Typo Corrector with Branch and Bound Algorithm

Averrous Saloom - 13520100
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10, Bandung
13520100@std.stei.itb.ac.id

***Abstract*—Typo is common and need to be solved. Writer hypothesized that the problem can be solved with Branch and Bound Algorithm. We can create a cost function for the algorithm from character position in QWERTY keyboard. Then, the cost can be improved with guess play with user.**

Keywords—Typo; Branch and Bound Algorithm; QWERTY Keyboard

I. Introduction

In this digital era, misunderstanding can come from a simple typo. In 2005, US congress have a debate about nuclear test procedure, they brought up a nuclear test with codename 'Sedan' that happened in 1960s as a proof that nuclear test needs comprehensive radioactive waste management. But the congressional transcript made a country in Africa panic, as it is written 'Sudan' instead of 'Sedan'. Creating a diplomatic dispute between two nations (BBC, 2005).

As a programmer, writer typed a lot. Which means writer meets with typo in a daily manner. Writer was helped a lot by multiple visual studio code extension, that give suggestion about the right syntax and correcting the typo occurred.

Writer then notices that this problem can be solved with an algorithm he just learned in Algorithm Strategy class, Branch and Bound Algorithm. The problem can be modeled to a state space tree, and the program will include

The urgency to solve this problem has been high, multiple implementations also had been done. Not undermining the solutions that have existed, writer try to approach this problem with the knowledge he just learned from Algorithm Strategy class, using Branch and Bound Algorithm.

II. Theory

Here, writer hypothesize that this problem can be solved with Branch and Bound algorithm. But what is Branch and Bound Algorithm? How to build one?

Branch and Bound algorithm are algorithm derived from Breadth First Search. Breadth First Search is a technique to access graph. This technique let the program to access every adjacent node first and then move to the next level adjacent nodes. While its counterparts, Depth First Search, ask for depth first. Which means every time the program accesses a node, the next node it will be accessed is the node that is adjacent with that node.

Notice that it derived from a graph search algorithm, which means there should be a graph to be searched. This graph is built with nodes that are called

$$State\ Space$$

These state spaces are possible states the problem can have. Take an example of 15 puzzle. States that are probable are, combinations of the 15 number positions in puzzle. Which will be represented in a matrix.

With state spaces, Branch and Bound Algorithm will solve with this thing with three things

1) Solution set

A solution state is defined as a set of steps the algorithm used to find the goal node, these set of steps are a finite state that are defined. In the fifteen-puzzle problem, the finite state are movements of the empty block can happen

$$Left, Right, Up, Down$$

2) Cost function

III. Problem Analysis

A. Divide the problem

The typo word problem is still hard to solve intuitively. We can't imagine exactly what will be the cost function as an n-length word, will have 26^n possible combination of the real word. There is no specific connection between word and its typo-ness, but each character has it.

A character is a button on the keyboard. Usually, they are arranged in Qwerty arrangement.

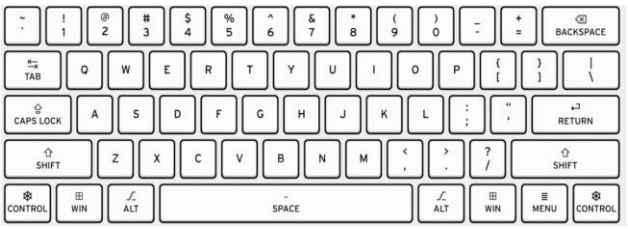


Figure 1 Qwerty arrangement

In Qwerty arrangement, we can see that each character has limited number of adjacent characters around it. Take a character S, the adjacent characters with it are,

$$\{Q, W, E, A, D, Z, X, C\}$$

When we are typing, typo can happen in a condition when,

we press a button of character that is not the same with our intention

Notice the word intention here, we intent to be right, so our finger is intended to press the correct button. But a thing happened so that we press the wrong one. This 'thing' is not intentional, means that the distance of our pressed button with the correct button shouldn't be far. It should be between those adjacent character.

Each character always has adjacent characters around them. These adjacent characters can be identified as "the best possible alternative" of a character typo. Hence, this relation can be the cost, if a character is close to a typo character, it is more probable that the character is the correct alternative.

With that behavior, character typo is easier to solve. Thus, we will focus on solving a character typo problem and develop it to the word typo problem.

B. *Cost analysis*

When a character typo occurs, there are possibilities of fault that can happen. Let's consider character S typo,

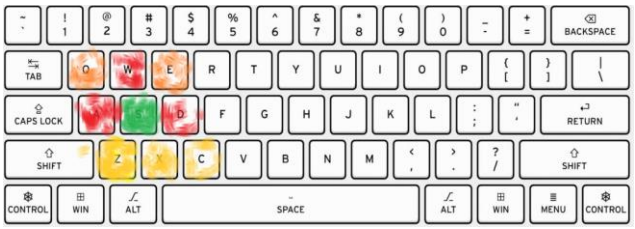


Figure 2 Writer's interpretation of possible fault

When a typo occurs, our finger presses the wrong button. What is the possibility of each adjacent character pressed?

To answer the question, we must understand how our fingers naturally move on the keyboard. In many typing techniques, we place our fingers on the leftmost and rightmost position of the second layer of the abjad characters arrangement.

Figure 3 Default fingers position

With this position, when we want to reach set of characters that is in the lowest layer of the arrangement, we need to stretch our finger inwards. And when we want to reach set of characters in the top layer, we need to stretch it outwards.

Notice that, our finger tends to stretch outward. When we let our finger uncontrolled, it will stretch outward. Hence,

Possibility of a adjacent character that is in the direct row above the target character has higher possibility to be the typo character

So, in character S case, the one who has the least cost is W as it is positioned directly above character S.

What about character Q and character E, it is adjacent and positioned in row above the target?

In ten finger typing principle, our fingers are arranged in such a way that to press a button it is assigned that a specific finger will press those. Let's list them by grouping them in finger,

1) *Left pinkie finger*

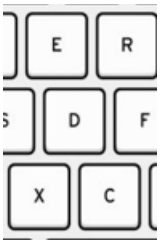
Characters that are pressed by left pinkie finger are:

Q	A	Z

2) *Left ring finger*

W	S	X

3) Left middle finger



E	D	C

4) Left index finger

R	F	V
T	G	B

5) Left thumb finger

SPACE

6) Right thumb finger

SPACE

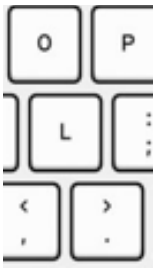
7) Right index finger

Y	H	N
U	J	M

8) Right middle finger

I	K	,

9) Right ring finger



O	L	.

10) Right pinkie finger

P	;	/

With this rule in mind, we can simplify the cost system for the BnB algorithm. But first, we must agree that this rule is so common that almost every typist use this rule. Because these habits are common, people will tend to make mistake that bound with the habits.

It is more probable to make typo to a character that is pressed with the same finger

Now we have three things that will affect the cost of the branching.

1. *Adjacency*
2. *Its row position*
3. *Its sister character (Character with same finger)*

C. Guessing

The program doesn't have a dictionary of words. Which means it don't really create a single state with that

IV. Branch and Bound Algorithm

As writer defined in part III, branch and bound algorithm is an algorithm that is used to scan to find that goal state and scan the possible states with cost approach.

V. Conclusion

A typo fault corrector can be implemented in branch and bound algorithm.

Acknowledgment

Thanks to our lecturers and teaching assistants who work hard to create the best course for us.

References

Munir, R. 2020, *Slide: Algoritma Branch and Bound*, ITB, Bandung

STATEMENT

Hereby I state that the article that I wrote is my own, not a copy nor translate from other person's article, and not a plagiarism.

Bandung, 20 Mei 2022

Averrous Saloom 13520100