String Matching Report

Boyer-Moore Implementation

The Boyer-Moore algorithm was implemented using the bad character rule. In this approach, a table is created to record the last occurrence of each character in the pattern, and the comparison between the pattern and the text proceeds from right to left. When a mismatch occurs, the algorithm determines how far the pattern can be shifted by consulting the bad character table, allowing it to skip unnecessary comparisons. Special cases such as empty patterns or patterns longer than the text were handled explicitly to ensure stability. Although the algorithm showed higher runtime in some practical scenarios due to overhead and input characteristics, the implementation produced correct results for all test cases.

GoCrazy Algorithm

GoCrazy is a hybrid algorithm selection method designed to adapt to different pattern sizes. Instead of implementing a new matching algorithm, it chooses among existing ones. For short patterns, it uses the Naive algorithm, for medium patterns it uses KMP, and for longer patterns it selects Boyer-Moore. This structure provides flexibility and good overall performance across a range of inputs. GoCrazy passed all tests and demonstrated competitive performance in many cases without adding unnecessary complexity to the project.

Pre-Analysis Strategy

The StudentPreAnalysis class uses a simple but effective decision process to select the most suitable algorithm before matching begins. If the text or pattern is very short, Naive is chosen due to its low overhead. When the pattern contains strong prefix repetition, KMP is selected because it handles such cases efficiently. Very long texts combined with long patterns lead to the choice of Boyer-Moore, while Rabin-Karp is preferred in moderately long cases where hashing can be advantageous. For all remaining scenarios, GoCrazy is used as a balanced fallback. The aim of this strategy is not to predict the fastest algorithm in every instance, but to make reasonable, explainable decisions with minimal computation cost.

Results Analysis

All algorithms passed every test in the provided environment, confirming both correctness and stability. Naive, KMP, Rabin-Karp, Boyer-Moore, and GoCrazy each completed all cases successfully. The Pre-Analysis module also produced valid results and functioned as expected. Boyer-Moore was slower in several scenarios, which is natural for the given implementation and language overhead, but accuracy remained guaranteed throughout.

My Research

During my work on this assignment, I reviewed online explanations of Boyer-Moore, KMP, and general string matching theory to refresh and validate my understanding. I also used ChatGPT to

clarify certain ideas, explore alternative strategy options, and verify that the reasoning behind my approach was consistent. I did not copy any external code and wrote all implementations myself while adapting them to the project structure. My research focused on understanding the strengths and limitations of each algorithm and how they behave in different input conditions.

My Journey

Throughout this assignment, I learned how various string matching algorithms perform under different circumstances and how important algorithm selection can be in real scenarios. Implementing Boyer-Moore and ensuring that it behaved correctly for all test cases was the most challenging part, while developing the pre-analysis strategy helped me better understand the relationship between pattern structure and algorithm performance. The homework was informative and well designed, and it helped me improve both my algorithmic thinking and practical coding skills.

Emirhan Akbaş - 23050151001