COMP90049 - Project 1

Student ID: 879849

Name: Xiao Shi

How to run

This program is built with the Go programming language and the frontend use Angular 1 to analyse the output json result.

1. Install Go, https://golang.org/dl/
2. Run `go build` in the project directory
3. Move the output executive to `_` directory
4. Run the executive
5. Browse `localhost:3000/viewer.html`

This program uses `sort.Slice()`, so it requires Go 1.8+. The frontend uses `async` and `fetch`, therefore please ensure you are using a modern browser, or you will not see the analyse result.

Performance

Developed and tested under macOS 10.13.4. The machine used is 2.6GHz Intel Core i7 (4 cores, 8 threads) with 16GB memory.

The program is written to fully utilise the multi-threading performance and consume at most 8.5GB memory. As the machine is 4 cores 8 threads, the program is always using around 770% CPU load, and can finish the 10 defined task in 3m30s (GED in 1m20s). So I think its performance is **quiet good**. The program is not much optimized for memory usages, as the load is easy for my machine.

Complex description of How to run

1. Install Go 1.8+, https://golang.org/dl/
2. Run `$ go build` in this directory
3. You will get an executive with name as the directory's name, `COMP90049` by default
4. Move the executive under `_` directory
5. Provided input data, viewer app and task config are in the `_` directory
6. Run the executive

7. You will get a `result.json`
8. (If using mac) The executive will also open browser and navigate to `localhost:3000` to show result
9. You can use `localhost:3000/viewer.html` to see the result detail

Sample Output

Here is a sample output I run and use in my report.

```
COMP90049 - Project 1
Student ID: 879849
Name: Xiao Shi

Launch: DirectMatch
Prepare: 215ns
648.92µs - 673/716
Complete: 493.632µs
Total: 1.196563ms

Launch: Neighbour(K=1)
Prepare: 174ns
24.794298ms - 716/716
Complete: 297.544µs
Total: 25.12983ms

Launch: Neighbour(K=2)
Prepare: 181ns
7.657576899s - 716/716
Complete: 632.264µs
Total: 7.658246504s

Launch: nGram(N=2)
Prepare: 886.093541ms
41.120857347s - 716/716
Complete: 323.177µs
Total: 42.007346101s

Launch: nGram(N=3)
Prepare: 971.87014ms
36.963994314s - 716/716
Complete: 499.54µs
Total: 37.936409353s

Launch: nGram(N=4)
Prepare: 608.106987ms
34.531673577s - 716/716
```

```
Complete: 1.724175ms
Total: 35.141550808s

Launch: GED
Prepare: 99.821065ms
1m16.159059734s - 716/716
Complete: 931.288µs
Total: 1m16.259851377s

Launch: Soundex(Cut=4)
Prepare: 667.921276ms
1.219183641s - 716/716
Complete: 273.262µs
Total: 1.887420079s

Launch: Soundex(Cut=8)
Prepare: 604.614093ms
1.220018637s - 716/716
Complete: 224.46µs
Total: 1.824904882s

TIME: 3m23.449191643s
```

Wed, 11 Apr 2018