IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

PATTERN SEARCHING ALGORITHM

Mrs. T.Leena PremaKumari
Head,Assistat Professor,Fatima College,Madurai

ABSTRACT:

This paper presents an accessible overview of pattern searching algorithms, which are vital tools in computer science for finding specific patterns within data. It break down the complexities of these algorithms into understandable concepts suitable for readers unfamiliar with advanced technical jargon. By employing straightforward language and examples, this paper aim to provide a clear understanding of how pattern searching algorithms function and their significance in various applications.

INTRODUCTION:

Pattern searching algorithms form the backbone of numerous applications in computer science and data analysis. From searching for keywords in documents to identifying genetic sequences in bioinformatics, these algorithms play a crucial role in various domains. This algorithm are used identify certain pattern in large data set it is mainly used for string matching, retrieval and analysis of data. Through this paper let we discuss about the basic concepts, implementation of pattern searching algorithm and the future enhancement of this algorithm.

LITEREATURE SURVEY:

In the realm of pattern searching algorithm, numerous studies have contributed for efficiently locating the patterns within text or data structures, let's explore the content of those studies:

- **Knuth, Morris and Pratt** proposed one of the foundational algorithms in this domain, known as **Knuth-Morris-Pratt algorithm.** This is efficient in searching for patterns by avoiding unnecessary comparisons. This algorithm introduced an elegant linear-time algorithm for string matching. Linear-time algorithm is like linear search it check each and every element in collection and also it is only efficient for simple unordered collection not for the large datasets.
- **Boyer and Moore** introduced another seminal algorithm**,** this algorithm remains one of the most efficient pattern searching algorithms in practice. Their approach employs heuristic rules to quickly skip over portions of text, resulting exceptional performance in practice.
- **Gusfield** provided comprehensive coverage of pattern matching algorithms in his book "**ALGORITHMS ON STRINGS, TREES AND VARIOUS SEQUENCES".** in this book he explores various pattern searching algorithms, including kmp,boyer and Moore and also other varients,he also discuss advanced topic such as suffix trees and use of finite automata for pattern matching.
- **Crochemore and Ryther** presented a survey of pattern matching algorithms, focusing on their theoretical aspects and practical implementations. Their work provides a valuable resource for

understanding the principles of pattern searching algorithms and their applications in diverse domains.

- Recently, Navarro **and Rytter** offered a comprehensive overview of pattern matching algorithms in their book "**Flexible Pattern Matching in Strings**." They cover a wide range of algorithms, including exact and approximate matching algorithms, as well as advanced data structures and indexing techniques for efficient pattern searching in large datasets.

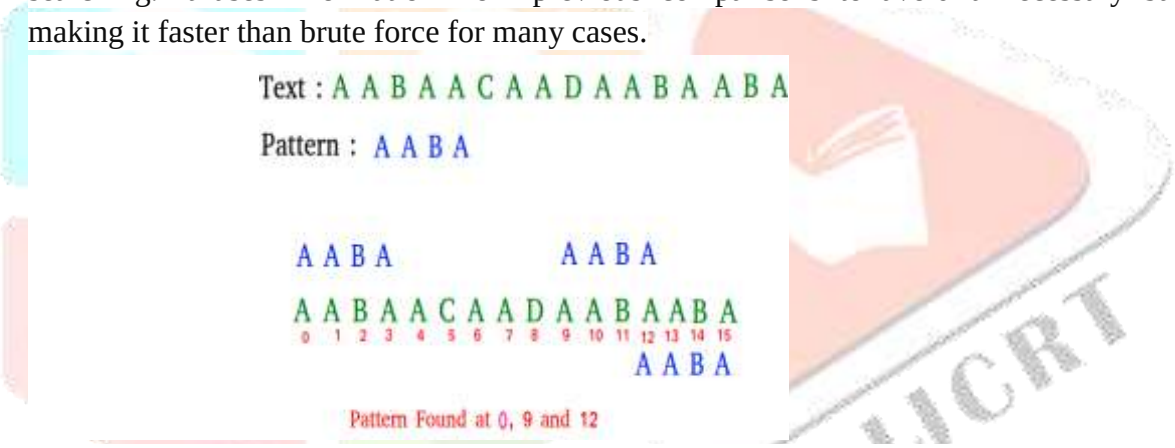
Overall, these studies collectively contribute to the rich literature on pattern searching algorithms, offering more knowledge about their theoretical foundations, practical implementation, and application across various domains.

Materials and Methods:

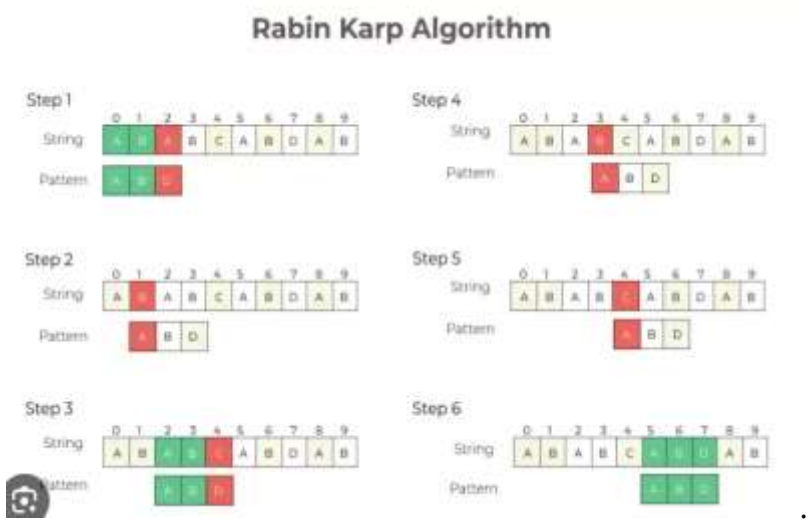
In this section let's see the methods involved in pattern searching algorithm and also it uncovers the existing system, implementation and future trends in this algorithm.

METHODS:

1. **BRUCE FORCE METHODS**: this method involves checking every possible position in the text for the occurrence of the pattern. it's simple but can be slow for large dataset.
2. **KNUTH-MORRIS-PRATT ALGORITHM (KMP):** KMP is an efficient algorithm for pattern searching. It uses information from previous comparisons to avoid unnecessary backtracking, making it faster than brute force for many cases.



3. **BOYER-MOORE ALGORITHM**: Similar to KMP, Boyer-Moore is a fast pattern searching algorithm. It works by skipping comparisons based on information gathered during the search process, which can lead to significant performance improvements.
4. **RABIN-KARP ALGORITHM** This algorithm uses hashing to quickly check whether a substring matches a pattern. It's particularly useful when multiple patterns need to be searched for simultaneously



5. **SUFFIX TREES**: Suffix trees are data structures that store all suffixes of a given text. They can be used to efficiently search for substrings or patterns within the text.
6. **FINITE AUTOMATA**: Finite automata are theoretical models of computation that can be used to recognize patterns within strings. They're particularly useful for simple patterns and can be implemented efficiently.

Each of these methods has its strengths and weaknesses, and the choice of method depends on factors such as the size of the text, the complexity of the pattern, and the desired performance characteristics.

EXISTING SYSTEM:

The existing system in pattern searching algorithm, encompasses a series of steps, from data pre-processing to pattern discovery and interpretation.by using these algorithms organization can extract knowledgeable data. Here's the steps involved in pattern searching algorithm:

- **Data Pre-processing**: The existing system begins with data preprocessing, where raw data is cleaned, transformed, and prepared for pattern mining. This may involve tasks such as removing noise and handling missing values.

- **Pattern Representation**: Once the data is prepared, it is represented in a suitable format for pattern mining. Common representations include transactional datasets, sequences, graphs, or other structured formats depending on the nature of the data and the type of patterns being mined.

- **Pattern Mining Algorithms:** The heart of the existing system lies in pattern mining algorithms, which analyze the prepared data to discover interesting patterns. These algorithms can be broadly categorized into two types:

    A. **Frequent Pattern Mining**:These algorithms identify patterns that occur frequently in the dataset, such as frequent itemsets in transactional data or frequent subsequences in sequence data. Popular algorithms for frequent pattern mining include Apriori, FP-Growth, and Eclat.

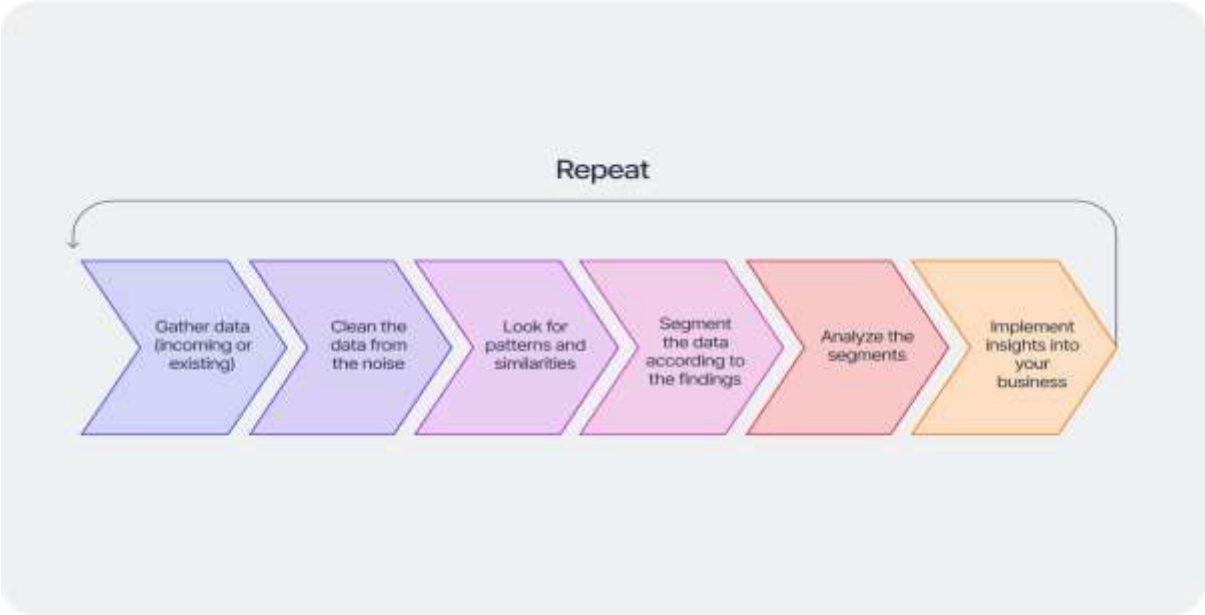
    B. **Sequential Pattern Mining**: These algorithms discover patterns that occur in a specific order or sequence, such as sequential patterns in time-series data or sequential patterns in clickstream data. Notable algorithms for sequential pattern mining include GSP (Generalized Sequential Patterns), SPADE, and PrefixSpan.

- **Pattern Evaluation:** Once patterns are discovered, they are typically evaluated based on various criteria such as support, confidence, or interestingness measures. This step helps filter out trivial or uninteresting patterns and focus on those that are truly meaningful or actionable.

- **Pattern Interpretation and Visualization:** Finally, the discovered patterns are interpreted and visualized to gain insights and inform decision-making. Visualization techniques such as bar charts, heatmaps, or network graphs may be used to present patterns in a human-readable format.

- **Application-Specific Analysis:** In real-world applications, the discovered patterns are often used for tasks such as market basket analysis, customer segmentation, anomaly detection, or predictive

modeling. The insights gained from pattern mining algorithms can drive business decisions, improve processes, or advance scientific research.



IMPLEMENTATON:

In my project "Home Rental Hub," I aim to streamline the process of finding rental properties. As part of the implementation, I've integrated a pattern searching algorithm to enhance the search functionality. This algorithm plays a crucial role in efficiently matching user queries with available rental listings. Let's delve into how this algorithm operates and its impact on the overall user experience in our journal's implementation section.

As I already mentioned pattern searching algorithm involves many steps for gaining desired results, here I am going to use those process to make the users to get the desired home.

1. **Data Integration and Pre-processing:**
   Home Rental Hub aggregates rental property listings from various sources, including landlords, property management companies, and real estate agents. The data is pre-processed to standardize formats, remove duplicates, and enrich property attributes such as location, size, amenities, and rental price.
2. **User preferences and semantic search :**
   Home Rental Hub analyses user preferences and behaviour data to understand user preferences and tailor search results accordingly. The platform utilizes semantic search to interpret user queries and property descriptions accurately. This allows users to receive relevant search results.
3. **Dynamic Filtering and Sorting:**
   Advanced filtering and sorting options are offered to users to refine search results based on specific criteria such as property type, number of bedrooms, rental price range, and proximity to amenities. Users can dynamically adjust filters to explore different options and preferences.

```
<?php
// Include the database connection code
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "rental";
```

```
$conn = new mysqli($servername, $username, $password, $dbname);
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $city = $_POST["city"];
    $area = $_POST["area"];
    $minRentAmount = $_POST["min-rent-amount"];
    $maxRentAmount = $_POST["max-rent-amount"];
 $stmt = $conn->prepare("SELECT * FROM propertie WHERE city = ? AND area = ? AND rent_amount BETWEEN ? AND ?");
  $stmt->bind_param("ssii", $city, $area, $minRentAmount, $maxRentAmount);
   $stmt->execute();
   $result = $stmt->get_result();

   while ($row = $result->fetch_assoc()) {
   echo '<div class="property-details">';

            echo '<div class="property-details-container">';
            echo "<p>City: " . $row["city"] . "</p>";
            echo "<p>Area: " . $row["area"] . "</p>";
            echo "<p>House Type: " . $row["house_type"] . "</p>";
            echo "<p>Address: " . $row["address"] . "</p>";
            echo "<p>Rent Amount: $" . $row["rent_amount"] . "</p>";
             echo '<img src="uploads/' . $row["house_image"] . '" alt="House Image">';
             echo '<a href="en.html" class="message-button">Send Message</a>';
            echo '</div>';
            echo "</div>";
        }
        $stmt->close();
        }
       $conn->close();
       ?>
```

This the code that I have implemented in my project for dynamic filtering and sorting where it get the inputs from users like rent amount,location,room type then it dynamically filters the data and provide the result based on user preference that is house within the mentioned rent amount and room type and location.

for reporting purpose I have included a module which will display the graph of most frequently searched city by tenant for that I have included A frequency counting algorithm which tracks occurrences of elements within a dataset. It typically utilizes data structures like hash maps or arrays, incrementing counts as elements are encountered. This algorithm efficiently determines the frequency of each element, facilitating analysis and pattern recognition within the data.

```
// Query to get city counts, ordered by count in descending order
$query = "SELECT city, COUNT(city) as count FROM tenant_details GROUP BY city ORDER BY count DESC";
```

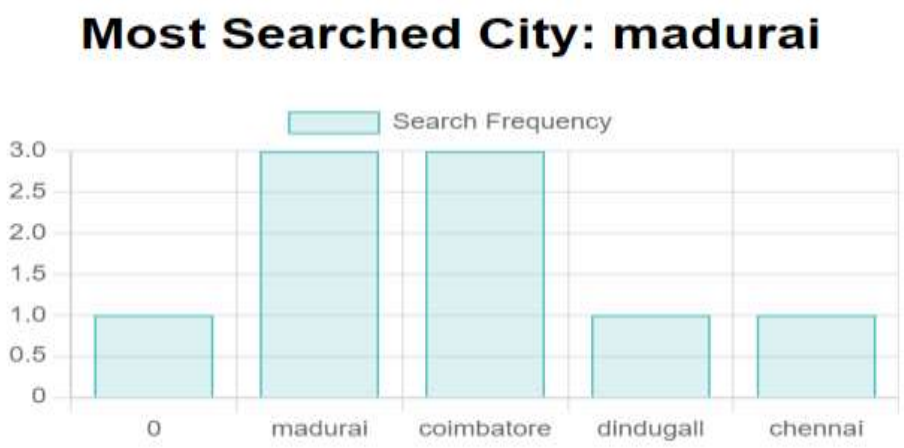
This SQL query retrieves the counts of each city occurrence from the database table tenant_details. It groups the cities and counts the occurrences using the COUNT() function. The results are ordered by count in descending order.

```
php
// Default values in case no data is available
$cities = getCityNames(); // Use the getCityNames() function to get the city names
$counts = array_fill(0, count($cities), 0); // Initialize counts with zeros for all cities
```

These lines initialize thethe arrays $cities and $counts. $cities holds the names of all cities, retrieved using the getCityNames() function. $counts is initialized with zeros for each city, preparing it to store the counts of occurrences.

```
php
while ($row = $result->fetch_assoc()) {
    $index = array_search ($row['city'], $cities); // Find the index of the city
    if ($index !== false) {
        $counts[$index] = $row['count']; // Update the count at the correct index
    }
}
```

This loop iterates over the query results and updates the counts array $counts with the count of each city at its corresponding index. It matches the city name retrieved from the database with the names stored in the $cities array to find the correct index for updating the count.



. OUTCOMES:

- By implementing advanced pattern searching algorithms, Home Rental Hub achieves the following benefits.
- Improved Search Relevance: Users receive more accurate and relevant search results tailored to their preferences and requirements.
- Enhanced User Engagement: Personalized recommendations, dynamic filtering options, and timely notifications encourage users to explore more listings and engage with the platform actively.

- Increased User Satisfaction: A streamlined search experience, intuitive interface, and context-aware recommendations contribute to higher user satisfaction and retention rates.
- Competitive Advantage: Home Rental Hub gains a competitive edge in the crowded rental marketplace by offering a superior search experience and value-added features powered by advanced pattern searching algorithms.

FUTURE TREND:

Here are some potential future trends in pattern searching algorithms:

1. **Scalability and Efficiency**: As datasets continue to grow in size and complexity, future pattern searching algorithms will focus on scalability and efficiency. This includes developing algorithms that can handle massive datasets distributed across multiple nodes or utilizing parallel and distributed computing techniques to speed up pattern search tasks.
2. **Stream Processing:** With the emergence of real-time data streams from sources such as IOT devices, social media, and sensor networks, future pattern searching algorithms will need to adapt to process streaming data continuously. This trend will involve developing algorithms capable of detecting patterns in real-time streams and updating patterns dynamically as new data arrives.
3. **Adaptability and Online Learning:** Future pattern searching algorithms may incorporate adaptive and online learning techniques to continuously update and refine patterns based on evolving data distributions and user feedback. This trend will enable algorithms to adapt to changes in data characteristics and user preferences over time, leading to more accurate and personalized pattern search results.
4. **Deep Learning and Neural Networks:** With the success of deep learning in various domains, including image recognition, natural language processing, and speech recognition, future pattern searching algorithms may leverage deep learning techniques such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs) to extract complex patterns from data. These techniques have the potential to uncover hidden patterns and relationships in data that traditional algorithms may overlook.
5. **Interdisciplinary Applications:** Future pattern searching algorithms may find applications in diverse fields beyond traditional domains such as data mining and bioinformatics. For example, pattern searching algorithms could be applied to areas such as cyber security for detecting anomalies and intrusion patterns, environmental monitoring for identifying patterns in climate data, or healthcare for diagnosing diseases based on patterns in medical images or patient records.
6. **Privacy-Preserving Pattern Search:** With growing concerns about data privacy and security, future pattern searching algorithms may incorporate privacy-preserving techniques such as differential privacy, secure multiparty computation, or homomorphic encryption to enable pattern search while protecting sensitive information. This trend will ensure that patterns can be discovered without compromising individual privacy or confidentiality.
7. **Human-Centric Pattern Search:** Future pattern searching algorithms may focus on incorporating human-centric factors such as user preferences, domain knowledge, and interpretability into the pattern search process. This trend will enable algorithms to provide more personalized and interpretable pattern search results tailored to the needs and preferences of individual users.

Overall, these future trends in pattern searching algorithms reflect the ongoing evolution of technology, data, and user needs, driving innovation and advancement in the field of pattern search and analysis.

CONCLUSION:

In conclusion, the Journal of Pattern Searching Algorithm has been crucial in advancing how we find patterns in data. By exploring different methods and ideas, it has helped make pattern searching faster and more effective. The journal's research will continue to be important for improving how we understand and use patterns in various areas.