

Лабораторная работа №1

Реализация алгоритмов поиска по тексту

Цель работы: научиться реализовывать на выбранном языке программирования алгоритмы поиска по тексту: прямой поиск; алгоритм Кнута, Морриса и Пратта; алгоритм Бойера-Мура.

Краткие теоретические сведения

Прямой поиск

Основная идея алгоритма прямым поиском заключается в посимвольном сравнении строки с подстрокой. В начальный момент происходит сравнение первого символа строки с первым символом подстроки, второго символа строки со вторым символом подстроки и т. д. Если произошло совпадение всех символов, то фиксируется факт нахождения подстроки. В противном случае производится сдвиг подстроки на одну позицию вправо и повторяется посимвольное сравнение (рис. 1). Символы, которые сравниваются, на рисунке выделены жирным. Рассматриваемые сдвиги подстроки повторяются до тех пор, пока конец подстроки не достиг конца строки или не произошло полное совпадение символов подстроки со строкой, то есть найдется подстрока.

	$i \rightarrow i \rightarrow i \rightarrow i \rightarrow i \rightarrow i \rightarrow i$													
	↓	↓	↓	↓	↓	↓	↓	↓						
Строка	А	В	С	А	В	С	А	А	В	С	А	В	D	
Подстрока	А	В	С	А	В	D								
		А	В	С	А	В	D							
			А	В	С	А	В	D						
				А	В	С	А	В	D					
					А	В	С	А	В	D				
						А	В	С	А	В	D			
							А	В	С	А	В	D		

Рис.1. Демонстрация алгоритма прямого поиска

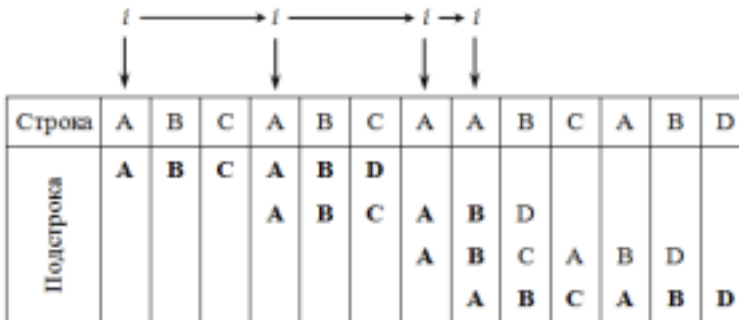
Данный алгоритм является малозатратным и не нуждается в предварительной обработке и в дополнительном пространстве. Большинство сравнений алгоритма прямого поиска являются лишними.

Поэтому в худшем случае алгоритм будет малоэффективен, так как его сложность будет пропорциональна $O((n-m+1)*m)$, где n и m – длины строки и подстроки соответственно.

Алгоритм Кнута, Морриса и Пратта

Основным отличием алгоритма Кнута, Морриса и Пратта от алгоритма прямого поиска заключается в том, что сдвиг подстроки выполняется не на один символ на каждом шаге алгоритма, а на некоторое переменное количество символов. Следовательно, перед тем как осуществлять очередной сдвиг, необходимо определить величину сдвига. Для повышения эффективности алгоритма необходимо, чтобы сдвиг на каждом шаге был бы как можно большим (рис. 2). На рисунке символы, подвергшиеся сравнению, выделены жирным шрифтом.

Если для произвольной подстроки определить все ее начала, одновременно являющиеся ее концами, и выбрать из них самую длинную (не считая, конечно, саму строку), то такую процедуру принято называть префикс-функцией. В реализации алгоритма Кнута, Морриса и Пратта используется предобработка искомой подстроки, которая заключается в создании префикс-функции на ее основе. При этом используется следующая идея: если префикс (он же суффикс) строки длиной i длиннее одного символа, то он одновременно и префикс подстроки длиной $i-1$. Таким образом, проверяем префикс предыдущей подстроки, если же тот не подходит, то префикс ее префикса, и т.д. Действуя так, находим наибольший искомый префикс.



Строка	А	В	С	А	В	С	А	А	В	С	А	В	D
Подстрока	А	В	С	А	В	С	А	В	С	А	В	D	

Рис. 2. Демонстрация алгоритма Кнута, Морриса и Пратта

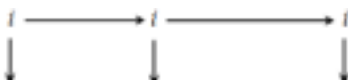
Точный анализ рассматриваемого алгоритма весьма сложен. Д. Кнут, Д. Моррис и В. Пратт доказывают, что для данного алгоритма требуется порядка $O(m+n)$ сравнений символов (где n и m – длины строки и подстроки соответственно), что значительно лучше, чем при прямом поиске.

Алгоритм Бойера и Мура

Существует множество вариаций алгоритма Бойера и Мура, рассмотрим простейший из них, который состоит из следующих шагов. Первоначально строится таблица смещений для искомой подстроки. Далее идет совмещение начала строки и подстроки и начинается проверка с последнего символа подстроки. Если последний символ подстроки и соответствующий ему при наложении символ строки не совпадают, подстрока сдвигается относительно строки на величину, полученную из таблицы смещений, и снова проводится сравнение, начиная с последнего символа подстроки. Если же символы совпадают, производится сравнение предпоследнего символа подстроки и т.д. Если все символы подстроки совпали с наложенными символами строки, значит, найдена подстрока и поиск окончен. Если же какой-то (не последний) символ подстроки не совпадает с соответствующим символом строки, далее производим сдвиг подстроки на один символ вправо и снова начинаем проверку с последнего символа. Весь алгоритм выполняется до тех пор, пока либо не будет найдено вхождение искомой подстроки, либо не будет достигнут конец строки (рис. 3). На рисунке символы, подвергшиеся сравнению, выделены жирным шрифтом.

Величина сдвига в случае несовпадения последнего символа вычисляется, исходя из следующего: сдвиг подстроки должен быть минимальным, таким, чтобы не пропустить вхождение подстроки в строке. Если данный символ строки встречается в подстроке, то смещаем подстроку таким образом, чтобы символ строки совпал с самым правым вхождением этого символа в подстроку. Если же подстрока вообще не содержит этого символа, то сдвигаем подстроку на величину, равную ее длине, так что первый символ подстроки накладывается на следующий за проверявшимся символом строки.

Величина смещения для каждого символа подстроки зависит только от порядка символов в подстроке, поэтому смещения удобно вычислить заранее и хранить в виде одномерного массива, где каждому символу алфавита соответствует смещение относительно последнего символа подстроки.



Строка	А	В	С	А	F	D	F	А	В	С	А	В	D
Подстрока	А	В	С	А	B	D							
						А	В	С	А	В	D		
								А	В	С	А	В	D

Рис. 3. Демонстрация алгоритма Бойера и Мура

Таким образом, данный алгоритм является наиболее эффективным в обычных ситуациях, а его быстродействие повышается при увеличении подстроки или алфавита. В наихудшем случае трудоемкость рассматриваемого алгоритма $O(m+n)$.

Задание к лабораторной работе

Написать программу на выбранном языке программирования, реализующую описанные выше алгоритмы для поиска подстроки в строке. Программа должна запрашивать имя входного файла. Оценить трудоемкость рассматриваемых алгоритмов.

Контрольные вопросы

1. Что такое поиск подстроки в строке?
2. Принципы работы алгоритма прямого поиска.
3. Принципы работы алгоритма Кнута, Морриса и Пратта.
4. Принципы работы алгоритма Бойера и Мура.
5. Как оценить трудоемкость алгоритма поиска по тексту?
6. Каким образом задается величина сдвига в алгоритме Кнута, Морриса и Пратта?
7. Каким образом задается величина сдвига в алгоритме Бойера и Мура?