**Mini Project**

**AHO-Corasick Algorithm for Pattern Searching**

**TEAM MEMBERS:**

1. Diya Maheshwari (9917102119) – E4
2. Mayank Tale(9917103228)-F7
3. Prakhar Khare (9917102102)-E4

**Introduction:**

Manipulating strings and searching for patterns in them are fundamental tasks in data science, and a typical task for any programmer. Efficient string algorithms play an important role in many data science processes. Often they are what make such processes feasible enough for practical use. For looking multiple patterns in a text efficiently, we use Aho-Corasick algorithm.

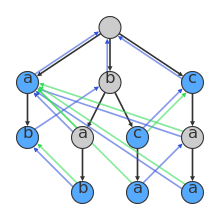
For Exp, Taking a dictionary with the 1,000 most common English words as patterns and using it to search the English version of Tolstoy’s “War and Peace” would take quite a while. The book is over three million characters long. If we take the 10,000 most common English words, the algorithm will work around 10 times slower. Obviously on inputs greater than this one, execution time will also grow. This is where the Aho-Corasick algorithm does its magic.

In computer science, the **Aho–Corasick algorithm** is a string-searching algorithm invented by [Alfred V. Aho](https://en.wikipedia.org/wiki/Alfred_V._Aho) and Margaret J. Corasick.[[1]](https://en.wikipedia.org/wiki/Aho%E2%80%93Corasick_algorithm#cite_note-1) It is a kind of dictionary-matching algorithm that locates elements of a finite set of strings (the "dictionary") within an input text. It matches all strings simultaneously. The [complexity](https://en.wikipedia.org/wiki/Time_complexity) of the algorithm is linear in the length of the strings plus the length of the searched text plus the number of output matches. Note that because all matches are found, there can be a quadratic number of matches if every substring matches (e.g. dictionary = a, aa, aaa, aaaa and input string is aaaa).

Informally, the algorithm constructs a [finite-state machine](https://en.wikipedia.org/wiki/Finite-state_machine) that resembles a [trie](https://en.wikipedia.org/wiki/Trie) with additional links between the various internal nodes. These extra internal links allow fast transitions between failed string matches (e.g. a search for cat in a trie that does not contain cat, but contains cart, and thus would fail at the node prefixed by ca), to other branches of the trie that share a common prefix (e.g., in the previous case, a branch for attribute might be the best lateral transition). This allows the automaton to transition between string matches without the need for backtracking.

When the string dictionary is known in advance (e.g. a [computer virus](https://en.wikipedia.org/wiki/Computer_virus) database), the construction of the automaton can be performed once off-line and the compiled automaton stored for later use. In this case, its run time is linear in the length of the input plus the number of matched entries.

The Aho–Corasick string-matching algorithm formed the basis of the original [Unix command](https://en.wikipedia.org/wiki/List_of_Unix_commands) [fgrep](https://en.wikipedia.org/wiki/Grep#Variations).



TIME COMPLEXITY

Suppose there are **M** patterns of lengths **L1**, **L2**, …, **Lm**. We need to find all matches of patterns from a dictionary in a text of length **N**.

A trivial solution would be taking any algorithm from the first part and running it **M** times. We have complexity of **O(N + L1 + N + L2 + … + N + Lm)**, i.e. **O(M \* N + L)**.

Any serious enough test kills this algorithm.

Taking a dictionary with the 1,000 most common English words as patterns and using it to search the English version of Tolstoy’s “War and Peace” would take quite a while. The book is over three million characters long.

If we take the 10,000 most common English words, the algorithm will work around 10 times slower. Obviously on inputs greater than this one, execution time will also grow.

This is where the Aho-Corasick algorithm does its magic.

The complexity of the Aho-Corasick algorithm is **O(N + L + Z)**, where **Z** is the count of matches. This algorithm was invented by Alfred V. Aho and Margaret J. Corasick in 1975.

REFERENCES.

1. https://en.wikipedia.org/wiki/Aho%E2%80%93Corasick\_algorithm.
2. https://www.toptal.com/algorithms/aho-corasick-algorithm.