Available online at http://www.journalcra.com

International Journal of Current Research
Vol. 9, Issue, 09, pp.57572-57574, September, 2017

INTERNATIONAL JOURNAL OF CURRENT RESEARCH

ISSN: 0975-833X

RESEARCH ARTICLE

ADVANCE-RABIN KARP ALGORITHM FOR STRING MATCHING

***Er. Mohammad Shabaz and Er. Neha Kumari**

Department of Computer Engineering, Chandigarh University, Gharuan, Punjab

ARTICLE INFO

Article History:
Received 14th June, 2017
Received in revised form
25th July, 2017
Accepted 22nd August, 2017
Published online 29th September, 2017

Key words:
Gynecologist, Perinatal Oral Health Care, Dental Home, Anticipatory Guidance, Dental caries.

ABSTRACT

In the vast field of computer science, carries lots of work with that having lots of problem with it. one of the traditional problem is string matching that looks very simple but internally major issue in the speedy world. The string matching problem operates usually on the text (0….n-1) and the pattern (0..m-1) with particular matching scenario. The string matching starts from the very left or right of the text and checks for occurrence of pattern. If pattern not found, pattern will move one shift towards the right. In this scenario, it works recursively till reach towards the right or left end of text. For handling problem of string matching there are number of approaches, one of them called Rabin karp but working of rabin karp have the dark-face. In the other word's indication towards, dealing with integers or alphabets taking more execution time. So in this paper, the designed approach deals with a String which is purely alphabetic and hybrid using Advance-Rabin karp taking less execution time than Rabin-Karp

Citation: Er. Mohammad Shabaz and Er. Neha Kumari, 2017. "Advance-rabin karp algorithm for string Matching", *International Journal of Current Research*, 9, (09), 57572-57574.

INTRODUCTION

The string matching is occurrence of substring of a string. In other words, string called text T={t_1,t_2……t_n} and substring to be matched called pattern P= {p_1,$p_{2....}p_m$}. Here the thing to noted that the length of text and pattern must be m<= n. The concept of matching string is being used in the real world as query matching in database schema, network system etc. Basically, for occurrence of string being match either by two techniques one of them is exact matching, another is approximate matching.

Needleman Wunsch Algorithm: This algorithm makes exact matches of two strings together in specific sequence. The main application of this algorithm in bioinformatics, in case to align the protein and nucleotide sequences.

Smith Waterman Algorithm: this algorithm, for determine the similarity of in particular portion in between the two possible patterns as in nucleotide or protein sequences. Its main focused on optimal similarity measurement

Dynamic Programming Algorithm: This algorithm typically solves the optimization problem with possible solution with minimum and maximum value.

Corresponding author:* **Dr. Neha Kumari
Department of Computer Engineering, Chandigarh University, Gharuan, Punjab

This algorithm uses most likely to divide and conquer approach. The following are the main steps undertaken by dynamic programming to get optimal solution:

1.Optimal substructure
2. recursive process
3. value of problem's and their optimized solution
4. construction

Brute Force Algorithm: Brute Force algorithm is one of the simplest algorithms for string matching. Where the comparison has been make in text T= {t_0,t_1…..t_{n-1}}with pattern P={p_0,$p_{1.....}p_{m-1}$}.there are two possible cases of matching the text and pattern which is either match found or not otherwise shift one index from left to right. The main advantage of this algorithm is ease to implement but have slower in comparison.

average case at running time is $O(n + m)$
worst case time is $O(n\,m)$

Fuzzy string Algorithm: Fuzzy string algorithm is an algorithm for approximately determine the text and pattern to be matched. The approach used by this algorithm is in two cases , firstly, search of substring is done which is not exactly within text. Secondly, match it with dictionary which is also not exact pattern.

Rabin Karp Algorithm: Rabin Karp Algorithm for string matching was introduced in 1987 by M. Karp and Michael O.

Rabin. With the use of hashing technique make this algorithm differ other algorithm for efficiently matching the pattern of length m from text n.

Average and Best case at running time is O (*n+m*)
worst-case time is O (*nm*)

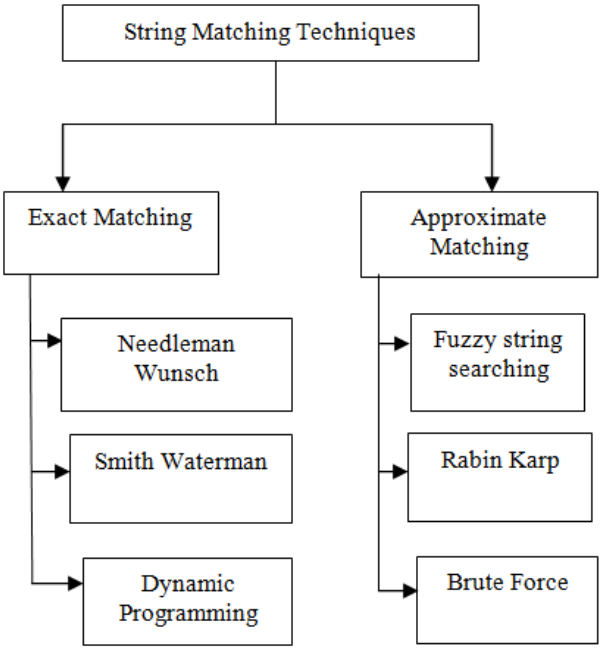

Figure 1. String Matching Technique

LITERATURE REVIEW

(Richard M. Karp and Michael O. Rabin (1987): In this paper the authors introduced the randomized algorithm for fingerprinting. The proposed approach which efficiently pattern matching applicable on irregular shapes by means of one dimensional or multiple dimensional. The technique used based on three randomized algorithm namely algorithm 1,algorithm 2, algorithm3 which have their own features to detect the matching based on parameters of fingerprinting, probability of error, run time.

In various cases	Time Complexity
While constructing tree T	O(q|T|)
Redundancy pruning(worst case)	$O(N_T^2)$
Suffix links	$O(N_T q)$
Matching algorithm using tree T	O(q|Q|)
Matching algorithm using Pruned tree T_P(worst case)	O(q|Q|)

Vibha Gupta et-al 2014: In this paper the author and team highlight the problem of Intrusion Detection system (IDS) in network security. The methodology used named as Signature base IDS, in which signature is used as pattern matching .

For example:
Text: Buffer overflow attack is performed
Pattern: attack

The applied approach is not only a pattern matching but also detect attack with comparative analysis for exact string and pattern matching algorithms, this is done on the behalf of following parameters as in Table. After the comparative analysis the author found that the Rabin Karp is efficient at run time due to hashing approach.

(Cynthia Kustanto et al (2009)). In this paper, the author, worked on automatic plagiarism detection, in which the rate of plagiarism will found. The tool named as Deimos is used, which based on the real condition of programming language. The approach implemented, based four parameters, firstly define the token set, then implement a new scanner and parser and register the name of the language to the interface. At last, put a call statement in the source code.

(Gaston H. et al (2012)). This paper, the author proposes the String can be converted into integer by creating tokens to overcome the identified problem Modulus of string can be obtained after converting it into integers and also discuss the Complexity can reduced to be (Expected value of O(n-m+1)).

(*Sonawane et al. (2015*)).In this paper the authors identifies the problem of Replica (imitate other's work and pretend it as own), Fusion (creation of duplicate work without add its reference) and Paraphrasing (slightly, make different from actual text without changed whole statement).

Parameters	Algorithms			
	Brute-Force	Rabin-Karp	KMP	Boyer-Moore
Pre-Processing	No	O(m)	O(m)	O(m+|∑|)
Search Type	L -> R	L -> R	L-> R	R -> L
Running Time	O(nm)	O(n+m)	O(n+m)	O(n/m)
Approach	Linear search	Hashing based	Heuristic based	Heuristic based
Search Idea	Search by matching all character	Compares hash values of the text and the pattern	Constructs an automaton from the pattern	Bad –character and good suffix heuristic to find valid shift

Prohald fogla et-al, April (2006): In this paper, author et-al, focused on string matching problem. The author used the q-gram algorithm with tree model, here q-gram focused on query matching and pruning algorithm helps in removing the redundant tree. This approach efficiently manages time and space complexity.

The idea behind this approach better match with rabin karp instead of storing in hash table. Also perform experiment on system call sequence, Bernoulli and markovian data as shown following figure:

And with respect to problem that formulated the solution such as Divide the real work into segments by using the parameters that conquer the problem into sub problem and use it as solution as in form of token. Similarly for the second identified problem named as Fusion with the proposed solution of Convert the original work into another format by casting and also using the Lempel Ziv algorithm Compresses tokens also bridge the gape of paraphrasing by using approach of plagiarism by using Kolmogorov complexity formulas tokens so that the original work as a whole would be changed.

(Lukas Hrbek et- al. (2016)). In this paper, author focused on approximate string matching with filtering algorithm, which classified into three categories namely portioning into exact search, neighborhood generation Intermediate Partitioning .Also discuss the modules to achieve its compressed form. This module discuss the namely phases generating neighbor, exact string match, verification of potential occurrence, global alignment with pattern comparison with BLAST.

(Nazimuddin sheikh et-al. (2016)). In this paper the focused on exact string matching algorithm for intrusion system. The comparison analysis done in following phases

1.Processing Phase
2.Search phase
3.Computational Phase
4.Matching Phase
Rabin-Karp algorithm is the best for time complexity of Boyer-Moore algorithm is quite good.

Results on Comparing Rabin Karp with Advance Rabin-karp using different string lengths

Elapsed Time in seconds. With size (Memory) = 64kb

String length	Pattern length	modulus	Advance R-K	Rabin-Karp
76	4	10	0.101145	0.101233
113	4	10	0.100576	0.101154
300	6	10	0.101154	0.101400
450	4	10	0.101293	0.101521
600	6	1	0.101284	0.101356
900	6	1	0.101243	0.101488
1300	8	1	0.101208	0.101373
2500	8	1	0.101590	0.102760

As we proceed further we found better results over large string.

Therefore the Worst-case time is O (*m(n-m+1)*).

Conclusion

The algorithms that we used day today, must require to work efficiently. This efficiently only comes with the focused on the reducing complexity and execution time.

The Advance Rabin karp algorithm upgrades the working of the Rabin karp algorithm by reducing its execution time and thus provides a better result in less time.

REFERENCES

Altschul, Stephen F., Gish, Warren, Miller, Webb, Myers, Eugene W. and Lipman, David J. 1990. Basic Local Alignment Search Tool. *Journal of Molecular Biology* 215(3), pp. 403-410.

Daly, C. and Horgan, J. 2005. "Patterns of plagiarism," in Proc. 36th SIGCSE Tech. Symp. Computer Science Education, New York, pp. 383–387.

C. Jason Coit, Stuart Staniford and Joseph McAlerney, 'Towards Faster String Matching for Intrusion Detection or Exceeding the Speed of Snort', 1-7.

http://en.wikipedia.org/wiki/Knuth%E2%80%93Morris%E2%80%93Pratt_algorithm.

http://www.dcs.gla.ac.uk/~pat/52233/slides/Strings1x1.pdf.

Hussain I., Kausar S., Hussain L., and Asifkhan M., Improved Approach for Exact Pattern Matching, International Journal of Computer Science Issues, Vol.10, Issue 3, No.1, 2013.

Karp, Richard M.; Rabin, Michael O. (March 1987). "Efficient randomized pattern-matching algorithms". IBM Journal of Research and Development. doi:10.1147/rd.312.0249

Ramazan S. Aygün "structural-to-syntactic matching similar documents", Journal Knowledge and Information Systems archive, Volume 16 Issue 3, August 2008.

Richard S. Bird, 'Polymorphic String Matching', 110-115, Haskell'05 2005, Tallin, Estonia.

Seema Kolkur , Madhavi Naik (Samant) "Program plagiarism detection using data dependency matrix method" , in proceedings of International Conference on Computer Applications 2010 , Pondicherry , India December 24-27, 2010, pp 215-220.

Yang T., Zhang M., A Quick String Matching Employing Mixing Up, International Journal of Hybrid Information Technology, Vol.6, No.4, 2013.

Yoan Pinzon, "Algorithm for approximate string Matching",dis.unal.edu.co/~fgonza/courses/2006.../ approx_string_matching.pdf August 2006.

Yu-lung Lo Chien-Chi Huang, 'Fault Tolerant Music Retrieval by similar String Matching', National Science Council of ROC Grant NSC98-2221-E-324-027,1-10.

****