Similarities between two C programs

Ivani Patel

October 8, 2022

Abstract

In basic terms, an algorithm is a set of well-defined steps or rules that you need to follow to obtain a pre-determined result. It is a strategy utilized for tackling an issue or playing out a calculation. When we talk about algorithms in computer programming, an algorithm would be all the defined steps to follow on the input to get the desired output. Apart from mathematics or computer programming, in real world it becomes hard to figure out the issue, so it requires significantly greater investment to develop an algorithm. From this in this paper we'll see a method for tracking down the similarities between two C programs.

1 Introduction

At the point when the word algorithm is tossed around it invokes pictures of tech new businesses and binary code. What a significant number of us don't understand is that algorithms are surprisingly basic. Without a doubt, they are utilized by the tech wizards of the world to fabricate programming, yet they are likewise utilized in regular daily existence. From an early age, we are taught how to complete day to day tasks. We figure out how to count, sort, walk, tie our shoes, and so on. Practically these cycles are instructed to us with a step-by-step procedure. Basically, we are learning algorithmic reasoning. To separate it, algorithms are a sequence of tasks done in response to an initial situation.We should limit the methodology and focus on a particular space which is innovation. The ongoing term of decision for a critical thinking methodology, algorithm, is ordinarily involved these days for the arrangement of rules a machine (and particularly a PC) follows to accomplish a specific objective. Throughout the long-term innovation has been developing quickly. Also, because of additional assets, it expands the trouble level to tackle the issue. In this paper, we depict a technique for looking at two C programs.

To plan an assessment to track down similarities between two C programs, without any hesitation we genuinely need to figure out that: Resemblance can be in various ways, Program's feedback and result or thinking of program or code written in the same style. Taking into account as far as possible, assessment changes. Tracking down various methods of finding the similarity between two C programs there can be two different ways Assuming the source code is given the C programs are white boxes and we can without much of a stretch look at the source code. Be that as it may, on the off chance that we don't have the foggiest idea about the source code then C programs are black boxes. We'll consider C programs as black boxes, and we need to find what are the different ways one can determine that both the black boxes are similar. While utilizing the black box strategy one issue happens, we don't have any idea what is inside the black boxes. Contrasting black boxes resembles looking at cat and fish, looking at television and coffeemaker. As we know nothing about things in black boxes it becomes hard to comprehend the issue and the issue which used to appear to be effectively resolvable now appears to be challenging to try and comprehend.

At the point when the issue is hard to comprehend, we concentrate on the issue and attempt to comprehend what it is attempting to convey. So, in the wake of concentrating on the issue, in this paper, we'll discuss the methodology that can be utilized to foster an algorithm to decide likenesses between two black boxes.

2 Methodology

Before beginning ahead, we want to determine the term similarity: in this paper Similarity between two C projects, the significance of the term comparable by its temperament ends up being uncertain. Programs have two factors which are function and the process of implementation. So, the similarity between the two programs should be discussed at the function level and the implementation level. Also, similarity should be discussed within a given range of inputs (the range could be anything). The function of the program also means the output of the program. We can characterize it with regards to similarity, or a few changes in the capability or its highlights in the way in which two projects are comparable.

Reviewing writing where the focal point of the topic is the black box approach where we don't know about the projects or the articles for which we are attempting to find the similarities. The main idea of the paper "Statistical Similarity of Binaries" is to decay the code into brief pieces and also, in some way or another track down similarities between those pieces. It talks about the way to deal with tracking down likenesses between two procedures through their stripped parallels despite the fact that when gathered with various compilers.

Imagine a scenario in which we broaden our scope to discover some summed-up approach where any two black box objects e.g. cat, fish, coffeemaker, or TV can be utilized, and a definite similarity can be determined.

Let's consider input-output occurring in a sequence and observe all the input-output occasions. At the point when this succession of input-output events are happening, we note down the timings of every single data and their related yield using a microprocessor. At the point when we cultivate the timing frame from the gathering of data yield then we use FFT technique where we endeavor to track down likenesses among signals and afterward utilizing that data, we choose if the two things are practically identical or not.

Sequence of events are the order in which things happen, the order in which a series of events occur at a unique time sequence. Let's consider a Telephone system after series of inputs like lifting receiver, dialing number, hanging up call, etc. and outputs like dial tone begins, phone ringing, ending call, etc. A state diagram or any flow chart can depict a good sequence of events which basically shows the flow of events and there might be temporary or even permanent (end) results.

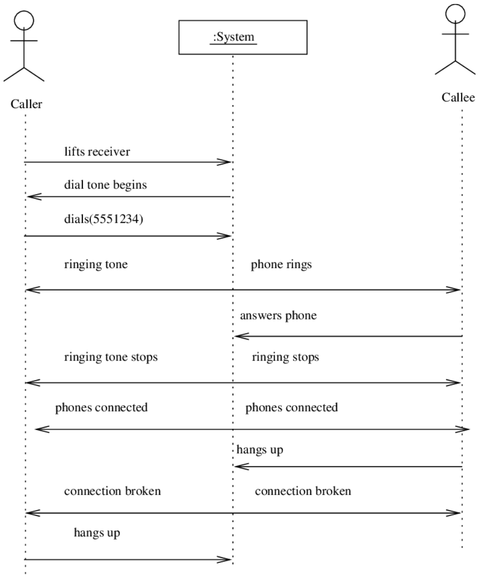

Figure 1: Telephone System Sequence of events

Presently utilizing chip 8085 we make timing chart for that we really want to change over the arrangement of occasions into time series signal. So, we can involve it in timing outline. Fundamental point of timing graph is to focus on timing limitations. Timing charts are utilized to investigate the ways of behaving of items all through a given period. A timing graph is a unique type of a grouping outline. Microchip 8085 utilize different from of events as information and creates waveform of events which is utilized for additional interaction.

Fast Fourier transform (FFT) is one of the most useful tools and is widely used in the signal processing. A fast Fourier transform (FFT) is an algorithm that computes the discrete Fourier transform (DFT) of a sequence. Fourier analysis converts a signal from its original domain (often time or space) to a representation in the frequency domain. The DFT is obtained by decomposing a sequence of values into components of different frequencies. An FFT rapidly computes such transformations by factorizing the DFT matrix into a product of sparse (mostly zero) factors. As a result, it manages to reduce the complexity of computing the DFT from O(N^2), which arises if one simply applies the definition of DFT, to O(N log N), where N is the data size.

The main idea of FFT algorithms is to decompose an N-point DFT into transformations of smaller length. Let's see how we can reformulate an N-point DFT in terms of DFTs of smaller length. To have a better insight into the algorithm, we will see the procedure through examining an eight-point DFT as an example.

Assume that $x^{(n)}$ is a sequence of length eight. we can find the eight-point DFT of $x^{(n)}$ as

$$X(k) = x(0)e^{\frac{-j2\pi k*0}{8}} + x(1)e^{\frac{-j2\pi k*1}{8}} + x(2)e^{\frac{-j2\pi k*2}{8}} + ... + x(7)e^{\frac{-j2\pi k*7}{8}}$$

Our goal is to examine the possibility of rewriting this eight-point DFT concerning two DFTs of more modest length. We see that the length of the DFT, N = 8 for this situation, shows up as the denominator in the perplexing exponentials. This leads us to the idea that what if we choose and group some terms of Equation in a way that allows us to simplify the fractions $e^{\frac{-j2\pi kn}{8}}$.In this way, we may extract a DFT of smaller length out of Equation. For example, since in our case N is an even number, we examine choosing all the terms with an even sample index, i.e. x(0), x(2),x(4),and x(6). After simplifying the fractions in the complex exponentials, we obtain

$$G(k) = x(0)e^{\frac{-j2\pi k*0}{8}} + x(2)e^{\frac{-j2\pi k*1}{8}} + x(4)e^{\frac{-j2\pi k*2}{8}} + x(6)e^{\frac{-j2\pi k*3}{8}}$$

We have shown that half of the computations in Equation can be replaced with a four-point DFT. But the remaining terms, which correspond to odd-index samples, are given by

$$H1(k) = x(1)e^{\frac{-j2\pi k*1}{8}} + x(3)e^{\frac{-j2\pi k*3}{8}} + x(5)e^{\frac{-j2\pi k*5}{8}} + x(7)e^{\frac{-j2\pi k*7}{8}}$$

To simplify the fractions, we can simply factor $e^{\frac{-j2\pi kn}{8}}$ and obtain

$$H1(k) = e^{\frac{-j2\pi k*1}{8}}(x(1)e^{\frac{-j2\pi k*1}{8}} + x(3)e^{\frac{-j2\pi k*2}{8}} + x(5)e^{\frac{-j2\pi k*3}{8}} + x(7)e^{\frac{-j2\pi k*4}{8}})$$

we observe that H1(k) is obtained by multiplying $e^{\frac{-j2\pi kn}{8}}$ by the four-point DFT of x(1), x(3), x(5), and x(7). Hence, we have achieved the goal of decomposing an eight-point DFT into two four-point ones.Defining

$$H(k) = x(1)e^{\frac{-j2\pi k*1}{8}} + x(3)e^{\frac{-j2\pi k*2}{8}} + x(5)e^{\frac{-j2\pi k*3}{8}} + x(7)e^{\frac{-j2\pi k*4}{8}}$$

We get the final result as

$$X(k) = G(k) + e^{\frac{-j2\pi k*1}{8}}H(k)$$

Now calculating X(k+4) we get

$$X(k+4) = G(k+4) + e^{\frac{-j2\pi(k+4)*1}{8}}H(k+4)$$

Since G(k) and H(k) are periodic

$$X(k+4) = G(k) + e^{\frac{-j2\pi(k+4)*1}{8}}H(k)$$

These given equations clarifies how breaking an eight-point DFT into two four-point DFTs allows us to use almost the same computations for both X(k) and X(k+4) and significantly decrease the number of calculations through the periodicity property.

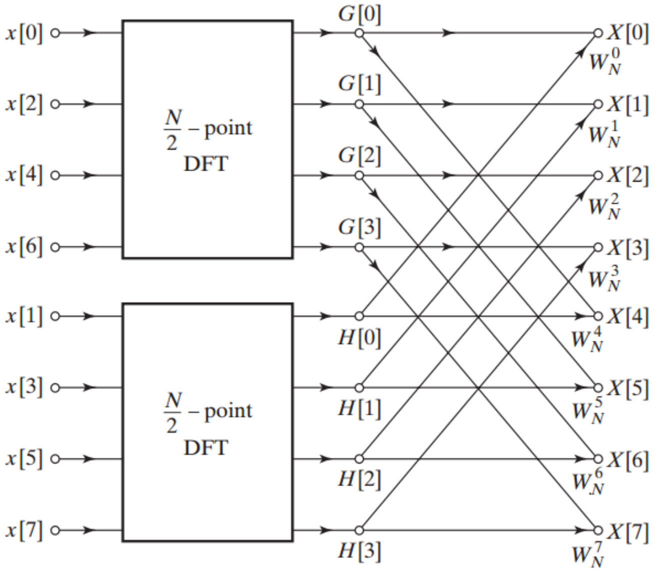

Figure 2: The flow graph of breaking an eight-point DFT into two four-point ones

We can express the gains in terms of Big O Notation as follows. The initial term comes from the way that we register the Discrete Fourier Transform two times. We increase the last option when taken to register the Discrete Fourier Change on a portion of the first info. In the last step, it makes N moves to include the Fourier Transform for a specific k. We account for this by adding N to the end result.

$$2(\frac{N}{2})^2 + N$$

$$= 2\frac{N^2}{4} + N$$

$$= \frac{N}{2}/2 + N$$

$$O(\frac{N^2}{2} + N) \sim O(N^2)$$

We can additionally work on the calculation by applying the divide-and-conquer approach, dividing the computational expense each time. In other words, we can keep on parting the issue size until we're left with gatherings of two and afterward directly compute the Discrete Fourier Transforms for each of those pairs.

Normalized Cross-Correlation (NCC) is by definition the inverse Fourier transform of the convolution of the Fourier transform of two signals. It can be measured by:

$$NCC = cos\theta = \frac{a.b}{|a||b|} = \frac{\sum_i a_i b_i}{\sqrt{\sum_i a_i^2 \sum_i b_i^2}}, -1 \leq NCC \leq 1$$

3 Algorithm

We analyzed procedure to handle the issue now we'll see the algorithm for the same:

- Note down the sequence of inputs/outputs events.Also record the time of events at which it has happened.
- Utilizing 8085 microprocessor plot the succession of input-output which will be in type of timing diagram. Microprocessor will change over the events into the type of signals.
- At the point when the signals are formed then utilizing the normalized cross-correlation by the Fast Fourier Transform method we can find how comparative the signals are.
- At the end we get last result which will show some number which will be the comparability index between the two black box objects.

4 Conclusion

In this, we have proposed a unique procedure for program relationship. The proposed system uses the black box approach and communicates a greater view game plan and an algorithm where on the off chance that two items/programs are completely unknown as a black box we can use the given technique and algorithm to find similarity index.