Queens College of CUNY
Department of Computer Science
Design and Analysis of Algorithms (CSCI 323)
June 2024

Assignment #10 "Substring Search Algorithms"

Due: June 24, 2024

Overview:

In this assignment we compare the empirical performance of several standard substring-search (string-matching) algorithms, using the template from previous assignments.

As in previous assignments, it is critical that all the algorithms have the same method (function) signatures. In this case, the signature will be *search_name*(*text*, *pattern*, *verbose*) The *verbose* parameter - when set to True - will tell the function to print additional information for illustration and debugging purposes. Allow the function itself to determine the respective lengths of the text and pattern. Each function should return the index of the substring if found, or -1 if not found.

Submissions:

In the Google form, please submit the following"

- Assignment10.py (source code)
- Assignment10.txt (console output, including both timings and pattern-location tables)
- Assignment10.png (bar graph with timings)

Tasks:

[0] Please consult previous assignments for a program template and structure, general guidelines about best practices, as well as sources of and policies concerning using public code. Wherever possible, import previous assignments and invoke their functions, rather than just copying code.

[1] Define functions that implement these sub-string search algorithms:
- Native Search that wraps the built-in string-search capability of your programming language
- Brute Force - see https://www.geeksforgeeks.org/naive-algorithm-for-pattern-searching/
- Rabin-Karp - see https://www.geeksforgeeks.org/rabin-karp-algorithm-for-pattern-searching
- Rabin-Karp Randomized - (use multiple hash functions with random moduli)
- Knuth-Morris-Pratt - see https://www.geeksforgeeks.org/kmp-algorithm-for-pattern-searching
- Boyer-Moore - see https://www.geeksforgeeks.org/boyer-moore-algorithm-for-pattern-searching

[2] Read in files Assignment07-Text.txt and Assignment07-Patterns.txt as inputs. Convert the content to upper-case to avoid case-sensitivity issues.

[3] Modify the existing code to iterate over a list of patterns.

[4] Modify the code to look for random patterns of different lengths. Use a method similar to how we chose the key in Assignment #1 - Search Algorithms

[5] Display results from the various searches in a text-table to verify that they work.

[6] Display the timings from the various searches in the dataframe and graph as in earlier assignments.