String Search II

Ben Thomas

Lyndsey Cairone

Robert Fitzgerald

Topics to be Covered

String Search II Algorithms

- What is String Search
- Knuth Morris Pratt Algorithm
- Boyer Moore Algorithm

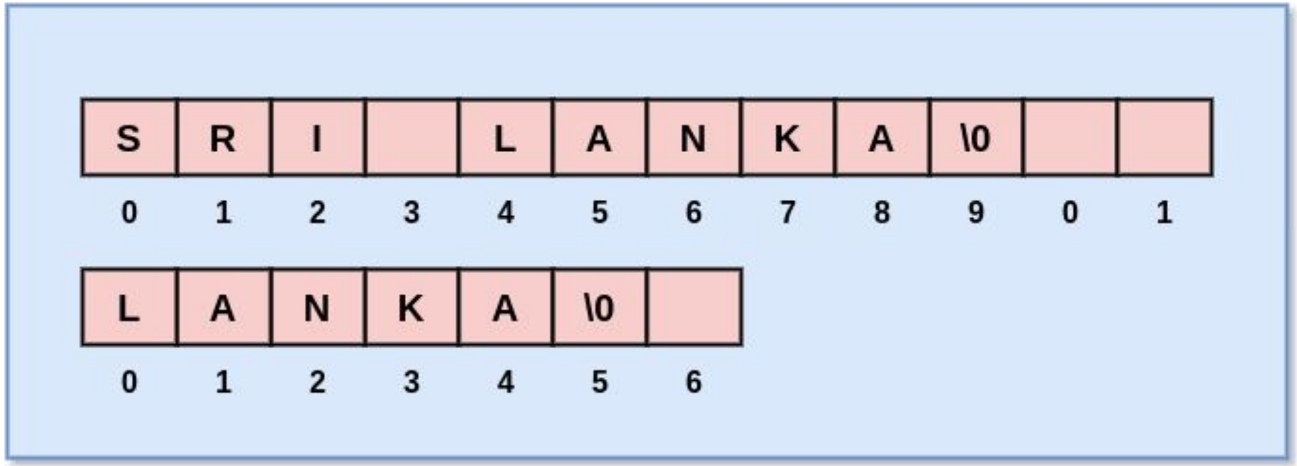
Applications of String Search

- Real World Applications
- Our Implementation of the Algorithms
- Benchmarking the Algorithms

What is String Search?

Basic Explanation:

Algorithms used to find the occurrences of a smaller string within a larger string.



Knuth Morris Pratt Algorithm

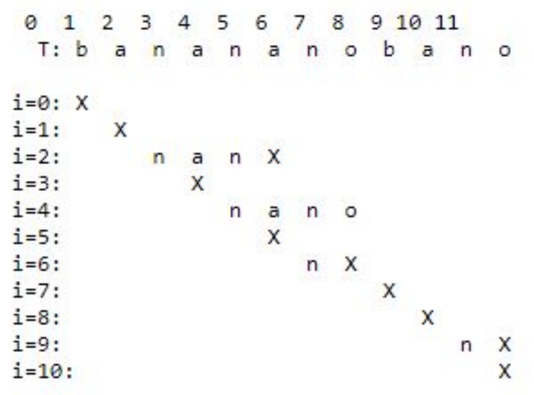
-Developed in 1970 and published in 1977 by James H. Morris, Donald Knuth, and Vaughan Pratt.

-This takes O(n + m) time complexity as preprocessing is in O(m) space and time complexity.

-The searched string travels across the text from left to right, being compared left to right, until the pattern is found.

Knuth Morris Pratt Algorithm Example

In this example, the word “nano” is being compared by the text “banananobano”



```
  0  1  2  3  4  5  6  7  8  9 10 11
   T: b  a  n  a  n  a  n  o  b  a  n  o

i=0:  X
i=1:     X
i=2:        n  a  n  X
i=3:           X
i=4:              n  a  n  o
i=5:                 X
i=6:                    n  X
i=7:                          X
i=8:                             X
i=9:                                n  X
i=10:                                  X
```

The algorithm checks the characters from left to right and if the characters match up, the algorithm continues to check for the rest of the searched string in the text.

As shown in the example, some of these comparisons are not required such as the comparison in i = 4 as it is known that T[4] = n

Boyer Moore Algorithm

-Developed in 1977 by Professor Robert Stephen Boyer and J Strother Moore.

-A search string algorithm that runs faster as the pattern length increases.

-Creates a shifting table of keys of the string you are searching for from tail to head so that it can skip along the larger string in jumps of multiple characters rather than searching through every character.

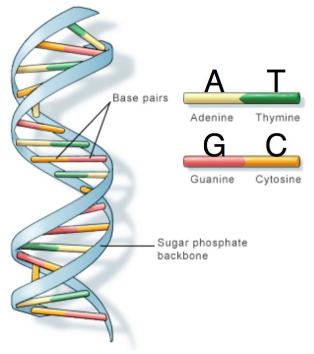
Boyer Moore Algorithm Example

If the string one is search for is HELLO the shifting table would be as follows:

Character	H	E	L	L	O
Shift	4	3	1	1	0

The algorithm uses this as reference and each time it reaches an H in the larger string it will shift down four characters and check the small string from tail to head. For an E it would shift 3 and so on…
Since there are two L's next to each other it does not know if it is the second or the first L so it can only shift one position.

Real World Applications



Bioinformatics

Search Engines

Detecting Spam Emails

Cipher texts(encoding/decoding)

Plagiarism detector

Information Retrieval

Our Implementation of the Algorithms

-We researched and implemented the Knuth Morris Pratt algorithm and the Boyer Moore algorithm into a program which will measure the time it takes for both algorithm ro run.

-We implemented a benchmarking function to gauge which algorithm runs the fastest when utilizing the same inputs.

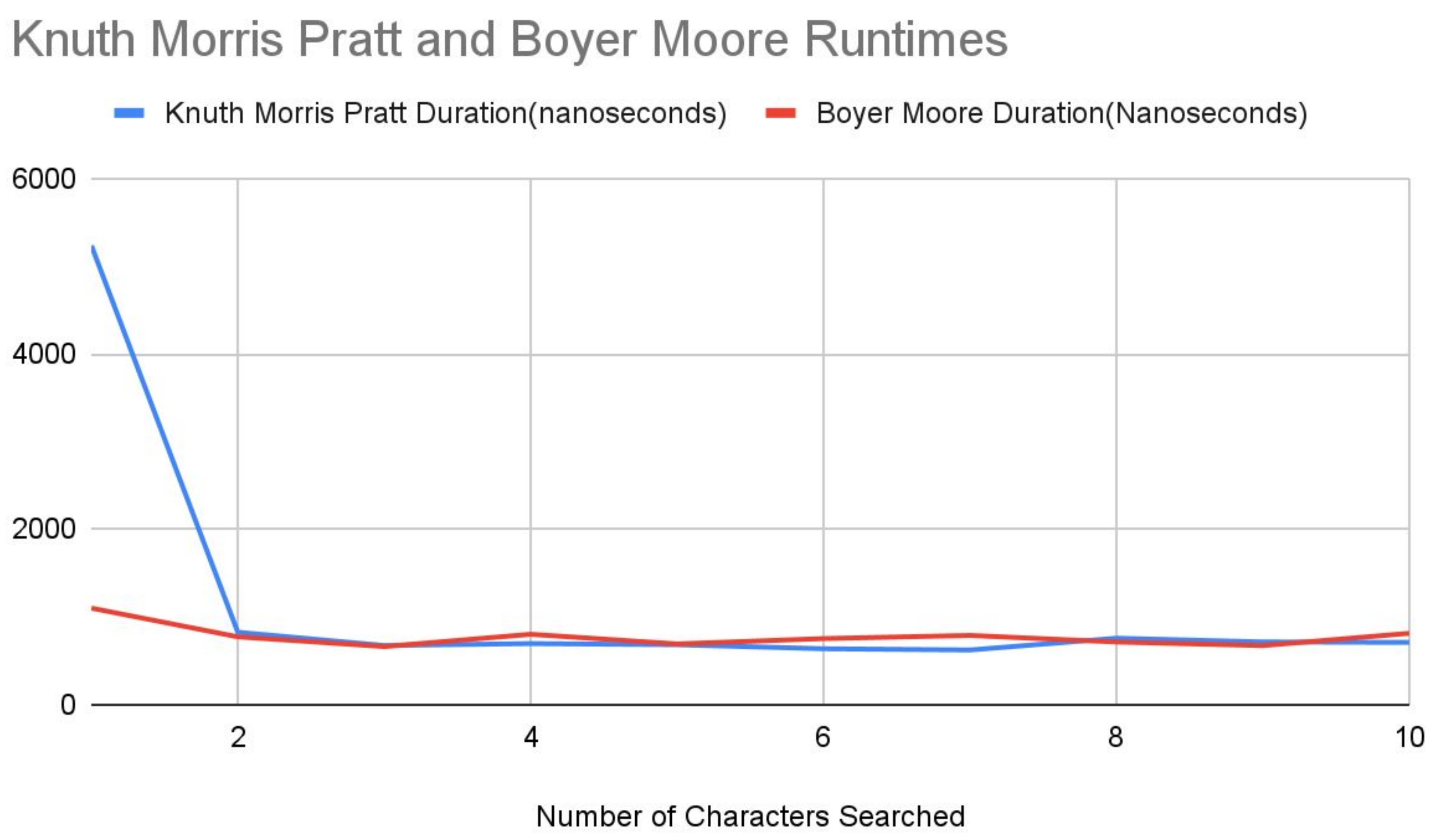
Implementation

https://ide.cs50.io/638088dd6cbf4d619197fb1ce8fa440b

Benchmarking our Algorithm

-We tested both algorithms with the same text and string to look for and output the results to a new file to compare the two runtimes.

-We increased the string to search for by one character each time for both algorithms. And recorded the runtime.


Knuth Morris Pratt and Boyer Moore Runtimes
Knuth Morris Pratt Duration(nanoseconds)
Boyer Moore Duration(Nanoseconds)
6000
4000
2000
0
2
4
6
8
10
Number of Characters Searched

Conclusion

-The Knuth Morris Pratt Algorithm has a much larger runtime when searching for one character than the Boyer Moore algorithm.

-As the amount of characters to search for increases, the runtimes stay about the same between 661-917 nanoseconds.

-The Boyer Moore algorithm is more efficient when only searching for one character. Otherwise they are relatively the same.