Atakan SEVİNÇLİ



CMPE 343 - Homework 5 - Part I

The table at the bottom of the page summarizes the algorithms we discussed for searching for substrings. - of them has appealing features, as is often the case when we have many algorithms for the same job. Brute-force search is easy to implement and works well in typical cases (Java's indexOf() method in String uses brute-force search); Knuth-Morris-Pratt is guaranteed linear-time with no backup in the input; Boyer-Moore is sublinear (by a factor of M) in typical situations. Each has disadvantages as well: brute-force can require time proportional to MN; extra space is used by Knuth-Morris-Pratt and Boyer-Moore. In the table below, these features are summarized.

algorithm	version	operation count		backup	correct?	extra
		guarantee	typical	in input?	correcti	space
brute force	_	MN	1.1 N	yes	yes	1
Knuth-Morris-Pratt	full algorithm	3N	N/M	yes	yes	R
Boyer-Moore	mismatched char heuristic only (Algorithm 5.7)	MN	N/M	yes	yes	R

Boyer-Moore's approach is to try to match the last character of the pattern instead of the first one with the assumption that if there is not match at the end no need to try to match at the beginning. This allows for "big jumps" therefore **BM** works better when the pattern and the text you are searching resemble "natural text"

Knuth-Morris-Pratt searches for occurrences of a "word" W within a main "text string" S by employing the observation that when a mismatch occurs, the word itself embodies sufficient information to determine where the next match could begin, thus bypassing re-examination of previously matched characters.

This means KMP is better suited for small sets like DNA (ACTG)

Differences between Boyer Moore and Brute Force

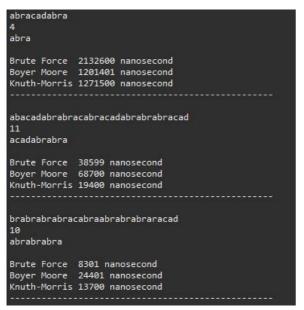
Table1: Performance comparison of Boyer-Moore and Brute-Force algorithm.

Table2: Behaviors of Boyer-Moore and Brute-Force algorithm.

Input Scale	Boyer-Moore	Brute-Force
n=10, m=2	3,636,899	10,692,715
n=100, m= 2	4,074,790	14,627,957
n=100, m= 10	3,718,222	15,314,766
n=100, m= 50	3,591,666	15,824,537
n=1000, m= 50	5,121,397	32,428,455
n=1000, m=200	4,566,272	32,886,075

Behaviors	Boyer-Moore	Brute-Force	
Order of			
comparison	Right to left	Left to right	
	Both bad		
	character rule	One by one	
Shifting rules	and good	character shift	
	suffix rule		
Preprocessing			
time	O(m+n)	No pre-	
complexity		processing	

Table 1 Table 2



In the present arena to find the exact content in minimum time is the most essential factor. String algorithms play a crucial role in this regard. A lot of research is going on at the level of software and hardware to make pattern searching faster. By the application of various algorithms in diverse fields the approximate best algorithm can be ascertained. It has been found that most applications uses Boyer Moore algorithm for its desirable and efficient work and also most of other applications uses the basics of this algorithm as it has minimum time complexity.