**UNIT-V**

**Syllabus**:

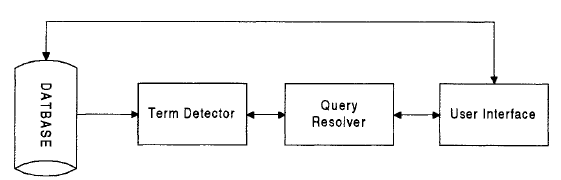
Text Search Algorithms: Introduction, Software text search algorithms, Hardware text search algorithms, Information system Evaluation: Introduction, Measures used in system evaluation, Measurement examples-TREC results

**Introduction:**

Three classical techniques for text retrieval techniques have been defined for organizing items in a textual database, for rapidly identifying the relevant items and for eliminating items that do not satisfy the search. The techniques are full text scanning (streaming), word inversion and multi-attribute retrieval. In addition to using the indexes as a mechanism for searching text in information systems, streaming of text was frequently found in the systems as an additional search mechanism. Streaming of text is a sequential search of the text. This technique is used to complete a query by searching for query terms that could not be satisfied by the index (e.g., imbedded search terms). It is also frequently used to locate the search terms for highlighting in the retrieved item prior to display.

**Introduction to Text Search Techniques:**

The basic concept of a text scanning system is the ability for one or more users to enter queries with the text of the items to be searched sequentially accessed and compared to the query terms. When all of the text has been accessed, the query is complete. One advantage of this type architecture is that as soon as an item is identified as satisfying a query, the results can be presented to the user for retrieval. Figure 9.1 provides a general diagram of a text streaming search system. The database contains the full text of the items. The term detector is the special





Hardware/software that contains all of the search terms and in some systems the logic between the terms. It inputs the text and detects the existence of the search terms. It outputs to the query resolver the detected terms, allowing for final logical processing of a query against an item. The query resolver performs two major functions: accepting search statements from the users and extracting the logic and search terms to pass to the detector. It also accepts results from the detector and determines which queries are satisfied by the item and possibly the relevance weight associated with hit. The query resolver passes information to the user interface, allowing it to continually update the search status and, on request, retrieve any items that satisfy the user search statement. The text streaming process is focused on finding at least one or all occurrences of a pattern of text (query term) in a text stream. It is assumed that the same alphabet is used in both search terms and text being streamed. In foreign language streamers, different encodings may have to be available for items from the same language (e.g., in Cyrillic there are over six encodings that can be used). The worst case search for a pattern of m characters in a string of n characters is at least n - m + 1 or a magnitude of O (n). Some of the original brute force methods could require *O (n\*m)* symbol comparisons (Sedgewick-88). More recent improvements have reduced the time to O (n + m).

Examples where index searches may not be able to satisfy the complete search statement are:

* Search for stop words
* Search for exact matches when stemming is performed
* Search for terms that contain both leading and trailing "don't cares"
* Search for symbols that are on the interword symbol list (e.g., ," ;)
* Search for "fuzzy" search terms

The major disadvantage of basing the search on streaming the text is the dependency of the search on the slowest module in the computer (the I/O module). Inversions/index’s gain their speed by minimizing the amount of data to be retrieved and provide the best ratio between the total numbers of items delivered to the user versus the total number of items retrieved in response to a query. But unlike inversion systems that can require storage overheads of 50 per cent to 300 per cent of the original databases, the full text search function does not require any additional storage overhead. There is also the advantage that items that satisfy the query may be returned to the user as soon as found.

Many of the hardware and software text searchers use finite state automata as a basis for their search algorithms. A finite state automaton is a logical machine that is composed of five elements:

I - a set of input symbols from the alphabet supported by the automata

S - a set of possible states

P - a set of productions that define the next state based upon the current state and input symbol

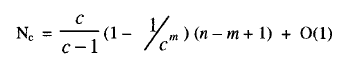
So - a special state called the initial state

SF - a set of one or more final states from the set S

**Software Text Search Algorithms:**

In software streaming techniques, the item to be searched is read into memory, and then the algorithm is applied. Although nothing in the described architecture prohibits software streaming from being applied to many simultaneous searches against the same item, it is more frequently used to resolve a particular search against a particular item. There are four major algorithms associated with software text search: the brute force approach, Knuth-Morris-Pratt, Boyer-Moore, Shift-OR algorithm, and Rabin-Karp. Of all of the algorithms, Boyer-Moore has been the fastest, requiring at most O (n + m) comparisons where n is the number ***of*** characters being searched and mis the size of the search string. Knuth- Pratt-Morris and Boyer-Moore both require O (n) preprocessing of search strings in addition to the search comparisons

The brute force approach is the simplest string matching algorithm. The idea is to match the search string against the input text. Whenever a mismatch is detected in the comparison process, the input text is shifted one position, and the comparison process is initialized and restarted. The expected number of comparisons when searching an input text string of n characters for a pattern of m characters is

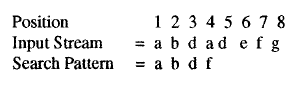


Where N, is the expected number of comparisons and c is the size of the alphabet for the text.

Applying the formula to an example where the alphabet is c=25 characters, the search pattern is m=7 characters and the search stream is an item with n=30,000 characters have the number of comparisons is:



The Knuth-Pratt-Morris algorithm made a major improvement in previous algorithms in that even in the worst case it does not depend upon the length of the search term and does not require comparisons for every character in the input stream. The basic concept behind the algorithm is that whenever a mismatch is detected, the previous matched characters define the number of characters that can be skipped in the input stream prior to starting the comparison process again. For example consider:



When the mismatch occurs in position 4 with a "f" in the pattern and a "c" in the input stream, a brute force approach may shift just one position in the input text and restart the comparison. But since the first three positions of the pattern matched (a b d), then shifting one position cannot find an "a" because it has already been identified as a "b." The algorithm allows the comparison to jump at least tile three positions associated with the recognized "a b d". Since the mismatch on the position could be the beginning of the search string, four positions cannot be skipped. To know the number of positions to jump based upon a mismatch in the search pattern, the search pattern is pre-processed to define a number of characters to be jumped for each position. The Shift Table that specifies the number of places to jump given a mismatch is shown in Figure 9.3 for a search pattern = abcabcacab. In tile table it should be noted that the alignment is primarily based on aligning over the repeats of the letters "a" and "ab." The Figure below provides an example application of the algorithm where S is the search pattern and I is the input text stream.

Boyer-Moore recognized that the string algorithm could be significantly enhanced if the comparison process starts at the end of the search pattern, processing right to left versus the start of the search pattern. The advantage is that

