МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Пермский государственный национальный исследовательский университет»

Колледж профессионального образования

ТЕОРИЯ АЛГОРИТМОВ

Методические рекомендации
для практических работ по изучению дисциплины
для студентов Колледжа профессионального образования
специальности
09.02.03 Программирование в компьютерных системах

Утверждено на заседании ПЦК
Информационных технологий
Протокол № 9 от 23.05.2018
председатель _________Н.А. Серебрякова

Пермь 2018

Составитель:

Ежова Марина Алексеевна, преподаватель высшей квалификационной категории, преподаватель ПГНИУ

Теория алгоритмов: методические указания по практической работе для студентов Колледжа профессионального образования по специальности 09.02.03 Программирование в компьютерных системах / сост. М.А. Ежова; Колледж проф. образ. ПГНИУ. – Пермь, 2017. – 14 с.

Методические указания «Теория алгоритмов» разработаны на основе требований Федерального государственного образовательного стандарта среднего профессионального образования по специальности 09.02.03 Программирование в компьютерных системах для оказания помощи студентам специальности 09.02.03 Программирование в компьютерных системах по дисциплине «Теория алгоритмов». Содержат типичные практические задания по всем разделам дисциплины.

Предназначены для студентов Колледжа профессионального образования ПГНИУ специальности 09.02.03 Программирование в компьютерных системах (СПО) всех форм обучения.

Печатается по решению педагогического совета Колледжа профессионального образования Пермского государственного национального исследовательского университета

СОДЕРЖАНИЕ

Нормальные алгоритмы Маркова

Задание 1. Применить к слову «21522116421» подстановку 21 → 1.

Решение:

В данном слове найдём первое схождение левой части подстановки «21»:

21522116421

Теперь заменим эти цифры на правую часть постановки «1»:

1522116421

Снова идя с начала слова, найдём следующее вхождение «21»:

152**21**16421

Заменяем:

152**1**16421

Опять начинаем с начала слова и ищем «21»:

15**21**16421

Заменяем:

15**1**16421

Ищем, применима ли подстановка еще раз:

151164**21**

Опять заменяем:

151164**1**

Далее подстановка не применима, так как в слове нет вхождения «21».

Итоговое слово: 1511641.

Задание 2. Алгоритм Маркова в алфавите A={a, b, c} задан системой ориентированных подстановок:

1) b → accc
2) ca → acc
3) aaa → ∧
4) ccc → ∧

Укажите множество итоговых слов, в которые преобразуются любые слова после применения к ним алгоритма.

Решение:

Очевидно, что невозможно перебрать всевозможные слова, построенные на основе данного алфавита.

Вариант 1. Исследуем множество этих слов, начиная с минимальных. Видим, что данный алгоритм не применим к словам: ∧, a, c, aa, ac, cc.

Эти слова входят в множество итоговых слов для этого алгоритма.

Теперь применим алгоритм к словам: b, ab, ba, bb, bc, ca, cb:

b	ab	ba	bb	bc	ca	cb
accc	aaccc	accca	acccb	acccc	acc	caccc
a	aa	accacc	acccaccc	ac		accccc
		acacccc	accaccccc			acc
		aacccccc	acaccccccc			
		aaccc	aaccccccccc			
		aa	aacccccc			
			aaccc			
			aa			

Теперь множество итоговых слов выглядит так: { ∧, a, c, aa, ac, cc, acc}.

Теперь рассмотрим трехбуквенные слова: aab, aac, aba, abb, abc, aca, acb. Слово «aaa» станет пустым словом, а слово «acc» - уже входит в множество итоговых слов.

aab	aac	aba	abb	abc	aca	acb
aaaccc		aaccca	aacccb	aacccc	aacc	acacc
ccc		aaccacc	aacccaccc	aac		aacccc
∧		aacacccc	aaccaccccc			aac
		aaacccccc	aacaccccccc			
		cccccc	aaaccccccccc			
		ccc	ccccccccc			
		∧	cccccc			
			ccc			
			∧			

Теперь множество итоговых слов выглядит так: { ∧, a, c, aa, ac, cc, aac, acc, aacc}.

Применяя алгоритм к любым другим словам, получим одно из уже указанных итоговых слов.

Вариант 2. Проанализируем алгоритм. Очевидно, что во множестве итоговых слов не может быть буквы «b» или сочетания «ca». То есть итоговые слова могут состоять только из букв «a» и «c». Причем, буквы «a» не может быть подряд трех и более, как и буквы «c». Перечислим все слова, удовлетворяющие этим условиям:

∧, a, c, aa, ac, cc, aac, acc, aacc

Примечание: Для уверенности данное задание нужно выполнить и первым и вторым способом.

Задание 3. Составить нормальный алгоритм Маркова на алфавите $A=\{a,b,c\}$. Заменить каждый символ в слове на его заглавную версию (например: $bacb \to BACB$). Проверить действие алгоритма на двух примерах.

Решение:

Данный тип заданий решается введением спецсимвола, который последовательно передвигается по слову слева направо. В данном случае спецсимвол «пепрыгивая» букву меняет её на заглавную.

Подстановка, создающая спецсимвол, должна распологаться в алгоритме последней.

$$\wedge \to +$$

Перед ней подстановка, удаляющая спецсимвол. Для уверенности эту подстановку делаем завершающей:

$$+ \Rightarrow \wedge$$

Примечание: также можно использовать обозначение: $+ \mapsto \wedge$

Теперь пропишем подстановки замены:

$$+a \to A+$$
$$+b \to B+$$
$$+c \to C+$$

Составим весь алгоритм:

$$+a \to A+$$
$$+b \to B+$$
$$+c \to C+$$

$+ \Rightarrow \wedge$

$\wedge \rightarrow +$

Проверим работу алгоритма на примерах «baba» и «cabca»:

baba	cabca
+baba	+cabca
B+aba	C+abca
BA+ba	CA+bca
BAB+a	CAB+ca
BABA+	CABC+a
BABA	CABCA+
	CABCA

Рекурсивные функции

Задание 1. Применить суперпозицию к функциям:
$f_1 = 2x$
$f_2 = 2y - 4$
$f_3 = z + 66t$
$F(a,b,c) = \sin(a) - c^2 + \ln(b)$

Решение:

Подставим вместо a – f_1, вместо b – f_2, вместо c – f_3:

$$F(f_1, f_2, f_3) = \sin(2x) - (z + 66t)^2 + \ln(2y - 4)$$

Задание 2. Применить простейшую рекурсию к функциям:

$g(x) = x$, $\psi(x,y,z) = z^x$ (то же, что и h(x,y,z))

Решение:

Исходя из определения простейшей рекурсии, вычислим несколько последовательных значений функции f(x, y) :

$$f(x,0) = x$$
$$f(x,1) = \psi(x, 0, f(x,0)) = \psi(x, 0, x) = x^x$$
$$f(x,2) = \psi(x,1, f(x,1)) = \psi(x,1, x^x) = (x^x)^x = x^{x^2}$$
$$f(x,3) = \psi(x,2, f(x,2)) = \psi\left(x, 2, x^{x^2}\right) = \left(x^{x^2}\right)^x = x^{x^3}$$

Сравнивая начало и конец каждой строчки, выведем общую формулу:

$$f(x,y) = x^{x^y}$$

Задание 3. Создать функции примитивной рекурсии для функции F(x, y) = (y + 1) x.

Решение:

Припишем значения функции для разных значений y:
F(x, 0) = x
F(x, 1) = 2 x
F(x, 2) = 3 x
F(x, 3) = 4 x

Зафиксируем взаимосвязь каждого значения функции со значением из предыдущей строчки:
F(x,0) = x = g(x)
F(x,1) = 2 x = x + x = F(x,0) + x = h (x, 0, F(x,0))
F(x,2) = 3 x = 2x + x = F(x,1) + x = h (x, 1, F(x,1))
F(x,3) = 4 x = 3x + x = F(x,2) + x = h (x, 2, F(x,2))

Помня, что h(x, y, z) = h(x, y, F(x,y)), получим,

g(x) = x, h(x, y, z) = z + x.

Задание 4. Применить оператор минимизации к функции

h(x,y,z) = x – y – z

Решение:

Приравняем функцию к 0 и выразим последнюю переменную z через предыдущие:

$x - y - z = 0$

$z = x - y$

$\mu(x,y) = x - y$

Задание 5. Получить из базовых функций с помощью основных операций функцию:

$$f(x,y) = (x + y) / (x - y)$$

Решение:

Так как в функции есть операция деления (или извлечения корня), то в начале обязательно используем обратную минимизацию (данная функция уже минимизирована, нужно найти исходную — до минимизации) :

$\mu(x,y) = (x + y) / (x - y)$

$z = (x + y) / (x - y)$

$z (x - y) = (x + y)$

$xz - yz = x + y$

$x (1 - z) + y (1 - z) = 0$

$F_1(x, y, z) = x (1 - z) + y (1 - z)$

При желании можно открыть скобки, на процесс решения это никак не повлияет.

Теперь нужно для данной функции F_1 найти $g(x, y)$ и $h(x, y, z, t)$ такие, что

$F_1(x, y, 0) = g(x, y)$

$F_1(x, y, z+1) = h (x, y, z, F_1(x, y, z))$, где t – значение функции F_1 от предыдущего значения переменной z.

Зададим z целые значения от 0.

$F_1(x, y, 0) = x (1 - 0) + y (1 - 0) = x + y$

$F_1(x, y, 1) = x (1 - 1) + y (1 - 1) = 0$

$F_1(x, y, 2) = x (1 - 2) + y (1 - 2) = - x - y$

$F_1(x, y, 3) = x (1 - 3) + y (1 - 3) = -2x - 2y$

Выразим текущее значение функции F_1 через предыдущее:

$F_1(x, y, 0) = x + y = g(x, y)$

$F_1(x, y, 1) = 0 = (x + y) - (x + y) = F_1(x, y, 0) - (x + y)$

$F_1(x, y, 2) = - x - y = 0 - (x + y) = F_1(x, y, 1) - (x + y)$

$F_1(x, y, 3) = -2x - 2y = (- x - y) - (x + y) = F_1(x, y, 2) - (x + y)$

Видим, что для получения текущего значения необходимо вычесть из предыдущего $(x + y)$. Зафиксируем $g(x, y)$ и $h(x, y, z, t)$:

$g(x, y) = x + y$

$h(x, y, z, t) = h (x, y, z, F_1(x, y, z)) = t - (x + y)$.

Примечание: в случае, если первоначальная функция f(x,y) содержит только сумму, произведение и целые степени, то можно сразу переходить к использованию обратной примитивной рекурсии для поиска g(x) и h(x, y, z).

В случае применения обратной минимизации число переменных в итоговых функциях (g и h) будет на одну больше, как указано в разобранном примере.

Машина Тьюринга

Задание 1. Определите третий шаг, указав текущее состояние машины Тьюринга по ленте:

	x	y	z	∧
q_0				∧ q_1R
q_1	∧ q_2R	∧ q_3R	∧ q_4R	∧ q_1R
q_2	x q_2R	y q_2R	z q_2R	x q_kS
q_3	x q_3R	y q_3R	z q_3R	y q_kS
q_4	x q_4R	y q_4R	z q_4R	z q_kS

∧	∧	z	z	x	y	∧
	▲					

Решение:

Лента изначально находиться в состоянии q_0:

∧	∧	z	z	x	y	∧	q_0
	▲						

Головка «видит» пустой символ. Исходя из таблицы, оставляем пустой символ, переходим в состояние q_1 и делаем шаг вправо.

∧	∧	z	z	x	y	∧	q_1
		▲					

Это был первый шаг. Находясь в состоянии q_1, видим символ «z». Согласно таблице заменяем «z» на «∧», переходим в состояние q_4 и шагаем вправо.

∧	∧	∧	z	x	y	∧	q_4
			▲				

Это был второй шаг. Теперь, находясь в состоянии q_4, видим символ «z». Согласно таблице оставляем «z» без изменений, остаемся в состоянии q_4 и шагаем вправо.

∧	∧	∧	z	x	y	∧	q_4
				▲			

Это был третий шаг.

Задание 2. Задано число в 8-чной системе счисления. Если оно чётное, то прибавить к нему 1, если нет – стереть всё слово. Построить машину Тьюринга и её граф состояний. Проверить работу машины на примере.

Решение:

Исходя из задачи, алфавит состоит из символов {0, 1, 2, 3, 4, 5, 6, 7, ∧ }.

Так как в тексте не указано, где находиться головка в начале работы программы, то самостоятельно определяем это. В данном случае удобней, чтобы головка находилась в конце слова. Чтобы совсем не упрощать решение, пусть головка расположена на пустой ячейке сразу после слова.

Например:

∧	7	4	2	6	∧	∧	q_0
					▲		

Вспомним, что для систем счисления с четным основанием (двоичная, четверичная, 6-ична, 8-ичная, 10-ичная и т.д.) число четное, если последняя цифра четная.

Распишем порядок работы алгоритмов:

1) 1 шаг влево / попадаем на последнюю цифру числа

2) Если видим нечетную цифру (1, 3, 5 или 7), стираем все цифры, двигаясь налево. Как только видим пустой символ (т.е. в числе больше нет цифр), останавливаемся.
3) Если видим четную цифру (0, 2, 4 или 6), заменяем её на большую (1, 3, 5 или 7, соответственно) и останавливаемся.

Реализуем этот алгоритм. Будем записывать программу Машины Тьюринга, указывая состояния по горизонтали, а символы – по вертикали. Первое действие алгоритма таково:

	q_0	q_1	q_2
∧	∧ q_1 L		
0			
1			
2			
3			
4			
5			
6			
7			

В состоянии q_1 «видим» последнюю цифру числа. Рассмотрим вначале случай нечетного числа:

	q_0	q_1	q_2
∧	∧ q_1 L		
0			
1		∧ q_2 L	
2			
3		∧ q_2 L	
4			
5		∧ q_2 L	
6			
7		∧ q_2 L	

Удаляем нечетный символ и переходим в состояние q_2 . В состоянии q_2 будем удалять все символы, идя налево:

	q_0	q_1	q_2
∧	∧ q_1 L		
0			∧ q_2 L
1		∧ q_2 L	∧ q_2 L
2			∧ q_2 L
3		∧ q_2 L	∧ q_2 L
4			∧ q_2 L
5		∧ q_2 L	∧ q_2 L
6			∧ q_2 L
7		∧ q_2 L	∧ q_2 L

В состоянии q_2 «видим» пустой символ, значит, в числе все цифры закончились, всё удалено. Останавливаемся:

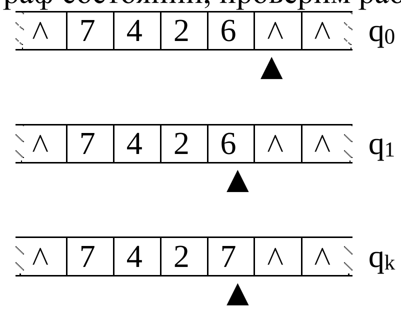
	q_0	q_1	q_2
∧	∧ q_1 L		∧ q_k S
0			∧ q_2 L
1		∧ q_2 L	∧ q_2 L

2			∧ q_2 L
3		∧ q_2 L	∧ q_2 L
4			∧ q_2 L
5		∧ q_2 L	∧ q_2 L
6			∧ q_2 L
7		∧ q_2 L	∧ q_2 L

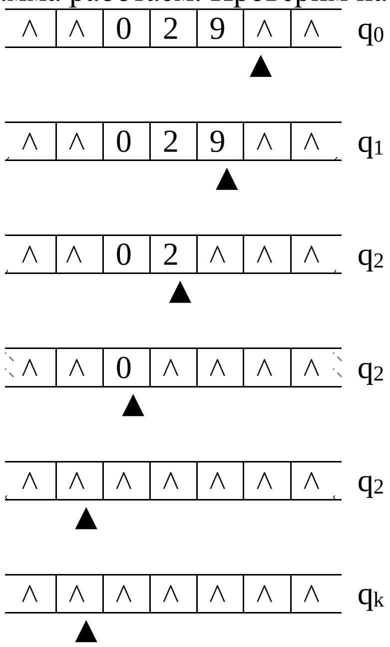
Рассмотрим случай, если последняя цифра четная. Если «видим» цифру «0», то должны её заменить на «1» и остановиться. Если «видим» «2» - на «3», и т.д.

	q_0	q_1	q_2
∧	∧ q_1 L		∧ q_k S
0		1 q_k S	∧ q_2 L
1		∧ q_2 L	∧ q_2 L
2		3 q_k S	∧ q_2 L
3		∧ q_2 L	∧ q_2 L
4		5 q_k S	∧ q_2 L
5		∧ q_2 L	∧ q_2 L
6		7 q_k S	∧ q_2 L
7		∧ q_2 L	∧ q_2 L

Прежде чем рисовать граф состояний, проверим работу на примерах.



На четном числе программа работаем. Проверим на нечетном:



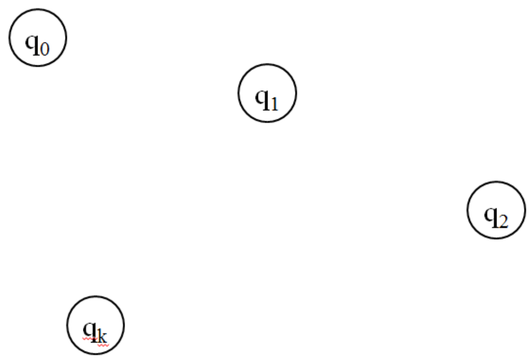
Единственный не рассмотренный случай – отсутствие числа на ленте. То есть в состоянии q_1 «видим» пустой символ, а не цифры. Дополним программу ещё одной остановкой:

	q_0	q_1	q_2
^	^ q_1 L	^ q_k S	^ q_k S
0		1 q_k S	^ q_2 L
1		^ q_2 L	^ q_2 L
2		3 q_k S	^ q_2 L
3		^ q_2 L	^ q_2 L
4		5 q_k S	^ q_2 L
5		^ q_2 L	^ q_2 L
6		7 q_k S	^ q_2 L
7		^ q_2 L	^ q_2 L

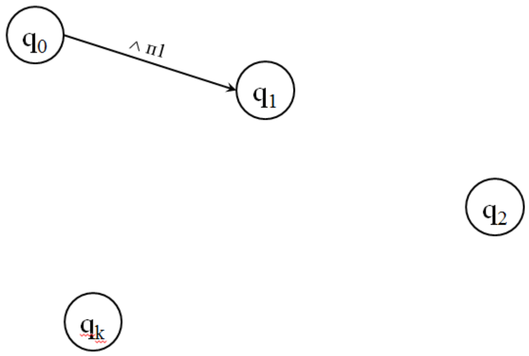
Теперь перейдем к построению графа состояний. Каждой заполненной ячейке присваиваем последовательные номера, начиная с 1. Это переходы.

	q_0	q_1	q_2
^	^ q_1 L 1	^ q_k S 2	^ q_k S 11
0		1 q_k S 3	^ q_2 L 12
1		^ q_2 L 4	^ q_2 L 13
2		3 q_k S 5	^ q_2 L 14
3		^ q_2 L 6	^ q_2 L 15
4		5 q_k S 7	^ q_2 L 16
5		^ q_2 L 8	^ q_2 L 17
6		7 q_k S 9	^ q_2 L 18
7		^ q_2 L 10	^ q_2 L 19

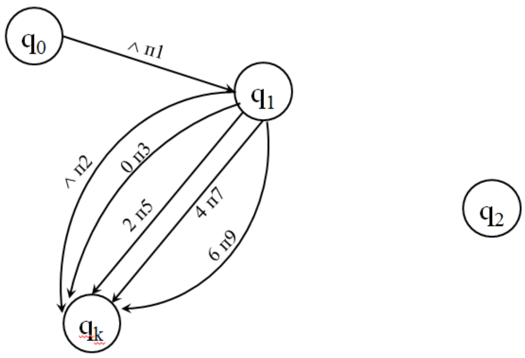
В программе используются 4 состояния: q_0, q_1, q_2, q_k. Исходя из логики программы, расставляем вершины-состояния:



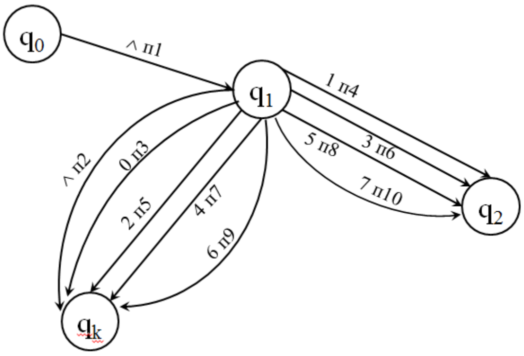
Проведем ориентированное ребро (стрелку) от q_0 . У каждого ребра должна быть подпись, состоящего из обозреваемого символа и номера перехода:



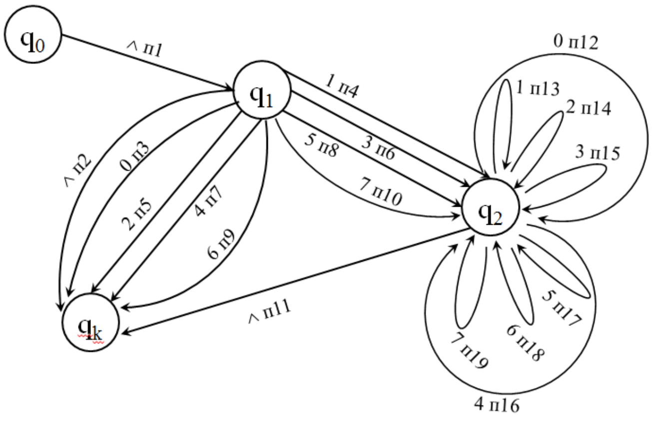
Проведем ребра от q_1 для случая четных цифр и пустого слова:



Теперь проведем ребра, соответствующие нечетным цифрами:



Из состояния q_2 выходят/входят восемь петлей и одно ребро к q_0.



Методическое издание

«Теория алгоритмов»:

методические указания по практической работе

для студентов Колледжа профессионального образования специальности

09.02.03 Программирование в компьютерных системах

Составитель:

Ежова Марина Алексеевна

Редактор ________________

Корректор _______________

Подписано в печать ________

Формат 60х84/14. Усл.печ.л. ______. Уч.-изд.л. ______.

Тираж 100 экз. Заказ

Редакционно-издательский отдел

Пермского государственного университета

614990. Пермь, ул. Букирева, 15

Типография Пермского государственного университета

614990. Пермь, ул. Букирева, 15