Rutgers University, New Brunswick

Data Structures and Algorithms - 16:332:573

Plagiarism Detection - Rabin Karp Algorithm

Pranav Shivkumar

194007405

04/07/2020

Table of Contents

Abstract

Plagiarism, in simple terms, is the act of taking someone else's work and passing it off as one's own work. This is a prevalent menace in the scientific world, especially in the research domain. Several technologies have been, and are being, invented to prevent any plagiarized work from being passed into the scientific community. These technologies are devised with the help of certain algorithms, particularly string searching algorithms.

This paper describes the basic principles behind the Rabin-Karp algorithm, also called as the Karp-Rabin algorithm. This algorithm is basically a string searching algorithm which detects patterns in words or text. This algorithm, when used in conjunction with other algorithms, can provide us with a competent plagiarism detector. This paper also focuses on the impact and applications of the Rabin Karp algorithm, particularly as a plagiarism detector.

Introduction

Examinations are one of the most fundamental ideas which are used to evaluate an individual, which are of various types, like academic, psychological, aptitude, to name a few. They also help a stranger understand the depth of an individual's understanding of a particular idea or concept. Due to the level of importance of the examinations and tests conducted, certain individuals resort to copying answers off other people. In the world we live in today, cheating is prevalent in the world of examinations and tests. Individuals copy answers to some questions and pass them off on their own, which provides a misleading idea as to how much the individual actually knows. With the migration of examinations and tests to online media, like computers, cheating is increasing day after day due to the ease with which it can be accomplished. While several methods have been implemented to prevent this, it has proven to not be enough.

In the world of science and research, this concept analogizes to plagiarism, which as explained mentioned before, is the act of stealing someone else's work and passing it as one's own, borne out of one's own efforts. For example, let's say a scientist, A, publishes a paper in an accredited journal and furnishes results based on the experiments he/she has conducted. Another scientist, B, publishes another paper on another concept. Now another scientist, C, comes across these papers and tries to publish a paper by copying particular sections of both the papers verbatim, and copies other papers as well. This is an example of a plagiarized paper which must be detected as plagiarized. However, plagiarism doesn't exist only in research; it also exists in everyday life, like when a website copies multiple other websites, or even in schools, when a student is asked to write a report and copies off a book or an internet source word for word.

This paper discusses the Rabin Karp algorithm, which is a string searching algorithm and can be used to help detect plagiarism.

String Searching Algorithms

A string searching algorithm is an algorithm that utilizes strings as the underlying data type, deployed in the form of arrays as the data structure. The basic functionality of a string searching algorithm, or a string-matching algorithm, is that it tries to find if one or more several strings, or text, is found in a larger string. The string that is being searched for is usually called the substring or a pattern. The efficiency of the string searching algorithm is affected by the type of encoding of the string; in particular, if variable length encoding (an encoding scheme where codes of varying length are used to encode the string) is utilized, the time taken to reach the Nth character of the string will be longer, proportional to N. There are various implementations of string searching algorithms, such as the Naïve String Search algorithm, Knuth-Morris-Pratt algorithm, Boyer-Moore algorithm and the Rabin Karp algorithm.

Overview of Rabin Karp Algorithm

The Rabin Karp algorithm is a type of string searching algorithm which can be used to search for several strings. It was developed by Richard M. Karp and Michael O. Rabin and utilizes hashing to check if a string exactly matches a pattern in the text. The purpose of the hash is so that the positions of the text that do not match the pattern are filtered out, reducing the number of positions to check for the pattern.

The expected time taken for a single search is linear, or O(N) in the average case, where the length of the pattern and text are the same. However, the expected time taken for the worst case is the product of the lengths of the pattern and the text.

Working of the Rabin Karp Algorithm

The Rabin Karp algorithm speeds up the performance compared to the naïve string searching algorithm. since the naïve string searching algorithm iterates throughout the entire string to search for a particular pattern, which takes linear time, or O(N), or in the worst case O(MN), where N is the length of the pattern and M is the length of the text. In contrast, since the Rabin Karp algorithm uses a rolling hash function which eliminates the positions where the pattern doesn't occur, thereby reducing the time taken.

In general, a hash function is a function that maps data of arbitrary values and size to fixed size data. The Rabin Karp algorithm makes use of a rolling hash, which is a hash function where the input is passed through a window of some size which helps to map the data. The rolling hash

function makes use of the previous hash values in order to compute the hash values for the succeeding input values. The old value is removed from the window and is replaced by the new value.

The rolling hash function used in the Rabin Karp algorithm is the Rabin Fingerprint. In this scheme, the n-bit message is converted into a polynomial of degree n-1 over a finite field GF(2) and a random irreducible polynomial is selected, say p(x) of degree k, over the same field. The remainder obtained on dividing the message polynomial by the k-degree polynomial is said to be the Rabin fingerprint.

The pseudo code of the Rabin Karp algorithm is as follows:

```
def RabinKarp(s[1…n], str[1…m):
    //hashing the pattern of length m
    hstr = hash(str[1…m])
        for i = 1 to n-m-1:
            hs = hash(s[1…i+m-1])
            if hs == hstr:
                if s[1…i+m-1] == str[1…m]:
                    return i
        return not found
```

The above pseudo code describes the working of the Rabin Karp algorithm. The function takes two arguments – the pattern (str) and the entire text (s), both as strings. The first line uses the hash function to hash the pattern (of length m) using the Rabin fingerprint or any other hash function that produces a similar effect. The next step is to hash the text using a window of size m + 1, which is the size of the hashed pattern. Since the algorithm makes use of a rolling hash function in order to compute the hash, the hash of the next character is computed quickly. Now if both the hashed pattern and the hashed text in the window are equal, then the function returns that the pattern is found in the text along with its position in the text; otherwise, the system returns that it is not found.

Analysis and Complexity

From the pseudo code in the previous section, we see that hashing the pattern has a time complexity of O(m), since it has to iterate throughout the length of the pattern (m).

Now, within the for loop, from index i to n-m+1, which creates a window of size m+1 throughout the length of the text (n), there are 2 cases, which depends on the manner in which the hash function is computed.

If the hash is computed for the pattern s[i+1….i + m], then the hash computation is said to be naïve. The time complexity is, thus, O(m). Since this applies for a single iteration of the for loop, the entire naïve hash computation takes a time complexity of

$$O(n - m + 1) \times O(m)$$

or,

$$O(nm)$$

This is said to be the worst-case scenario of the Rabin Karp algorithm. This complexity is the same as other straightforward string-matching algorithms, like the naïve string-matching algorithm.

However, if the hash is computed in constant time, then the speed and hence the overall complexity would reduce. Since the variable hs already holds the hash of the pattern or substring, it can be used to compute the hash of the succeeding values of the text. This method comes into play with the help of the rolling hash function. In this case, the complexity will be O(1), since the successive hash values are computed using the previous hash values in the rolling hash function, for a single iteration of the for loop. For the entire run-time of the for loop, the complexity will be

$$O(m) + O(n) \times O(1) = O(m + n)$$

Since,

$$m < n$$

$$m + n < 2n \ (m + n \ will \ be \ bounded \ by \ 2n)$$

As a result of this,

$$O(m + n) \approx O(2n) \approx = O(n)$$

This said to be the best-case scenario for the Rabin Karp algorithm.

The average case remains the same as the best case, since the rolling hash calculation takes a time complexity of O(n) and once the algorithm finds a potential match, it should verify each letter of the pattern to make sure that it is a perfect match, which takes m steps. Thus, the average case for the Rabin Karp algorithm is:

$$O(m+n), or\ O(n)$$

And the number of comparisons for the algorithm is,

$$O(m+n)$$

The results are summarized in the table below:

Case	Complexity
Best Case	O(m + n) or O(n)
Worst Case	O(nm)
Average Case	O(m + n) or O(n)

Applications

The Rabin Karp algorithm finds extensive applications in intrusion detection, bioinformatics and string searching, particularly in string matching. As such, it is used in plagiarism detection, in conjunction with other algorithms like the Winnowing algorithm.

Winnowing Algorithm

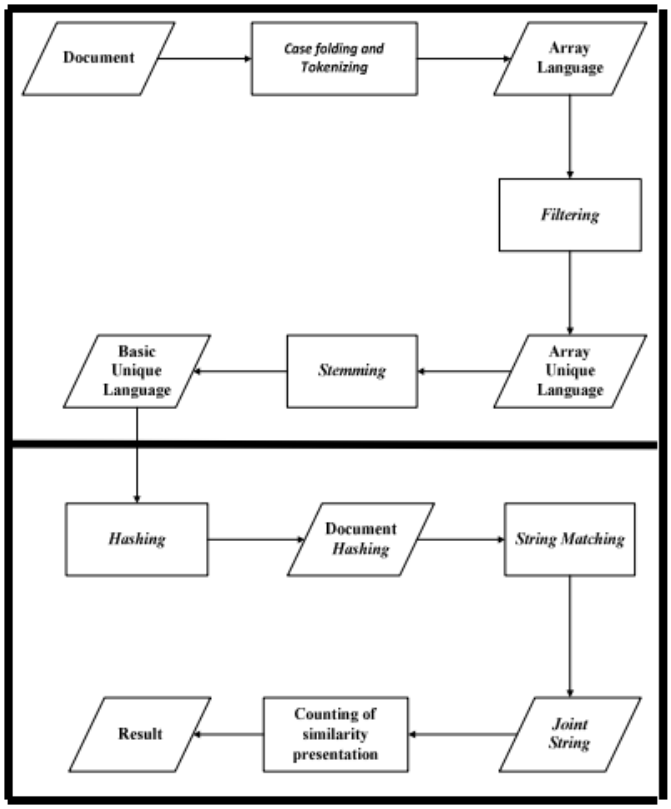
The winnowing algorithm is basically used to detect the similarity between documents or portion of texts. However, in the context of plagiarism detection, it is used to compute the hash values for the documents being checked for plagiarism, using the rolling hash function. These hash values are then used to form a window, in which a minimum hash value is selected, and all the selected hash values are stored as the fingerprints of the document. In the case of multiple minimum hash values, the rightmost hash value is selected as the minimum hash value.

Plagiarism Detection

An implementation of the Rabin Karp algorithm and the Winnowing algorithm can be used to design a plagiarism detector. In this implementation, the input to the detector is two documents – the original document and the document to be tested for plagiarism. The system first checks the documents to obtain information like the number of sentences, words, paragraphs and so on. Once this information is obtained, a number of processes take place, like tokenizing (removing sensitive data from the system and encrypting it), filtering (removing words and other punctuation that are of less importance) and stemming (breaking words down into their basic form). Once these preprocessing steps are executed successfully, the data is sent to the

Winnowing algorithm block, where the hash is calculated using the rolling hash function. Finally, the hashed values are sent to the Rabin Karp algorithm block, where the similarities in both documents can be determined. In this way, plagiarism can be easily detected.

The flowchart for the implementation is shown below:



Flow Chart for Plagiarism Detection using the Rabin Karp and Winnowing algorithms [1]

Conclusion

This paper focused on the working and overview of the Rabin Karp algorithm, along with an insight to the time complexity for the best case, worst case and average case scenarios of the algorithm. Finally, a practical application of the algorithm is in plagiarism detection, which can be used with algorithms that effectively compute the hash function, like the Winnowing algorithm in this case.

References

1. Dedi Leman, Maulia Rahman, Frans Ikorasaki, Bob Subhan Riza, Muhammad Barkah Akbbar – Rabin Karp and Winnowing Algorithm for Statistics of Text Document Plagiarism Detection
2. M. Misbah Musthofa, Ainul Yaqin – Implementation of Rabin Karp Algorithm for Essay Writing Test System on Organization xyz - 2019 International Conference on Information and Communications Technology (ICOIACT)
3. https://en.wikipedia.org/wiki/Rabin%E2%80%93Karp_algorithm
4. https://en.wikipedia.org/wiki/Rabin_fingerprint
5. https://en.wikipedia.org/wiki/String-searching_algorithm