file, it appears to have a runtime of roughly O(3n), which simplifies to O(n).