Optimizing Agricultural and Construction Tool Usage through Android Devices: A Comprehensive Survey

Pradeep Kumar KG [1], Aneesh S Mayya [2], Anwesh Krishna B [2], Chinmaya Thejasvi U S [2], Ganesh B S [2]

[1] Faculty, Department of CSE, Vivekananda College of Engineering & Technology, Puttur, Karnataka
[2] Students, Department of CSE, Vivekananda College of Engineering & Technology, Puttur, Karnataka

Abstract
An all-encompassing rental and booking platform for agricultural tools and construction tools designed to revolutionize the workflow in the agricultural sector and construction sector respectively. It is a versatile tool of central application that will bridge the gap between demand of tools and its supply. This will help the owners to effectively run their farm-lands. Providers will keep inventory for the tools and implementation of tool allotment. When customers don't find the tools that they need in the current inventory they may purchase those from hardware shops. In that situation, the customers can buy tools from the selected hardware shops connected with the site by direct integration with the site owner's shop. Real-time online tool availability checks and vendor invoice generation are required in this process when clients want to buy.
A flexible communication system that can be developed as an application to facilitate customer communication directly with suppliers, which will be considerably more convenient than the conventional approaches and will remove the above.

Keywords: Farmer, Agriculture, Construction, Tools Management, Android Application

1. Introduction

Agricultural tools support farmers in their work, such as the use of cutting-edge equipment on the farm and the provision of free time and labor efforts from workers at all production levels. A farmer's life on agricultural land is made much easier by the evolving agricultural gadgets that follow trends. To the greatest extent, it is impossible to view a farming region nowadays without the use of contemporary agricultural equipment. With the right equipment, farmers can work more efficiently on the field, potentially averting the world's food deficit in the near future. The availability of agricultural equipment boosts sustainability, productivity, and efficiency, which helps to satisfy future food demands and contributes to economic growth and turnover. Farm equipment is used in every step of the process, from plowing and tilling the land to choosing the best seeds, planting them, and watering and grazing cows. It helps farmers prepare their fields and manage them so they can plant and harvest seeds. Additionally, with the fast population growth, urbanization, and ruralization, tools and machinery can significantly lower the labor and cost of farm work, prevent pest attacks, and occasionally increase agricultural output. Some of the farming tools which are much used by the agriculturists and farmers are shown in below figure 1.

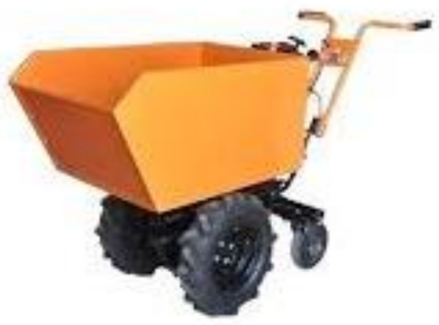
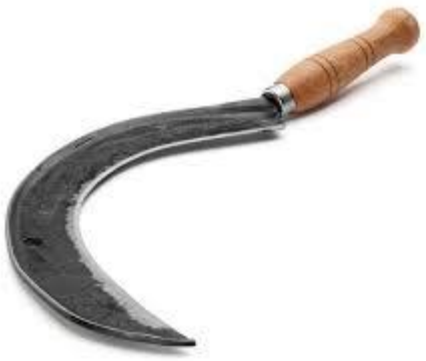
a. Pick axe and spade	**b. Gardening Fork**
c. Hand trowel	**d. Weed/bush cutter**
	
e. Farm cart	**f. Sickle**

Figure 1. Farming tools

Construction tools, assists from site selection to accomplish site-projects safely, accurately, and cost-effectively within estimated time. Advanced tools also alleviate the physical strain on workers, improving their overall well-being also benefits in time management for the work contractors, supervisors. Moreover, modern technology in construction instrument, allows for time management, better accuracy and control, efficiency, reducing errors and the need for rework. Many complex tasks would be difficult or even impossible to perform without specialized tools, which further emphasizes their necessity in both farming and construction industries. The below diagram(figure 2) consolidates most frequently used construction tools and materials which can be rented or purchased from vendors like surveying equipment, leveling instrument, hammer, 4-piece concrete tool set, excavator, bulldozer, cement and concrete mixers etc.

a. 4-Piece concrete tool set	**b. Surveying Equipment**
c. Leveling instrument	**d. Hammer**
e. Excavator	**f. Bulldozer**

g. Dump trucks	**h. Transit concrete mixture**
i. Small concrete mixer	**j. Essential building materials**

Figure 2. Representing most used construction tools and materials

2. Literature Survey

Ramshankar Choudhary et.al [1] presents details on using Boyer-Moore algorithm for solving crucial problem of string matching. Boyer-Moore compares characters from one side to another and uses two precomputed functions, the good-suffix and bad-character shifts, to optimize the search process. It also included improvements to BM such as the Boyer-Moore-Horspool (BMH), Boyer-Moore-Horspool-Sunday (BMHS), BMHS2, improved BMHS, Boyer-Moore improvement (BMI), and composite Boyer-Moore (CBM) algorithms. These variants enhance performance by simplifying preprocessing, improving shift functions, and utilizing previous comparison information. They explained these algorithms, compares them using an example, and analyses their efficiency in different scenarios.
David Kulp [2] provides an analysis of the Boyer-Moore string search algorithm, emphasizing its efficiency and practical applications. It performs efficient exact pattern recognition in un-indexed sources by comparing characters from one side to another. This algorithm proposed by Boyer Moore in 1977, performs efficient exact pattern recognition in un-indexed sources by comparing characters from one side to another. They illustrated the Boyer-Moore process with examples and discusses modifications for longest matching, which involves searching from one side to another and ensuring no partial matches are missed. Despite modifications potentially slowing down the average performance, the worst-case time complexity remains O(N + M).
Domenico Cantone et.al [3] gives comparison of Boyer-Moore algorithm with some of the most efficient string-matching algorithms such as quick search, Horspool, Tuned Boyer-Moore etc. The techniques are examined in terms of run-time efficiency, the number of text character inspections, and the number of character comparisons. They also help in achieving good results especially in case of very short patterns or small alphabets. This paper also introduces a variant of the Fast-String-Matching algorithm, a recent extension of the Boyer-Moore algorithm, which aids in efficient string matching. This variant of Boyer-

Moore String-Matching works on the basis of Forward-Fast-Search which makes the string searching and matching easier.

Shmuel T. Klein et.al [4], offers information about how Boyer–Moore searches on binary texts. Furthermore, this article generalised, which is useful when dealing with matching in compressed texts. Boyer-Moore uses two heuristics to shift the pattern: delta1 and delta2. Delta1 is not that good for binary data because the two characters 0 and 1 are reasonably frequent. The second heuristic, delta2, considers mismatches and suffix reoccurrences. It is necessary and optimisations of the algorithm has a significant gain in performance of search for a given text, especially for highly compressed texts. Additionally, they provide a theoretical examination of the time complexity of their extended algorithm, emphasizing its benefits compared to traditional methods in specific scenarios

Paul Y Gloess et.al [5] outlines the process and results of using the Boyer-Moore theorem to confirm the accuracy of a parser. The proof consisted of 147 functions and lemmas, listed in the appendix of a second paper that also explained the original problem, the development of the proof, and the need for the typographical map of auxiliary functions, which the authors remarked they never would have conceived of in a manual proof. It ended with remarks on the experiment and the practical usefulness of the theorem prover: And so, it was that little knowledge of the internal working of the theorem prover, and no dummying up of the problem to make it easier for the theorem prover to do its job were necessary to constitute a mechanical proof. Despite some contextual issues, the paper applies a clear-headed methodology to ‘automatic theorem proving’ earning a fair level of confidence. And yet, its conclusions – that these tools are valuable and that they would be particularly useful in demonstrating the correctness of software – are mistaken about software’s role in contemporary society. In fact, the intermediate steps of a human proof are only rigorous because someone deems them so. Like any domain of human endeavour, there is a basic common-sense trust that allows us to carry on; yet ultimately, one requires proof only when someone who knows better challenges it.

Julio C. Rivera et.al [6] provides a study on an improved pattern matching technique based on Boyer-Moore, which uses matched partitions to avoid unnecessary comparisons. The paper reviews related work, presents the new model, and concludes with experimental results. The authors offer in-depth analyses and comparisons with existing approaches, highlighting notable improvements in speed and accuracy. These findings showcase the potential of the new model to push forward advancements in pattern matching, providing resolution that demand quick and reliable text search capabilities.

Thierry Lecroq et.al [7] discusses about Boyer-Moore algorithm which is a widely-used method for efficient string matching, particularly in applications like information retrieval and molecular biology. It scans text from right to left, employing precomputed functions for shifting the matching window. Among these functions, the best matching shift aligns the longest pattern suffix with a matching text prefix. The paper highlights the algorithm's effectiveness and versatility, demonstrating its application in diverse fields such as information retrieval, where quick and accurate string matching is crucial, and molecular biology, where it can be utilized to determine sequences within genetic data. By leveraging these precomputed shift functions, the Boyer-Moore algorithm achieves significant performance improvements, making it a valuable tool for efficient string-matching tasks.

Heikki Hyyro [8] discusses about how Boyer-Moore algorithm helps in the search process. Our goal here is to make this preprocessing algorithm much easier to understand by showing that it shares fundamental ideas with the Morris-Pratt algorithm. By seeing how the Boyer-Moore preprocessing step fits into this more intuitive mould, we gain a purchase on the Boyer-Moore algorithm that we wouldn’t otherwise have, especially with regard to its efficiency. I believe that this paper accomplishes these three valued goals. The Boyer-Moore algorithm can be implemented in a less intimidating fashion than was generally perceived before it was recognised to have deeply Morris-Pratt-ish aspects.

Ricardo A. Baeza-Yates et.al [9] focuses on analysing the Boyer-Moore string-matching algorithm, which is essential for tasks like text editing and data retrieval. It determines its qualities, the amount of character comparisons made in its operation. The authors tackle the problem of analysing the Boyer-Moore algorithm using more refined algebraic instruments of combinatorics. In particular, the authors reduce the problem to a stationary process, applied for large alphabets. The Boyer-Moore algorithm minimises the amount of character comparisons; this is a key factor in its overall speed and efficiency. Before that, the problem was understood in a heuristic way. Now they were analysing its systemic behaviour. The authors were able to explain the strengths of their optimised algorithm using combinatorial insights about sets and classes.

Jornaa Tarhio et.al [10] says Boyer-Moore algorithm outperform existing methods, especially for large patterns and alphabets. These are able to speed up the process of finding matches by many folds. They are simple – the pattern is scanned a few times along the text to come up with a molecular fingerprint. The observed result demonstrated significant speed-up of the process compared with the previously known methods. In particular, they demonstrated success in scaling to large patterns and alphabet sizes. They highlight, it is possible to design new and innovative conceptualization that can improvise the execution of a well-studied problem and its applications, such as text editing, data retrieval and others, that rely heavily on the speed of the matching. Efficient and trustworthy approach to the string-matching problem is very much required in designing modern software that deals with data. They concluded by stressing on need of developing tools that can tackle the ever-growing needs of computational tasks emerging every day.

Nadia E1-Mabrouk et.al [11] addresses the challenge of approximate pattern matching, which involves finding patterns with mismatches in a text from a finite alphabet. This paper reviews various algorithms, noting that Boyer-Moore-based methods are the fastest for exact matching. By reviewing the various approaches, the paper offers important perspectives on the current state of approximate pattern matching algorithms and identifies areas where Boyer-Moore-based methods have been successfully adapted. The authors underscore the importance of continuing to refine these algorithms to balance the trade-offs between speed and accuracy, making them more applicable to a broader range of real-world problems.

Yusuke Shibata et.al [12] investigates the challenge of compressed pattern matching, aiming to find patterns in compressed text without decompressing it, thus speeding up the search process. The focus is on achieving faster search times in compressed files compared to original files, a goal more difficult than surpassing the combined time of decompression and search. The paper explores various techniques and algorithms designed to handle different compression schemes. It highlights the difficulties and complexities involved in adapting traditional pattern matching algorithms, like Boyer-Moore, to work effectively on compressed data. The authors present experimental finding showing that their proposed methods can indeed achieve faster search times in compressed files, showcasing significant improvements over conventional approaches that require decompression.

Robbi Rahim et.al [13] presents a study on the Boyer-Moore algorithm, highlighting its string-matching process through visual simulation. The algorithm's steps involve checking for character mismatches and shifting the pattern accordingly, making it one of the most effective string-matching algorithms. The visual simulation in the paper helps to visualize and understand each step of the algorithm, from initializing the pattern to completing the search. The method assists in understanding the mechanics of the Boyer-Moore algorithm and emphasizes its status as leading string-matching algorithm available. By providing a clear demonstration of its workings, the paper contributes to the educational resources available for studying algorithms in computer science and related fields.

Thierry Lecroq [14] offers detail on Boyer-Moore algorithm which functions as a fast-string-matching method that works by matching a word against a text from right to left, finding the longest suffix at each position, and shifting the word for the next match attempt. The algorithm's strategy involves finding the longest suffix of the pattern that matches a prefix of the text at each position during the scan. By identifying these matches and utilizing efficient shift mechanisms, Boyer-Moore optimizes the search process, reducing the number of comparisons required to locate instances of the pattern within the text. Overall, this paper underscores the Boyer-Moore algorithm's robustness and efficiency in various applications requiring rapid and accurate string matching, such as text search engines, bioinformatics, and data processing tasks.

Sinan Sameer Mahmood Al-Dabbagh, et.al [15] offers a solution to string-matching problem which functions as a crucial in computer science, impacting areas such as intrusion detection, search engines, and computational biology through Boyer-Moore algorithm. This paper also discusses the implementation and optimization of the Boyer-Moore algorithm for various applications. They may also explore its adaptation and integration into specific domains such as intrusion detection systems, where rapid pattern matching is critical for identifying suspicious activities or patterns in network traffic. Similarly, in search engines and computational biology, efficient string matching enables quick retrieval of relevant information and analysis of genetic sequences, respectively. Overall, the paper contributes to advancing the understanding and application of the Boyer-Moore algorithm in real-world scenarios, demonstrating its utility and effectiveness in addressing diverse string-matching challenges across different fields of computer science.

3. Methodology

The proposed methodology provides a detailed visualization of the application's processes and the interactions between various system components. The following diagram in figure 3 represents, a workflow of a worker may be a farmer or a construction site worker via our proposed application. Figure 4 shows, handling of new tool orders placed by the farmer and/or construction site work contractors by managing inventory at hardware business shop. Figure 5 shows , when a customer places an order with payment gateway. Overall the proposed android application, serves as a single window to handle all the tools management.

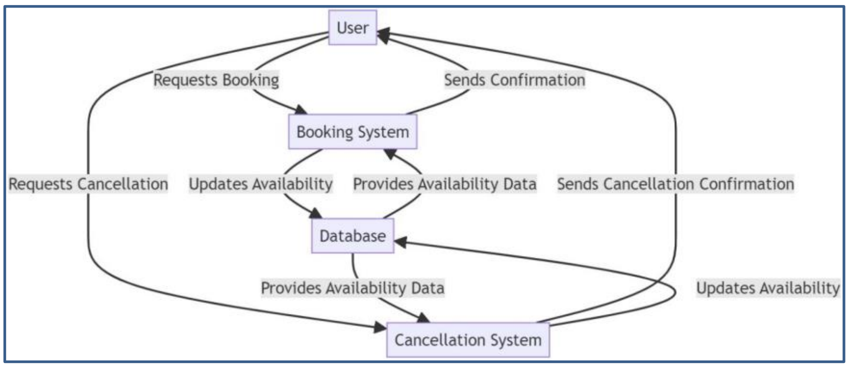


Figure 3. User requests for a tool

The first part of what is really the application workflow is that tool providers enter information about the tools they provide. This ends up being a huge part of the set-up. A tool provider has to put all of the tool names they have, descriptions of the tools, and images of the tools, to inform the customers to make choices about their needs. This sort of database-building is the runway to the goods: if a potential customer wants information about a saw and this tool is up-to-date in someone else's system, this should appear here. Often information in the tool database will be time- and other context-dependent; the expectation is that the tool provider will be able to return to the tool and change information about the available tools they are renting or selling as need be.
Once the respective tools are built and input at the system, the key to find them is just enough to key in tool name or category of tool. The Keyed data are read by the system and given to block to take decision. (The searching club) The feedback returned from block is used to confirm whether the product is available or not. The necessary data is brought from the matched database.(the main control and instrument board) The search bar where the customer types their search key. The Boyer-Moore string search algorithm can be utilized(named for its creators Edgar E Boyer and J Stolfi) which is highly efficient for searches of text strings. It does this by preprocessing the search pattern and then comparing the pattern against the data. This works because Boyer-Moore skips over large sections of text that it can tell will not match the pattern being sought. This means it is far faster than a conventional linear search algorithm that would progress one character at a time through the database. The improvement of speed is tremendous. The algorithm means that, even if the database was very large, and the customer's input into the field already extensive, the system would still return the tools required within seconds of the customer beginning input.

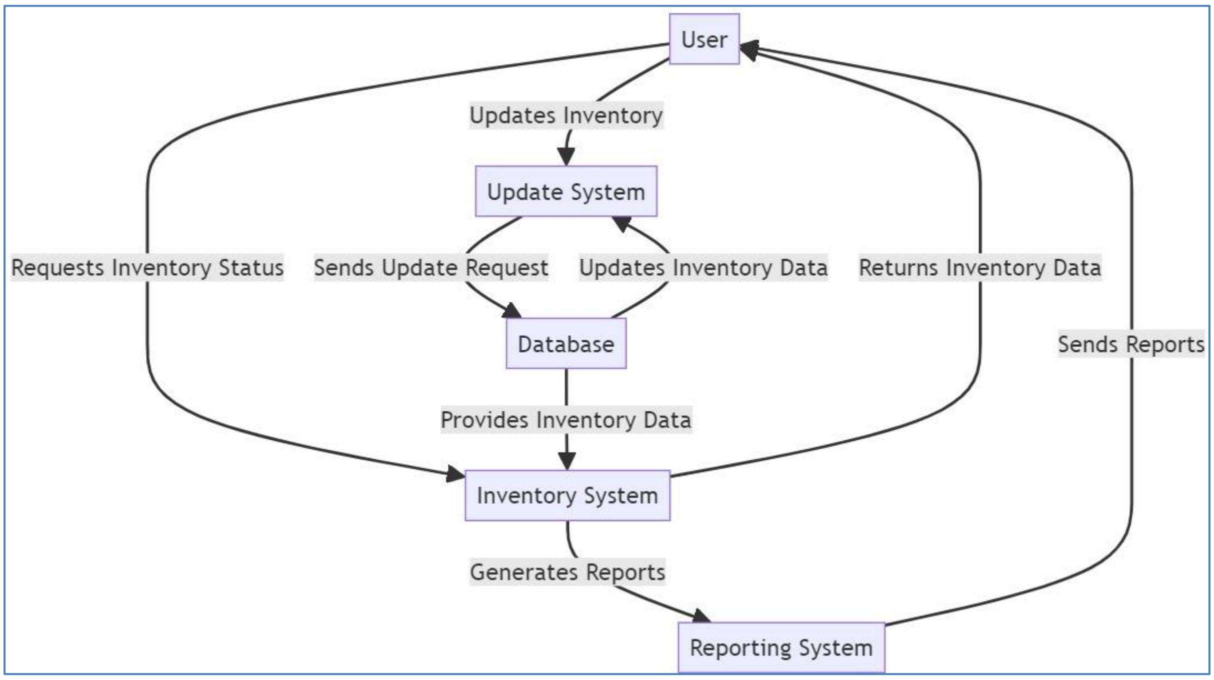


Figure 4 Hardware-business owner window receiving orders also maintaining inventory

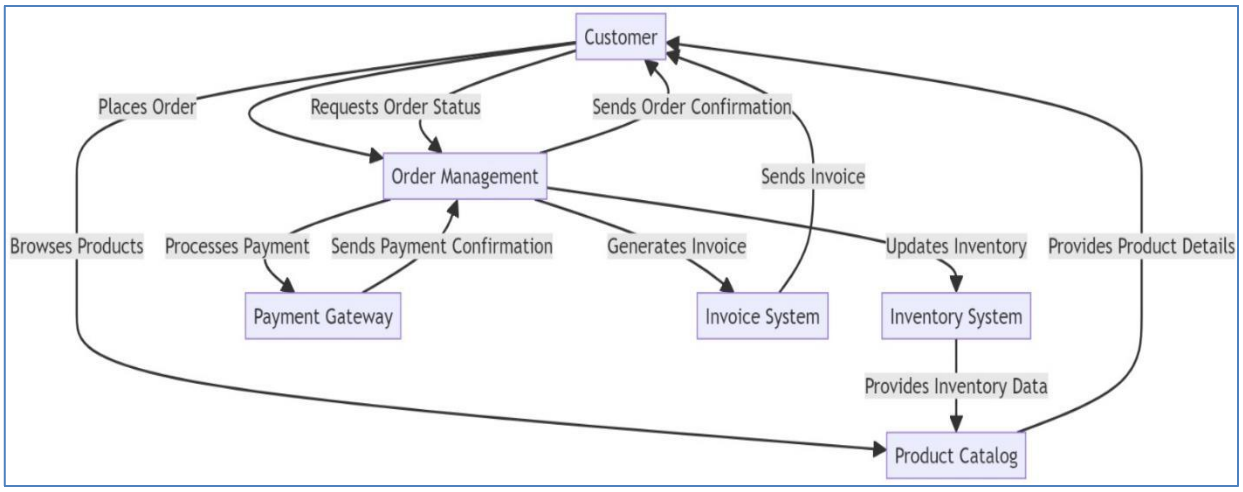


Figure 5. Order placement

The steps followed in Boyer-Moore as specified below highlights major phases in it.

Step 1. Input Text and the Pattern

Step 2: Preprocessing

Bad Character Rule Preprocessing

Initialize bad character table with -1 (default for all characters)

Fill the table with the last occurrence of each character in the pattern

Good Suffix Rule Preprocessing

Process the pattern in reverse to find the borders

Fill in the good suffix table

Step 3: Start comparing pattern from the rightmost end with the text

Compare pattern with text from right to left

If found, then print

else

Calculate shift using Bad Character Rule or Good Suffix Rule

Step 4. Stop

For instance, if the customer enters in the name of a tool or a portion of its description into the search engine, a search algorithm would almost instantly match it with the entries it finds in the database, and provide the search results. This 'instant' search experience is a key functionality that drives a responsive and pleasant user experience. After the tool the customer is looking for has been identified, clicking on it will trigger the showcase of its rental price or availability, and/or further specifications in other tabs. If everything seems acceptable to the customer, they could then proceed to make a confirmation. Thus, with as few clicks as possible, we see the customer performing 'search', 'pick' and 'confirm' to finish a 'single' transaction in that order – illustrating the common instinct to keep the system free from the intricacies of the real world as much as possible.

The second function is the search and selection function. Anyone who intends to rent or buy tools can call up the database, search for the relevant items, remove unwanted information using refinement functions, and look at illustrations before deciding if they want a tool. Once the tool is checked out of the database, the second function is triggered: management starts. The provider of the tool can update listings, modify prices, and manage availability of the offered tools in real time, whenever he or she is needed. QA checks may be adapted during the database update as an option in the management function. This is where the trust factor for the user's goodwill lies. Whenever some entries in the database have been moved, modified, added or deleted, the system must automatically perform a QA check. It must flag discrepancies and identifies potential errors to the system's manager. The manager can then check these fields for correctness and the database is updated after the approval. The dialogues between the application and the database make sure that customer enquiries, tool availability and other relevant data points are processed in a smooth way, handling of entire tool management will become much easier. The system's adaptability makes it suitable for various applications, whether in business or consumer-oriented environments. As the project evolves, additional features like automated notifications or advanced reporting tools could be integrated, further enhancing its functionality and appeal to both tool providers and customer.

4. Conclusion

The planned platform focuses on providing good interaction between the customer and the tools provider. Normally we see that many people visiting shops in order to buy or rent tools which is quite waste of time. As a solution, a mobile application is designed which is user-friendly and helps the customer to have a good communication with the tool provider. Mobile app allows tool provider to add tools information along with their image. Customer can login to the application using their credentials and then looks for the tools available. Customers also get the description regarding the tools and their price. Hence customer gets the clear-cut information about the tools with rental and purchase options. The proposed channelizes the way of getting tools with proper information, in-turn helping people to easily access the required tools.

References

1. Ramashankar Choudhary, Prof. Akhtar Rasool & Dr. Nilay Khare, "Variation of Boyer-Moore String Matching Algorithm: A Comparative Analysis", International Journal of Computer Science and Information Security, Vol 10, No.2, February 2012.
2. David Kulp, "Very Fast Pattern Matching for Highly Repetitive Text", Journal of Department of Computer Science, University of Canterbury, December 22 1992.
3. Domenico Cantone & Simone Faro, "Forward-Fast-Search: Another Fast Variant of The Boyer-Moore String Matching Algorithm", Journal of Automata Languages and Combinatorics, Proceedings of the Prague Stringology Conference 2003.
4. Shmuel T. Klien & Miri Kopel Ben-Nissan, "Accelerating Boyer-Moore Searches on Binary Texts", Journal of Theoretical Computer Science, Department of Computer Science, bar Ilan University, Ramat-Gan 52900, Israel, 2009.
5. Paul Y. Gloess, "An Experiment with The Boyer-Moore Theorem Prover: A Proof of The Correctness of a Simple Parser of Expressions", International Journal from Sri International Menlo Park, California 94025,U.S.A and Boursier de Recherche I.R.I.A., 78150 Le Chesnay, France, 2005.
6. Julio C. Rivera, Paul M. Di Gangi, James L. Worrell, Sammuel C. Thompson, Allen C. Johnston, "Pattern Matching Based on Boyer-Moore Algorithm, A New Method", Journal from Management, Information Systems and Quantitative Methods Department, The University of Alabama at Birmingham, AL 35294, June 2015.

7. Maxime Crochemore & Thierry Lecroq, “A Fast Implementation of The Boyer-Moore String Matching Algorithm”, Journal from University of Rouen Normandy, France, January, 2008.
8. Heikki Hyyro, “On Boyer-Moore A Preprocessing”, Department of Computer Sciences, University of Tampere, Series of Publications, D-NET Publications, D- 2004-1, November, 2004.
9. Ricardo A. Baeza-Yates, Gsaston H. Gonnet & Mirelle Regnier, “Analysis of Boyer- Moore Type String Matching Algorithms”, Conference of Proceedings of the First Annual ACM-SIAM Symposium on Discrete Algorithms, California, U.S.A Vol.1 22-24 January 1990.
10. Jornaa Tarhio and Esko Ukkonen, “Boyer-Moore Approach to Approximate String Matching”, Journal from University of Helsinki, Department of Computer Science, Teollisuskatu , 23 SF-00310 Helsinki, Finland, 1993.
11. Nadia El-Mabrouk and Maxime Crochemore, “Boyer-Moore Strategy to Efficient Approximate String Matching”, Annual Symposium on Combinatorial Pattern Matching, International Research Paper, Vol 1075, 2005.
12. Yusuke Shibata, Tetsuya Matsumoto, Masayuki Takeda, Ayumi Shinohara and Setsuo Arikawa, “A Boyer-Moore Type Algorithm for Compressed Pattern Matching”, Journal from Department of Informatics, Kyushu University, Fukuoka Japan, 2000.
13. Robbi Rahim, Ansari Saleh Ahmar, Ayu Putri Ardyanti and Dicky Nofriansyah, “Visual Approach of Searching Process Using Boyer-Moore Algorithm”, International Conference on Information and Communication Technology, Journal of Physics, Conf. Series 930, 2017.
14. Thierry Lecroq, “A Variation on The Boyer-Moore Algorithm”, International Research Paper of Theoretical Computer Science, Vol. 92, Issue 01, 6 January 1992.
15. Sinan Shameer Mahmood Al-Dabbagh, Mustafa Abdul Sahib Naser, Nuraini Bint Abdul Rashid and Nawaf Hazim Barnouti, “Fast Hybrid String Matching Algorithm Based on The Quick-Skip and Tuned Boyer-Moore Algorithms”, International Journal of Advanced Computer Science and Applications, Vol. 8 No.6, 2017.
16. Pradeep Kumar KG, Dr. Karunakara K, and Dr. Thyagaraju G. S., “Automated Identification of Diabetic Retinopathy: A Survey”, International Journal on Recent and Innovation Trends in Computing and Communication (IJRITCC), June 17, Volume 5, Issue 6, ISSN: 2321-8169, PP: 514 – 520