Keyboard Layout Optimization using Python

Siva Sundar, EE23B151

October 13, 2024

1 Abstract

This project focuses on optimizing a keyboard layout based on minimizing the total travel distance of the fingers while typing a given input string. The aim is to reorganize the keys on the keyboard such that more frequently used keys are positioned on the home row to reduce finger movement. This is achieved through a technique called **Simulated Annealing (SA)**, which iteratively attempts to minimize the total travel distance of fingers.

2 Introduction

In this experiment, a crucial part of optimization is done with the help of Simulated Annealing, which is a method inspired from the annealing process in metallurgy.

Annealing is a heat treatment process used in metallurgy to alter the physical and sometimes chemical properties of a material, typically metals. The primary goal is to increase ductility and reduce hardness, making the material more workable. (Workable in the sense, we can convert it into a desired shape).

Figure 1: Hot metal can be bent easily to get required shapes.

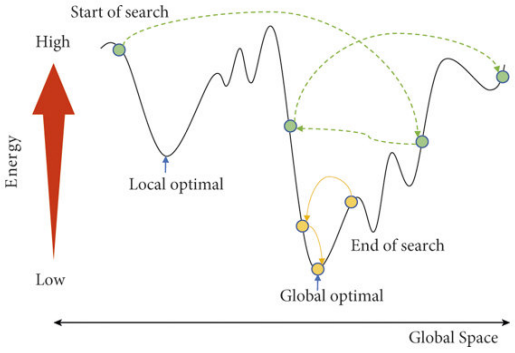

Figure 2: Higher energy levels allow for larger jumps between solutions

In **Simulated Annealing**, we mimick this process using a temperature variable to find an approximate global optimum for large and complex problems. The analogy to the metallurgical process is:

"At high temperatures, the algorithm explores thorugh a wide range of possible solutions, even accepting worse solutions with a certain probability to **avoid getting stuck in local minima**. As the temperature decreases, the probability of accepting worse solutions reduces, focusing more on refining the current solution."

To narrow down the solutions, the initial solution given to SA is the keyboard layout with **highly frequent keys** for the given string, as **home row keys**. Now, SA only tries to shift the other keys and find the best layout. With this, we find the optimized keyboard layout which has least distance for a given string input.

3 Methodology

The approach for this assignment involves the following key steps:

1. **Data Extraction and Key-press identification:** Utilize Python to read data from command line and find the corresponding key presses used by the typist to type each of the letters in the input. To find the key presses, we use the given layout.
2. **Finding the key-press frequency for each key:** After knowing the sequence of key presses, we can find the frequency of key press for each key by iterating through the sequence.
3. **Finding the initial layout with home keys as frequent keys**: From the frequency dictionary, we can find the highly frequent keys. We then replace the home keys of the current layout by these keys.

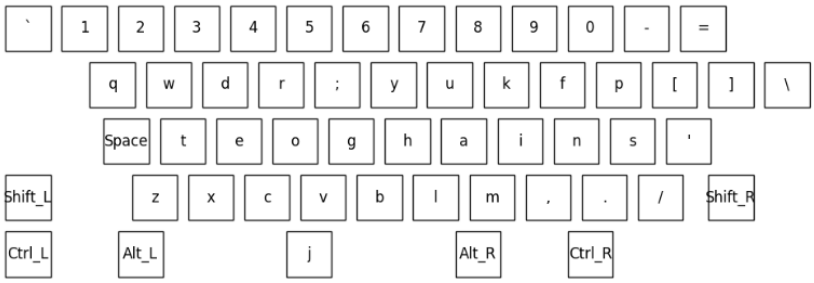

Figure 3: Swapped the home keys with the highly frequent keys ('Space' being the highest, 's' being the lowest of these 8 keys for some string input)

4. **Optimizing the layout with SA**: We give this new layout as our initial solution to SA which then swaps keys (other than the home keys) to different locations to arrive at the final optimized layout.

Figure 4: Optimized layout for the same string
(the string used is the description of this assignment)

5. **Visualizing the SA process:** Using the lists of distances and layouts of each iteration in SA, we create multiple plots which then can be turned into an animation.

For **BONUS**, the same steps can be done for **other keyboard layouts**. Just change the index in layouts list. This is explained in the notebook at the start.

CLICK HERE to see **jupyter notebook** and the **keyboard layout files**. Instructions on how to run it is provided in the notebook.

4 References

[1] How to copy a dictionary and only edit the copy?

[2] How to Create Different Subplot Sizes in Matplotlib?

[3] Placing text boxes

[4] RuntimeError: The init_func must return a sequence of Artist objects.

[5] Wikipedia: Simulated Annealing

[6] ScienceDirect: Evolutionary Algorithms and Metaheuristics