**Introduction**

Pattern search operations are important for searching for a pattern within a text file. There are many algorithms to search for a substring, such as Brute Force, KMP, Boyer Moore, Rabin Karp. These algorithms have their own benefits and drawbacks.

In my opinion, the Boyer Moore or KMP will be the best algorithm and the Brute Force will be the worst algorithm for this situation. Since Brute Force has MN time complexity and Boyer Moore has the complexity of 3N, where N is the length of text and M is the length of the pattern.

If we compare other algorithms, then KMP will perform better than Rabin Karp as KMP uses DFA for not backing up the input and the time complexity is 2N. Whereas Rabin Karp uses hashing for generating the hash code for the pattern and then search the given pattern and time complexity is 7N.

**Procedure**

The algorithms of substring search are mentioned in Algorithms book by Sedgewick. With the help of that, I wrote the code for Brute Force, KMP, Boyer Moore and Rabin Karp.

To find the running time for each algorithm I have taken nano time at the beginning of main function and end of the main function. Then I calculated the difference between two of the time. By doing this I can check which algorithm is working better for this particular problem.

**Analysis**

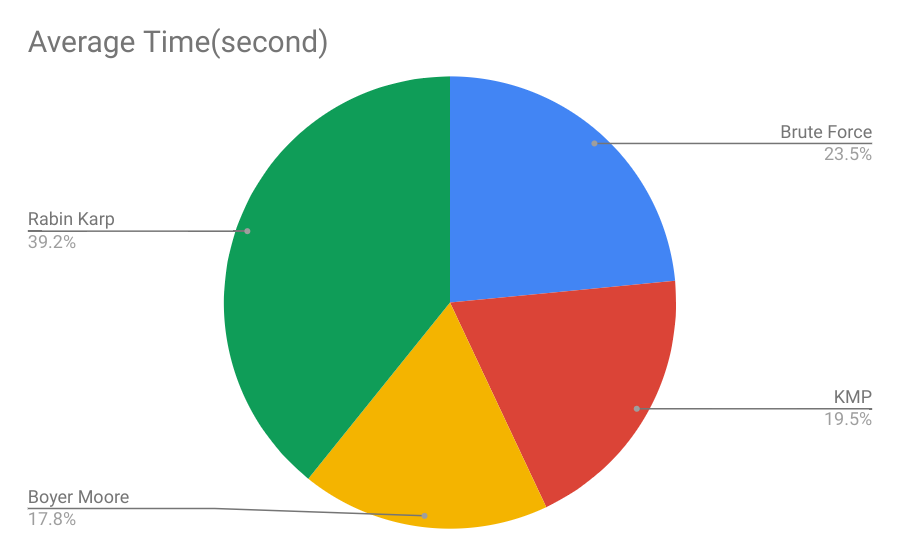
The analysis made by me is wrong, as the analysis made by me is that Brute Force will be the worst algorithm, but Rabin Karp is the algorithm which performs even worse than the Brute Force algorithm.

Boyer Moore: The best algorithm which performs best is Boyer Moore as it uses bad match table to which checks from right to left and it skips up to M characters when found mismatch which leads to approx N/M characters compares to search for a pattern of length M in a text of length N. It works efficiently on long pattern length as then the jump will get larger through text.

KMP: Then KMP performs better as it accesses each pattern character once when computing DFA and each text character once (in the worst case) in search method. It performs better than Brute Force because it never keeps the backup. This performs well for the small sets of the pattern. It is guaranteed linear time with no back up in the input.

Brute Force: The time complexity of Brute Force is O(MN). It works better than Rabin Karp because it doesn’t calculating any value for matching. It doesn’t require pre-processing (calculating the number of repetitions for each character in the pattern) of the text. It has quadratic running time.

Rabin Karp: The time complexity of Rabin Karp is O(M + N). Its drawback is that it has a very long inner loop (as we have to compute a hash code for each character). In this case it is slower than Brute Force as arithmetic operations are slower than char compares.



**Analysis of Algorithm for the searching pattern in Tale.txt**

**Conclusion**

To summarize there are four algorithms for the same task, each of them have their own appealing features. Based on the particular pattern and text, we have to decide which algorithm apt for our problem. As different algorithms work differently for different problems. Therefore, for this case, Boyer Moore is the best algorithm to search pattern.