Keyboard optimisation and visualisation

EE2016 Applied Programming Lab '24

Sanjeev Subrahmaniyan S B

EE23B102

1 Objective/ Problem Statement

Optimise the QWERTY layout keyboard for a given input string. The objective is to minimise the travel distance of the fingers as the string is typed. This is achieved by using a simulated annealing approach.

2 Solution Idea

2.1 Simulated Annealing

The problem under consideration is NP-hard and hence does not have any polynomial-time optimisation algorithm. Hence, simulated annealing is chosen as an attempt to perform the optimisation. The steps involved in the implementation are:

1. Receive a multiline string as input. Parse the input, create a dictionary of key press frequencies. Build a general layout for the keyboard.
2. Start by making a copy of the current keyboard layout. (All of these steps are identical to the solution of the previous assignment).
3. Two keys are randomly chosen from the keyboard and the new costs(the travel distance after the switching is computed). The switching is done such that all dependencies on the switched keys also switch. For instance, if 'f' and 'q' are switched, 'q' is now a homerow key over which the left index finger hovers.
4. If this cost is lesser than the current best cost, this layout is accepted. Otherwise, it is accepted with a probability that exponentially decreases with time(see the probability function in the code implementation).
5. The probability function is designed to decrease with progress of the optimisation - this reduces the probability of an non optimal step being performed as time progresses.
6. Note that the probability function I have chosen is different from the one used in the demonstration. This is because it provides a better optimisation over a fixed number of iterations is what I observe.

7. This process is continued until a maximum number of iterations are reached or when the cost becomes a specified fraction of the initial cost. This final layout will be considered the optimised layout.

2.2 Visualisation

The visualisation of the optimised layout is done as:

1. Create a heatmap visualisation of the optimised layout. An animation is made which shows the evolution of cost functions(both the best cost functions and the cost that is evaluated after every key switch instance).
2. The notebook also has a functionality which shows the evolution of the keyboard and heatmap itself. This part takes a while to run and is normally disabled.
3. The visualisation is in the form of a HTML as demonstrated in the class. The evolution of the cost functions can be viewed by pressing the appropriate buttons.

3 Key asssumptions

There are a number of assumptions that went into this solution due to this being a one week assignment. The most important ones are:

1. The foremost assumption is that all the keys are of the same size and any two keys on the keyboard may be switched - specifically, keys such as Spacebars and Shifts can be freely moved to anywhere on the keyboard.
2. Secondly, it is assumed that the optimisation is to reduce the travel distance of the typist - this does not guarantee even load distribution and such(while this is practically important - typing on one hand if about half as fast).
3. The input line contains only English alphabets and special characters on the standard QWERTY keyboard. Other language characters or special characters not covered in such a keyboard are assumed to not be used in the analysis.
4. The input can be a multiline string given it adheres to the following format: The multiline input can be given line after line, and an empty line is assumed to mark the end of input. While processing this text, I deliberately choose to ignore the newline character because an Enter key is not specified in the given QWERTY keyboard layout.
5. It is also assumed that keys that are not specified in the keys and characters dictionary will never be pressed. They are not valid keys for the execution.

6. The Gaussian heatmap is made to ensure the output looks smooth and informative at the same time. It is assumed that the reader can understand the heatmap from the reference chart on the sides. Specifically, the lowest part of the spectrum should be ignored(deep purple in my case), which is created to smoothen the effect.

4 Finding and results

The key result is as expected - over a reasonable duration of optimisation, the most frequently pressed keys are on the home row and the other keys are closer to the home row when they occur with high frequency. Specifically, because the Spacebar is very commonly used in texts, it almost always finds a place in the home row. Parallelly, less frequently used characters end up farthest from the homerow.

This can be bypassed by assuming that the Spacebar is already a homerow key(with the thumb being used) or that it cannot move over optimisation. Given that the QWERTY layout given considers spacebar as a non homerow key, I will not consider it as well.

I also note that while the problem as such is NP hard, simulated annealing can often reduce the cost to 30 percent of the initial value in less than 500 iterations(this is the reason I chose those initial conditions). In general, as the number of iterations increases, the cost drastically decreses for most types of inputs.

Another noteworthy observation is due to the randomness of the method - sometimes, for long periods of iterations, there is no improvement in the best cost. This is often followed by a large fall in the cost as well. This implies that it best to run the optimisation for a large number of iterations to get a good optimisation.

On the temperature schedules, I see that using a higher cooling rate(lower value) often results in a suboptimal solution while a lower rate leads to impractical optimisation times.

Finally, in most of the cases, the keys that end up in the home row are the same as the ones that are in the home row of the DVORAK keyboard. This provides insights to why DVORAK can be much faster for english text.

5 Conclusion

In all, I appreciate the applicability of simulated annealing to solve an optimisation problem. I also gained insights on optimal keyboard design for speed.