VOL. 11, NO. 9, MAY 2016 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

A NEW MULTI PROCESSOR PARALLEL STRING MATCHING WITH OMEGA MODEL

K. ButchiRaju
Department of Computer Science Engineering, GRIET, Telangana, India
E-Mail:butchiraju.katari@gmail.com

ABSTRACT

In this paper, parallel string matching with omega computing model is proposed. This algorithm especially designed in a way that it works in omega computing model where text is split into number of chunks. The numbers of chunks depend on the target number of processors. The chunks and patterns are assigned to processors in the omega model; later the processors perform the search operation and return results. The proposed process improves performance and efficiency.

Keywords: string matching, multi-processor, omega model, parallel string matching.

1. INTRODUCTION

Over the last 30 years, string matching is an extensively studied problem because of its wide applications. The applications include Information Retrieval Systems (IRS), Pattern Recognition, Text editors, Text Search etc. The advantages of this string matching are - it results in accurate search, improves efficiency, reduces the number of comparisons and the search time [1-7].

Many string matching algorithms which had been proposed earlier, like Brute force [8], KMP [9] and Boyer moore[10], are the basic algorithms to most ofthe recently proposed algorithms.

In Brute force string matching mechanism, the left most character of the pattern is compared with the corresponding character of the text. If a complete match/mismatch occurs, then the search process is shifted exactly one position to the right.

In the KMP string matching mechanism, the left most character of the pattern is compared with the corresponding character of the text. If a complete match/mismatch occurs then the shift position is calculated according to the KMP skip rule.

In Boyer-moore string matching mechanism, the right most character of the pattern is compared with the corresponding character of the text. If complete match/mismatch occurs then the shift position is calculated according to the Bad character rule and the good suffix rule.

This work proposes parallel string matching with omega model, analyses its results and finally concludes the study.

2. PROPOSED SYSTEM STRUCTURE

In this paper, we propose a system for parallel processing with omega model. The name of the omega model is taken from the omega switch which it uses for inter process communication [11, 12, 13]. The switch supports a processor-to-processortransfer rate of some limitation. Figure-1 illustrates parallel string matching with omega model.

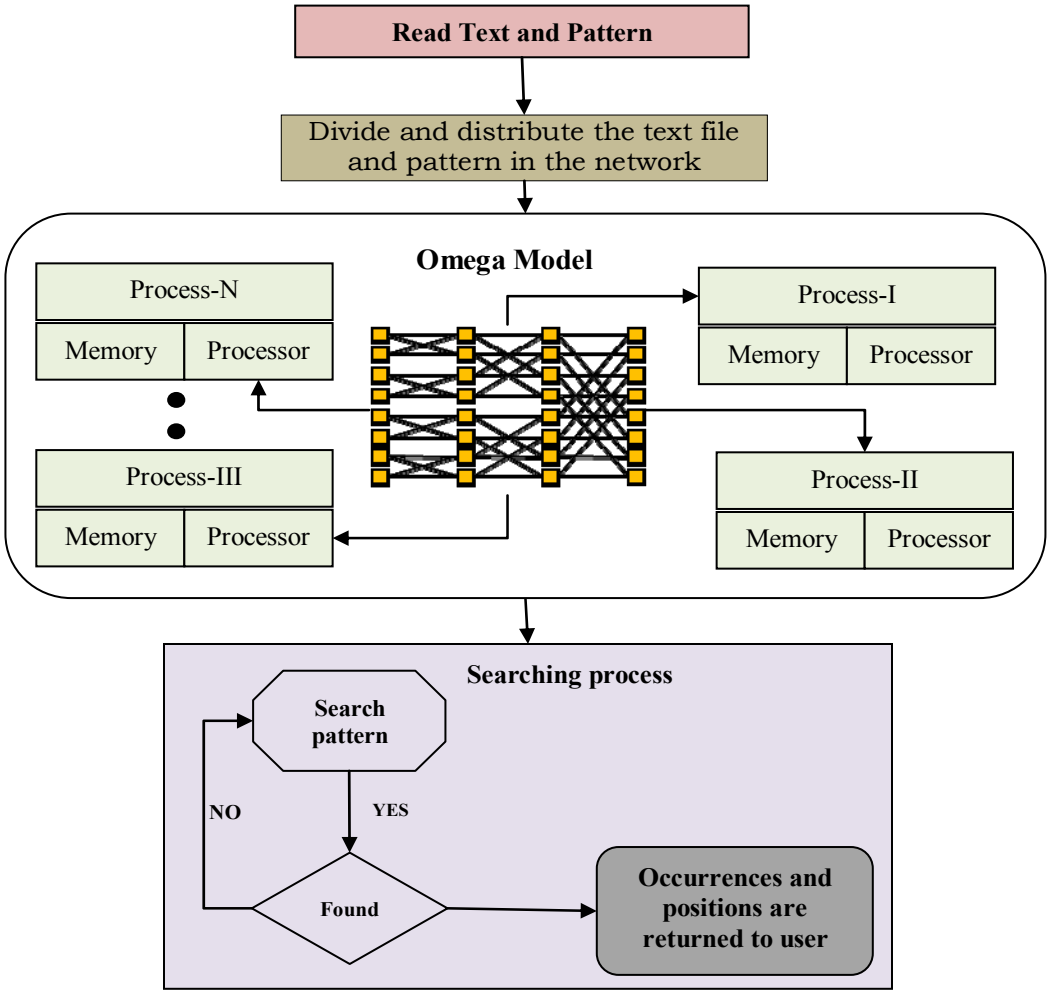

Figure-1. Parallel string matching with omega model.

3. PROCESS

m elements of the pattern are initially distributed to m processors on the first column, one processor and $L_{i,j}$ distributed to the m*(n-m+1) processors on the row and column. Where $1 \le i \le m$ and $m \le j \le m*(n-m+1)$. Each first processor $p_{i,1}$, where $1 \le i \le m$, broadcasts the element ci,1 to every processor in the ith row using only row buses. After this multiple row broadcast communication operations, each processor $P_{i,j}$ saves received element as $r_{i,j}$. Each processor $P_{i,j}$ compares $r_{i,j}$ with $X_{i,j}$, if $r_{i,j} = X_{i,j}$ sets result=1 otherwise, result=0

Compare and Set results: Each processor $P_{i,j}$ compares $r_{i,j}$ with $X_{i,j}$, if $r_{i,j} = X_{i,j}$ it sets result=1 otherwise, result=0

Sum up 1's: perform a one-dimensional binary prefix sum operation on each column simultaneously for the value of result. Each processor $P_{i,j}$ where $1 \le (i,j) \le m$ stores the binary prefix sums to $b_{i,j}$.

Based on hamming distance: If $P_{m,j} = 0$ then the string matching(i.e. exact string matching) otherwise approximate string matching with k mismatches.

Text string T(n)=
AGATAGATTTGGGAAACCC
TGGGCCCTTTAGAAAGAACT
GATAGATTGATCGGGAAACA
Subtext-file-1: AGATAGATTTGGGAAACCC
Subtext-file-2: TGGGCCCTTTAGAAAGAACT
Subtext-file-3: GATAGATTGATCGGGAAACA
Pattern string P(m)= TTTAGGG

TTTAGGG	Line 1 from Subtext-file-1	Line 2 from Subtext-file-1	Line n from Subtext-file-1
TTTAGGG	Line 1 from Subtext-file-2	Line 2 from Subtext-file-2	Line n from Subtext-file-2
TTTAGGG	Line 1 from Subtext-file-3	Line 2 from Subtext-file-3	Line n from Subtext-file-3

TTTAGGG	<TTTAGGG, Line 1 from Subtext-file-1>	<TTTAGGG, Line 2 from Subtext-file-1>	<TTTAGGG, Line n from Subtext-file-1>
TTTAGGG	<TTTAGGG, Line 1 from Subtext-file-2>	<TTTAGGG, Line 2 from Subtext-file-2>	<TTTAGGG, Line n from Subtext-file-2>
TTTAGGG	<TTTAGGG, Line 1 from Subtext-file-3>	<TTTAGGG, Line 2 from Subtext-file-3>	<TTTAGGG, Line n from Subtext-file-3>

Proposed mechanism
a) Open the TEXT file and Read the number of lines from the file b) Read the pattern of size m and number of processors c) Split the file into sub text files based on the processor count d)Distribute the sub text files and patterns to all the processors in the omega model structure e) Each Processor searches the pattern in the sub text file with string matching and returns matched position and occurrence.

4. RESULTS

The genome sequence is considered as data set for evaluating the efficiency of the proposed approach with omega model [14,15].The search times of proposed approach with omega model and existing mechanisms are depicted in Table-1. These search times are obtained for genome sequence data sets of different data sizes.

Table-1. Search times of different methods.

Method / File size(Mb)	Proposed	Brute-force	KMP	Boyer-Moore
50	15	504	499	328
55	16	723	702	436
60	15	687	671	468
65	12	720	702	484
70	14	701	686	515
75	14	789	780	562
80	15	867	858	546
85	63	1745	1732	577
90	16	1745	1732	655
95	16	1924	1903	702
100	16	1602	1591	671

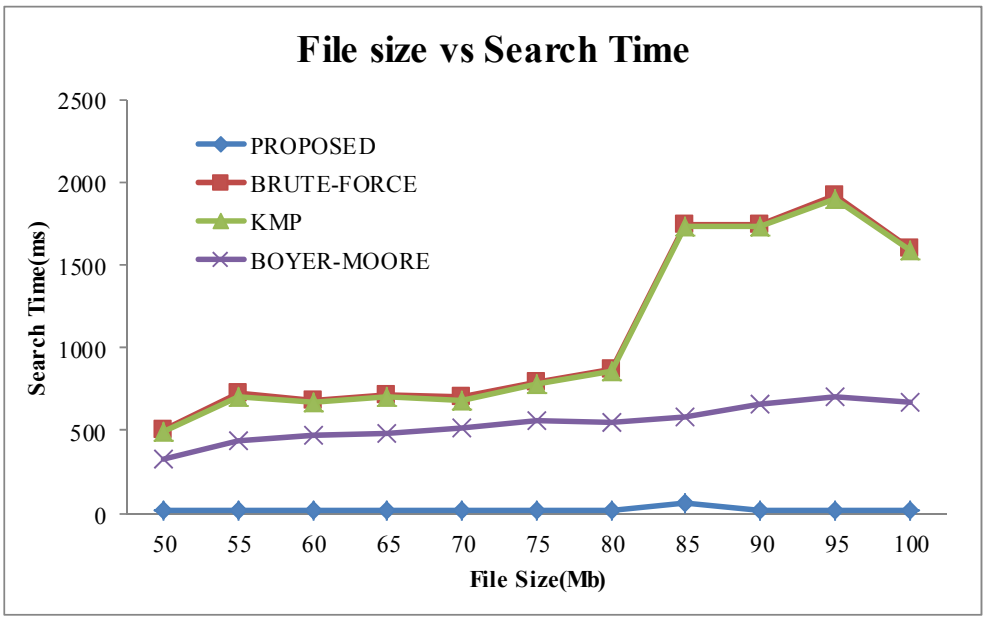

Figure-2. File size vs. search time.

The Figure-2 has shown the search time results of proposed approach with omega model and existing mechanisms in graphical representation. From the Figure-2, it is observed that proposed approach can reduce the search time than existing mechanisms.

5. CONCLUSIONS

In this paper we have proposed algorithm that provides parallel computing in the context of omega architecture. The proposed algorithm reduces the number of comparisons and also reduces the search time compared with existing string matching algorithms.

ACKNOWLEDGEMENTS

This work is partially supported by UGC, Government of INDIA, Technology Bhavan, New Delhi-110016

REFERENCES

[1] Crochemore, Wojciech Rytter and MaximeCrochemore "Text algorithms. Oxford University Press, 1994.

[2] Y Simon and MInayatullah M. 2004. Improving Approximate Matching Capabilities for Meta Map Transfer Applications.Principles and Practice of Programming in Java. pp. 143-147.

[3] K S Grabowski. 2009. Average-Optimal String Matching. Journal of Discrete Algorithms.pp. 579-594.

[4] L Luis Russo *et al*. 2009. Approximate String Matching with Compressed Indexes Algorithm.pp. 1105-1136.

[5] L Ilie. 2010. The Longest Common Extension Problem, Revisited and Applications to Approximate String Searching.Journal of Discrete Algorithms.pp. 418-428.

[6] K Fredriksson and Grabowski. 2009. Average-Optimal String Matching. Journal of Discrete Algorithms.pp. 579-594.

[7] S Viswanadha Raju *et al.* 2011. Recent Advancement is Parallel Algorithms for String matching on computing models - A survey and experimental results. LNCS, Springer. pp. 270-278.

[8] Aho V Alfred V and John E Hopcroft. 1974. Design and Analysis of Computer Algorithms. Pearson Education India.

[9] D Knuth, J Morris and V Pratt. 1977. Fast pattern matching in strings.SIAM Journal on Computing. pp. 322-350.

[10] R Boyer and J Moore. 1977. A fast string searching algorithm. Communication of the ACM, pp.762-772.

[11] John Garofalakis and Eleftherios Stergiou. An analytical performance model for multistage interconnection networks with

blocking.Communication Networks and Services Research.pp. 373-381.

[12] Josep Torrellas and Zheng Zhang. 1977. The performance of the cedar multistage switching network. Parallel and Distributed Systems.pp. 321-336.

[13] Suresh K Bhogavilli and Hosame Abu Amara. 1977. Design and analysis of high performance multistage interconnection networks.Computers. pp. 110-117.

[14] http://www.ncbi.nlm.nih.gov/BLAST/blastcgihelp.shtml.

[15] https://en.wikipedia.org/wiki/FASTA_format.