ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ
ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ
УНИВЕРСИТЕТ»

Т.К. Кацаран, Л.Н. Строева

МАШИНА ТЬЮРИНГА И РЕКУРСИВНЫЕ ФУНКЦИИ

Учебное пособие для вузов

Издательско-полиграфический центр
Воронежского государственного университета
2008

Утверждено научно-методическим советом факультета ПММ 25 мая 2008 г., протокол № 9

Рецензент д. т. н., проф. кафедры математических методов исследования операций Т.М. Леденева

Учебное пособие подготовлено на кафедре нелинейных колебаний факультета ПММ Воронежского государственного университета.

Рекомендуется для студентов 1 курса факультета ПММ ВГУ, Старооскольского и Лискинского филиалов ВГУ.

Для специальности 010500 – Прикладная математика и информатика

ВВЕДЕНИЕ

Слово «алгоритм» происходит от algorithmi – латинского написания имени узбекского математика и астронома, жившего в VIII–IX веках (783–850 гг.), Мухаммеда бен Мусы аль-Хорезми. Под этим именем в Средневековой Европе знали величайшего математика из Хорезма (город в современном Узбекистане). В своей книге «Об индийском счете» он сформулировал правила записи натуральных чисел с помощью арабских цифр и правила действий над ними. Затем понятие алгоритма стало использоваться в более широком смысле и не только в математике. Как для математиков, так и для практиков понятие алгоритма имеет важное значение.

Таким образом, можно сказать, что алгоритм – это точное предписание о выполнении в определенном порядке некоторой системы операций для решения всех задач одного и того же типа, определяющее последовательность действий, обеспечивающую получение требуемого результата из исходных данных. Заметим, что это не определение понятия «алгоритм», а только его описание, его интуитивный смысл.

Алгоритм может быть предназначен для выполнения его как человеком, так и автоматическим устройством.

Данное представление об алгоритме не является строгим с математической точки зрения, так как в нем используются такие понятия как «точное предписание» и «исходные данные», которые, вообще говоря, строго не определены.

Особенностью любого алгоритма является его способность решать некоторый класс задач. Например, это может быть алгоритм решения систем линейных уравнений, нахождение кратчайшего пути в графе и т. д.

Создание алгоритма, пусть даже самого простого, – процесс творческий. Он доступен исключительно живым существам, а долгое время считалось, что только человеку. Другое дело – реализация уже имеюще-

гося алгоритма. Ее можно поручить субъекту или объекту, который не обязан вникать в сущность дела, а возможно, и не способен его понять. Такой субъект или объект принято называть формальным исполнителем. Примером формального исполнителя может служить стиральная машина-автомат, которая неукоснительно исполняет предписанные ей действия, даже если вы забыли положить в нее порошок. Человек тоже может выступать в роли формального исполнителя, но в первую очередь формальными исполнителями являются различные автоматические устройства, и компьютер в том числе. Каждый алгоритм создается в расчете на вполне конкретного исполнителя. Те действия, которые может совершать исполнитель, называются его допустимыми действиями. Совокупность допустимых действий образует систему команд исполнителя. Алгоритм должен содержать только те действия, которые допустимы для данного исполнителя.

Поэтому обычно формулируют несколько общих свойств алгоритмов, позволяющих отличать алгоритмы от других инструкций.

Алгоритм должен обладать следующими свойствами.

Дискретность (прерывность, раздельность) – алгоритм должен представлять процесс решения задачи как последовательное выполнение простых (или ранее определенных) шагов. Каждое действие, предусмотренное алгоритмом, исполняется только после того, как закончилось исполнение предыдущего.

Определенность – каждое правило алгоритма должно быть четким, однозначным и не оставлять места для произвола. Благодаря этому свойству выполнение алгоритма носит механический характер и не требует никаких дополнительных указаний или сведений о решаемой задаче.

Результативность (конечность) – алгоритм должен приводить к решению задачи за конечное число шагов.

Массовость – алгоритм решения задачи разрабатывается в общем виде, то есть, он должен быть применим для некоторого класса задач, различающихся только исходными данными. При этом исходные данные могут выбираться из некоторой области, которая называется областью применимости алгоритма.

Теория алгоритмов – это раздел математики, который изучает общие свойства алгоритмов. Различают качественную и метрическую теорию алгоритмов.

Основной проблемой качественной теории алгоритмов является проблема построения алгоритма, обладающего заданными свойствами. Такую проблему называют алгоритмической.

Метрическая теория алгоритмов исследует алгоритм с точки зрения их сложности. Этот раздел теории алгоритмов известен также как алгоритмическая теория сложности.

При отыскании решений некоторых задач долго не удавалось найти соответствующий алгоритм. Примерами таких задач являются: а) указать способ, согласно которому для любой предикатной формулы за конечное число действий можно выяснить, является ли она тождественно-истинной или нет; б) разрешимо ли в целых числах диофантово уравнение (алгебраическое уравнение с целыми коэффициентами). Так как для решения этих задач найти алгоритмов не удалось, возникло предположение, что такие алгоритмы вообще не существуют, что и доказано: первая задача решена А. Черчем, а вторая – Ю.В. Матиясевичем и Г.В. Чудновским. Доказать это с помощью интуитивного понятия алгоритма в принципе невозможно. Поэтому были предприняты попытки дать точное математическое определение понятия алгоритма. В середине 30-х годов XX века С.К. Клини, А.А. Марков, Э. Пост, А. Тьюринг, А. Черч и другие предположили различные математические определения

понятия алгоритма. Впоследствии было доказано, что эти различные формальные математические определения в некотором смысле эквивалентны: вычисляют одно и то же множество функций. Это говорит о том, что, по-видимому, основные черты интуитивного понятия алгоритма правильно отражены в этих определениях.

Далее рассмотрим математическое уточнение алгоритма, предложенное А. Тьюрингом, которое называют машиной Тьюринга.

1. МАШИНА ТЬЮРИНГА

§ 1. Математическая модель машины Тьюринга

Идея создания машины Тьюринга, предложенная английским математиком А. Тьюрингом в тридцатых годах XX века, связана с его попыткой дать точное математическое определение понятия алгоритма.

Машина Тьюринга (МТ) – это математическая модель идеализированной цифровой вычислительной машины.

Машина Тьюринга является таким же математическим объектом, как функция, производная, интеграл, группа и т. д. Так же как и другие математические понятия, понятие машины Тьюринга отражает объективную реальность, моделирует некие реальные процессы.

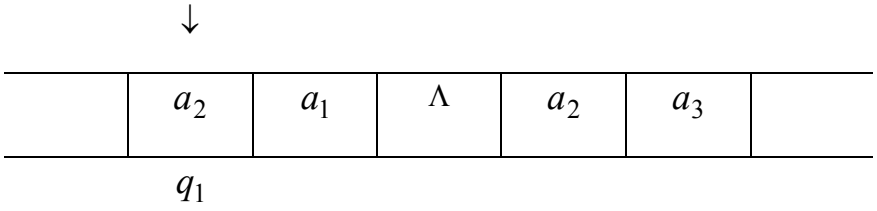
Для описания алгоритма МТ удобно представлять некоторое устройство, состоящее из четырех частей: ленты, считывающей головки, устройства управления и внутренней памяти.

1. Лента предполагается потенциально бесконечной, разбитой на ячейки (равные клетки). При необходимости к первой или последней клетке, в которой находятся символы пристраивается пустая клетка. Машина работает во времени, которое считается дискретным, и его моменты занумерованы 1, 2, 3, … . В каждый момент лента содержит конечное число клеток. В клетки в дискретный момент времени может быть записан только один символ (буква) из внешнего алфавита $A = \{\Lambda,\ a_1, a_2, ..., a_{n-1}\}$, $n \geq 2$. Пустая ячейка обозначается символом Λ, а сам символ Λ называется пустым, при этом остальные символы называются непустыми. В этом алфавите A в виде слова (конечного упорядоченного набора символов) кодируется та информация, которая подается в МТ. Машина «перерабатывает» информацию, поданную в виде слова, в новое слово.

2. Считывающая головка (некоторый считывающий элемент) перемещается вдоль ленты так, что в каждый момент времени она обозревает

ровно одну ячейку ленты. Головка может считывать содержимое ячейки и записывать в нее новый символ из алфавита А. В одном такте работы она может сдвигаться только на одну ячейку вправо (*П*), влево (*Л*) или оставаться на месте (*Н*). Обозначим множество перемещений (сдвига) головки $D = \{П, Л, Н\}$. Если в данный момент времени t головка находится в крайней клетке и сдвигается в отсутствующую клетку, то пристраивается новая пустая клетка, над которой окажется головка в момент $t+1$.

3. Внутренняя память машины представляет собой некоторое конечное множество внутренних состояний $Q = \{ q_0, q_1, q_2, ..., q_m \}$, $m \geq 1$. Будем считать, что мощность $|Q| \geq 2$. Два состояния машины имеют особое значение: q_1 – начальное внутреннее состояние (начальных внутренних состояний может быть несколько), q_0 – заключительное состояние или стоп-состояние (заключительное состояние всегда одно). В каждый момент времени МТ характеризуется положением головки и внутренним состоянием. Например, под ячейкой, над которой находится головка, указывается внутреннее состояние машины.



4. Устройство управления в каждый момент *t* в зависимости от считываемого в этот момент символа на ленте и внутреннего состояния машины выполняет следующие действия: 1) изменяет считываемый в момент *t* символ a_i на новый символ a_j (в частности оставляет его без изменений, т. е. $a_i = a_j$); 2) передвигает головку в одном из следующих направлений: *Н*, *Л*, *П*; 3) изменяет имеющееся в момент *t* внутреннее состояние машины

q_i на новое q_j, в котором будет машина в момент времени t +1 (может быть, что $q_i = q_j$).

Такие действия устройства управления называют командой, которую можно записать в виде:

$$q_i a_i \to a_j \; D \; q_j, \tag{1}$$

где q_i – внутреннее состояние машины в данный момент; a_i – считываемый в этот момент символ; a_j – символ, на который изменяется символ a_i (может быть $a_i = a_j$); символ D есть или *Н*, или *Л*, или *П* и указывает направление движения головки; q_j – внутреннее состояние машины в следующий момент (может быть $q_i = q_j$). Выражения $q_i a_i$ и $a_j \; D \; q_j$ называются левой и правой частями этой команды соответственно. Число команд, в которых левые части попарно различны, является конечным числом, так как множества $Q \setminus \{q_0\}$ и A конечны.

Не существует команд с одинаковыми левыми частями, т. е. если программа машины T содержит выражения $q_i a_i \to a_j \; D \; q_j$ и $q_t a_t \to a_k \; D \; q_k$, то $q_i \neq q_t$ или $a_i \neq a_t$ и $D \in \{П, Л, Н\}$.

Совокупность всех команд называется программой машины Тьюринга.

Максимальное число команд в программе равно $(n+1) \cdot m$, где $n+1 = |A|$ и $m+1 = |Q|$. Считается, что заключительное состояние команды q_0 может стоять только в правой части команды, начальное состояние q_1 может стоять как в левой так и в правой части команды.

Выполнение одной команды называется шагом. Вычисление (или работа) машины Тьюринга является последовательностью шагов одного за другим без пропусков, начиная с первого.

Итак, МТ задана, если известны четыре конечных множества: внешний алфавит A, внутренний алфавит Q, множество D перемещений головки и программа машины, представляющая собой конечное множество команд.

§ 2. Работа машины Тьюринга

Работа машины полностью определяется заданием в первый (начальный) момент: 1) слова на ленте, т. е. последовательности символов, записанных в клетках ленты (слово получается чтением этих символов по клеткам ленты слева направо); 2) положения головки; 3) внутреннего состояния машины. Совокупность этих трех условий (в данный момент) называется конфигурацией (в данный момент). Обычно в начальный момент внутренним состоянием машины является q_1, а головка находится либо над первой слева, либо над первой справа клеткой ленты.

Заданное слово на ленте с начальным состоянием q_1 и положение головки над первым словом называется начальной конфигурацией. В противном случае говорят, что машина Тьюринга не применима к слову начальной конфигурации.

Другими словами, в начальный момент конфигурация представима в следующем виде: на ленте, состоящей из некоторого числа клеток, в каждой клетке записан один из символов внешнего алфавита A, головка находится над первой слева или первой справа клеткой ленты и внутренним состоянием машины является q_1. Получившееся в результате реализации этой команды слово на ленте и положение головки называется заключительной конфигурацией.

Например, если в начальный момент на ленте записано слово $a_1 \Lambda a_2 a_1 a_1$, то начальная конфигурация будет иметь вид:

	a_1	a_2	Λ	a_1	a_1	
	q_1					

(под клеткой, над которой находится головка, указывается внутреннее состояние машины).

Работа машины Тьюринга состоит в последовательном применении команд, причем, применение той или команды определяется текущей конфигурацией. Так в приведенном выше примере должна применится команда с левой частью $q_1 a_1$.

Итак, зная программу и задав начальную конфигурацию, полностью определяем работу машины над словом в начальной конфигурации.

Если в работе машины Тьюринга в некоторый момент t выполняется команда, правая часть которой содержит q_0, то в такой момент работа машины считается законченной, и говорят, что машина применима к слову на ленте в начальной конфигурации. В самом деле, q_0 не встречается в левой части ни одной команды – этим и объясняется название q_0 «заключительное состояние». Результатом работы машины в таком случае считается слово, которое будет записано на ленте в заключительной конфигурации, т. е. в конфигурации, в которой внутреннее состояние машины есть q_0. Если же в работе машины ни в один из моментов не реализуется команда с заключительным состоянием, то процесс вычисления будет бесконечным. В этом случае говорят, что машина не применима к слову на ленте в начальной конфигурации.

§ 3. Примеры машин Тьюринга, работающих в алфавите {a, b}

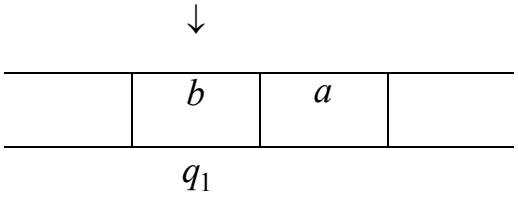
Проиллюстрируем работу машину Тьюринга на следующем примере.

Пример 1. Построить машину Тьюринга T_1, которая применима ко всем словам с внешним алфавитом $\{a,b\}$ и делает следующее: любое слово $x_1 x_2 ... x_n$, где $x_i = a$ или $x_i = b$ $(i = 1,2,...n)$ преобразует в слово $x_2 ... x_n x_1$, т. е., начиная работать при слове $x_1 x_2 ... x_n$ на ленте в начальной конфигурации, машина остановится, и в заключительной конфигурации на некотором участке ленты будет записано слово $x_2 ... x_n x_1$, а все остальные клетки ленты (если такие будут) окажутся пустыми.

Решение. За внешний алфавит машины T_1 возьмем множество $A=\{\Lambda,a,b\}$, а за внутренний $-\ Q=\{q_0,q_1,q_2,q_3\}$. Команды определим следующим образом: $q_1a\to\Lambda\Pi q_2$, $q_1b\to\Lambda\Pi q_3$, $q_iy\to y\Pi\Pi_i$, где $y\in\{a,b\}$, $i=2,3$; $q_2\Lambda\to aHq_0$, $q_3\Lambda\to bHq_0$.

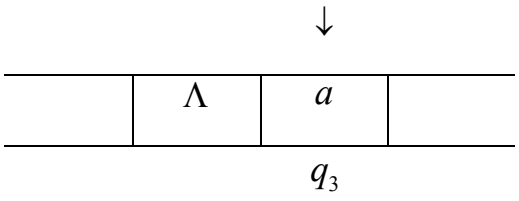
Рассмотрим работу машины T_1 над словом ba. В работе машины над словом ba начальная конфигурация имеет следующий вид:

(1)



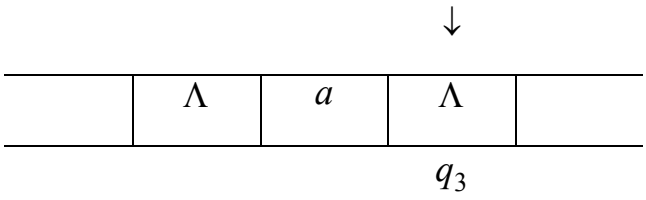
На первом шаге действует команда: $q_1b\to\Lambda\Pi q_3$. В результате создается следующая конфигурация:

(2)



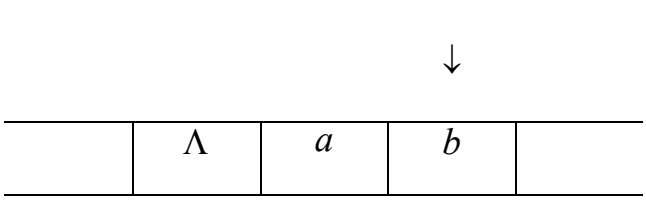
На втором шаге действует команда $q_3a\to\ a\Pi\Pi_3$, и на машине создается конфигурация

(3)



Наконец, третий шаг обусловлен командой $q_3\Lambda\to bHq_0$. В результате чего создается конфигурация:

(4)



Эта конфигурация является заключительной, так как машина оказалась в состоянии остановки q_0.

Таким образом, слово ba переработано в слово ab.

Полученную последовательность конфигураций можно записать более коротким способом. Конфигурация (1) записывается в виде следующего слова в алфавите $A \cup Q$: q_1ba (содержимое обозреваемой ячейки записано справа от состояния, в котором находится данный момент машина). Далее, конфигурация (2) записывается так: q_3a, конфигурация (3) – $aq_3\Lambda$, и наконец, (4) – abq_0. Вся последовательность записывается так: $q_1ba \Rightarrow \Lambda q_3a \Rightarrow aq_3\Lambda \Rightarrow abq_0$.

Пример 2. Применить машину Тьюринга T_1 из примера 1 к слову *bbabb*, исходя из начального положения, при котором в состоянии q_1 обозревается крайняя левая ячейка, в котором содержится символ этого слова.

Решение

(1)		b	b	a	b	b	
		q_1					
(2)		Λ	b	a	b	b	
			q_3				
(3)		Λ	b	a	b	b	
				q_3			
(4)		Λ	b	a	b	b	
					q_3		
(5)		Λ	b	a	b	b	
						q_3	

(6)

	Λ	b	a	b	b	Λ	
						q_3	

(7)

	Λ	b	a	b	b	b	
						q_0	

Более короткая запись этой последовательности конфигураций, т. е. процесса работы машины будет

$$q_1bbabb \Rightarrow \Lambda q_3babb \Rightarrow \Lambda bq_3abb \Rightarrow \Lambda baq_3bb \Rightarrow$$

$$\Rightarrow \Lambda babq_3b \Rightarrow \Lambda babbq_3\Lambda \Rightarrow babbb.$$

Таким образом, слово $bbabb$ переработано машиной в слово $babbb$.

Пример 3. Задается машина Тьюринга внешним алфавитом $A = \{a, b, c\}$, алфавитом внутренних состояний $Q = \{q_0, q_1, q_2, q_3\}$ и программой:

$$q_1a \to q_1Лa,\ \ q_2a \to q_3Пb,\ \ q_3a \to q_1Лa,\ \ q_1b \to q_2Лa,\ \ q_2b \to q_2Лb,$$
$$q_3b \to q_3Пb,\ \ q_1c \to q_0a,\ \ q_2c \to q_2Лc,\ \ q_2\Lambda \to q_2Нa, q_3c \to q_3Пc.$$

Заметим, что программа этой машины может быть записана в виде следующей таблицы:

	q_1	q_2	q_3
a	$q_1Лa$	$q_3Пb$	$q_1Лa$
b	$q_2Лa$	$q_2Лb$	$q_3Пb$
c	q_0a	$q_2Лc$	$q_3Пc$

Для того чтобы определить по таблице, что будет делать машина, находясь, например, в состоянии q_2 и наблюдая в обозреваемой ячейке символ b, нужно найти в таблице клетку, находящуюся на пересечении столбца q_2 и строки b. В этой клетке записано $q_2bЛ$. Это означает, что на следующем

шаге машина останется в прежнем состоянии q_2, сохранит содержимое обозреваемой ячейки b и перейдет к обозрению следующей левой ячейки на ленте.

Предположим, что в начальной конфигурации головка находится над крайней правой клеткой.

Применим эту машину к слову *bbcbb*. Последовательность конфигураций, возникающих в процессе работы машины (исходная конфигурация – стандартная начальная):

$$bbcbq_1b \Rightarrow bbcq_2ba \Rightarrow bbq_2cba \Rightarrow bq_2bcba \Rightarrow q_2bbcba \Rightarrow q_2abbcba \Rightarrow$$

$$\Rightarrow bq_3bbcba \Rightarrow bbq_3bcba \Rightarrow bbbq_3cba \Rightarrow bbbcq_3ba \Rightarrow bbbcbq_3a \Rightarrow bbbcq_1ba \Rightarrow$$

$$\Rightarrow bbbq_2caa \Rightarrow bbq_2bca \Rightarrow bq_2bbca \Rightarrow q_2bbbca \Rightarrow q_2abbbca \Rightarrow bq_3bbbca \Rightarrow$$

$$\Rightarrow bbq_3bbca \Rightarrow bbbq_3bca \Rightarrow bbbbq_3ca \Rightarrow bbbbcq_3 \Rightarrow bbbbq_1ca \Rightarrow bbbbq_0aa.$$

Нетрудно заметить, что данная МТ реализует операцию сложения: в результате ее работы на ленте записано подряд столько букв b, сколько их было всего записано по обе стороны от буквы c перед началом работы машины.

Из приведенных примеров следует, что МТ – это некоторое правило (алгоритм) для преобразования слов алфавита $A \cup Q$, т. е. конфигураций. Таким образом, для определения (построения) машины Тьюринга нужно задать ее внешний и внутренний алфавиты, программу и указать, какие из символов обозначают пустую ячейку и заключительное состояние.

§ 4. Способы задания машин Тьюринга, операции над ними

Рассмотрим три основные операции, применяемые над машинами Тьюринга.

1. Пусть машины Тьюринга T_1 и T_2 имеют соответственно программы $П_1$ и $П_2$. q_0^1 – заключительное состояние T_1, q_1^2 – начальное со-

стояние T_2. Предположим, что внутренние алфавиты этих машин не пересекаются. Заменим везде в программе $П_1$ состояние q_0^1 на состояние q_1^2 и полученную программу объединим с программой $П_2$. Новая программа П определяет машину Т, которая называется *композицией машин* T_1 и T_2 (по паре состояний (q_0^1, q_1^2)) и *обозначается* $T_1 \circ T_2$ или T_1T_2. Более подробная запись полученной машины будет выглядеть – Т = = $T(T_1, T_2, (q_0^1, q_1^2))$. Внешний алфавит композиции $T_1 \circ T_2$ является объединением алфавитов машин T_1 и T_2.

2. Пусть q_0 – некоторое заключительное состояние машины Т, а q_k – какое-либо состояние этой же машины Т, не являющееся заключительным. Заменим всюду в программе П машины Т символ q_0 на q_k. В результате получим программу П', которая определяет машину Т' (q_0, q_k). Машина Т' называется *итерацией* машины Т по паре состояний (q_0, q_k).

3. Пусть машины Тьюринга T_1, T_2 и T_3 задаются программами $П_1$, $П_2$ и $П_3$ соответственно. Внутренние алфавиты этих машин не пересекаются. Пусть $q_0^{'}$ и $q_0^{''}$ – какие-либо заключительные состояния машины T_1. Заменим в программе $П_1$ состояние $q_0^{'}$ некоторым начальным состоянием $q_1^{'}$ машины T_2, а состояние $q_0^{''}$ некоторым начальным состоянием $q_1^{''}$ машины T_3. Затем новую программу объединим с программами $П_2$ и $П_3$. Получим программу П, задающую машину Тьюринга $T = T(T_1(q_0^{'}, q_1^{'}), T_2(q_0^{''}, q_1^{''}), T_3)$, которая называется *разветвлением машин* T_2 и T_3, *управляемым* машиной T_1.

Отметим, что при построении сложных машин Тьюринга применяют так называемую *операторную запись алгоритма*. Этот способ построения впервые был предложен А.А. Ляпуновым в 1953 году. Так как специальный операторный язык для записи алгоритмов носит вспомогательный характер, то не имеет смысла давать его строгое формально-логическое определение. Остановимся на кратком описании операторного языка и рассмотрим пример.

Операторную запись алгоритма представляет собой строку, состоящую из символов, обозначающих машины, символов перехода (вида $\underline{|q'}_{k}$ и $\underline{q''|}_{k}$), а также символов α и ω, служащих для обозначения соответственно начала и окончания работы алгоритма. В операторной записи (некоторого алгоритма) выражение $T_i \underline{|q_{i0}}_{k} T_j \ldots T_m \underline{q_{n1}|}_{k} T_n$ обозначает разветвление машин T_j и T_n, управляемое машиной T_i, причем заключительное состояние q_{i0} машины T_i заменяется начальным состоянием q_{n1} машины T_n, а всякое другое заключительное состояние машины T_i заменяется начальным состоянием машины T_j (одним и тем же). Если машина T_i имеет одно заключительное состояние, то символы $\underline{|q_{i0}}$ и $\underline{q_{n1}|}$ служат для обозначения безусловного перехода. Там, где могут возникнуть недоразумения, символы q_{i0} и q_{n1} опускаются.

Пример 4. Операторная схема

$$\underline{|}_{2}\ \underline{|}_{1} T_1\ \alpha\ T_2\ \underline{|q_{20}}_{1}\ T_3\ \underline{|q_{30}}_{2}\ T_4\ \omega$$

описывает следующий «процесс вычисления». Начинает работу машина T_2. Если она заканчивает работу в состоянии q_{20}, то начинает работать машина T_1, а по окончании работы T_1 вновь «выполняет работу» машина T_2. Если же машина T_2 останавливается в некотором заключительном состоянии, отличном от q_{20}, то работу продолжает машина T_3. Если T_3 приходит в заключительное состояние q_{30}, то начинает работу машина T_1; если же T_3 заканчивает работу в некотором заключительном состоянии, отличном от q_{30}, то работу продолжает машина T_4. Если машина T_4 когда-либо остановится, то процесс вычисления на этом заканчивается.

§ 5. Задачи и упражнения для самостоятельного решения

1. Выяснить, применима ли машина Тьюринга, задаваемая программой в алфавите $\{\Lambda, 1\}$

	Λ	1
q_1	$\Lambda\ П\ q_2$	$1\ П\ q_1$
q_2	$\Lambda\ П\ q_3$	$1\ Л\ q_1$
q_3	$\Lambda\ Н\ q_0$	$1\ Л\ q_2$

к слову Р: 1) $P = 1^3\ 0^2\ 1^2$; 2) $P = 1^3\ 0\ 1^3$; 3) $P = 1[01]^2\ 1$. Если применима, то найти результат применения МТ к слову Р.

Ответ: в случаях 1), 3) – машина Тьюринга применима, в 2) – машина Тьюринга не применима.

2. По заданной машине Тьюринга и начальной конфигурации $К_1$ найти заключительную конфигурацию.

T_1

	Λ	1
q_1	$\Lambda\ Н\ q_0$	$1\ П\ q_2$
q_2	$1\ Л\ q_0$	$\Lambda\ П\ q_3$
q_3	$1\ Л\ q_1$	$\Lambda\ П\ q_1$

1) $К_1 = 1\ q_1\ 1^5$;
2) $К_1 = q_1\ 1^3\ 01$;
3) $К_1 = 10\ q_1\ 1^4$;

T_2

	Λ	1
q_1	$\Lambda\ Н\ q_0$	$\Lambda\ П\ q_2$
q_2	$\Lambda\ П\ q_1$	$1\ Л\ q_2$

1) $К_1 = 1^2\ q_1\ 1^3\ 01$;
2) $К_1 = 1\ q_1\ 1^4$.

Ответ: 1) $1^2\ 0^2\ 1\ q_0\ 01$; 2) $[10]^2\ 0\ q_0\ 1^2$.

3. Построить машину Тьюринга, которая применима ко всем словам в алфавите {Λ, a, b} и делает следующее: любое слово $x_1x_2...x_n$, где $x_i = a$ или $x_i = b$, $i = \overline{1,n}$, преобразует в слово $x_3...x_nx_1x_2$.

2. ФУНКЦИИ, ВЫЧИСЛИМЫЕ ПО ТЬЮРИНГУ

§ 1. Описание класса функций

С развитием науки и практики появились задачи, для которых не были найдены методы их решения. Возник вопрос: отсутствие алгоритма решения для данного класса задач есть результат недостаточного знания о задачах этого класса или решающего алгоритма для данного класса не существует?

Для решения этой проблемы было введено понятие вычислимой функции.

Будем рассматривать функции f от одного или нескольких аргументов, заданные на множестве $N_0 = \{0,1,2,3,...,n,...\}$ всех неотрицательных целых чисел или на некоторых его подмножествах (частичные функции) и принимающие значения в множестве N. Область определения Df функции f – это подмножество множества $N_0^n = N_0 \times N_0 \times ... \times N_0$:

$$Df = \{(x_1,...,x_n) \in N_0^n : f(x_1,...,x_n) \in N \text{ – определено}\}.$$

Значение функции $f(x_1,...,x_n)$ на наборе $(m_1,...,m_n) \in N_0^n$ определено, если $f(m_1,...,m_n) = m$, где $m \in N_0$, в противном случае функция f считается неопределенной на заданном наборе.

Под *числовыми функциями* будем понимать функции, значениями которых и значениями их аргументов являются неотрицательные целые числа.

Опишем класс числовых функций, совпадающий с множеством всех вычислимых функций.

Назовем исходными следующие числовые функции:

1) нуль-функция: $o(x) = x$ при каждом $x \in N_0$;

2) функция следования: $s(x) = x + 1$ при каждом $x \in N_0$;

3) функция выбора аргумента: $I_m^{(n)}(x_1, x_2,...,x_n) = x_m$ при всех $(x_1,...,x_n) \in N_0^n$, $m = \overline{1,n}$, $n = 1,2,3...$

Например, функциями выбора аргументов являются:

$$I_2^{(2)}(x_1,x_2)=x_2,\ \ I_1^{(1)}(x_1)=x_1,\ I_1^{(3)}(x_1,x_2,x_3)=x_3.$$

Для построения более сложных функций используют различные операции. Важнейшие из них – это операции суперпозиции и примитивной рекурсии. Эти операции рассмотрим в разделе 3.

При вычислении числовых функций на машинах Тьюринга часто пользуются специальным кодированием чисел. Например, натуральное число m будем задавать набором из $m+1$ единиц, который будем обозначать через 1^{m+1}. Итак, 0 будем задавать 1, 1 – 11, 2 – 111 и т. д.

Числовая функция $f(x_1,...,x_n)$ *называется вычислимой по Тьюрингу*, если существует машина Тьюринга T_f, удовлетворяющая следующим двум условиям:

1) для любого набора $(m_1,\ ...,m_n)\in Df\subset N^n$ и такого, что $f(m_1,...,m_n)=m$, машина Тьюринга применима к слову

$$1^{m_1+1}\Lambda 1^{m_2+1}\Lambda...\Lambda 1^{m_n+1}, \qquad (2)$$

и в заключительной конфигурации на некотором участке ленты будет записано слово 1^{m+1}, а остальные участки ленты, если такие будут, окажутся пустыми.

2) если $(m_1,...,m_n)\notin Df$ машина Тьюринга T_f не применима к слову (2).

Если функция f вычислима по Тьюрингу с помощью машины T_f, то будем говорить, что машина T_f вычисляет функцию f.

§ 2. Примеры функций, вычислимых по Тьюрингу

Пример 5. Построить машину Тьюринга T_3 с внешним алфавитом $\{\Lambda,1\}$, которая вычисляет функцию $f(x)=x+1$.

Решение. Будем предполагать, что перед началом работы на ленте машины записаны исходные значения аргумента и считывающая головка обозревает первый слева значащий символ (см. рис. 1). Кроме того, после выполнения вычислений считывающая головка останавливается в заключительной конфигурации.

Прежде чем приступить к написанию программы работы машины Тьюринга, следует определить порядок ее работы для получения результата. В нашем случае после окончания работы машины на ленте должно быть занято на одну ячейку больше, чем на ней занято ячеек перед началом работы.

Команды этой машины могут быть определены следующим образом:

$$q_1 1 \to 1 П q_1, \; q_1 \Lambda \to 1 H q_0 .$$

Работа машины T_3 при вычислении $f(1)$ состоит из конфигураций:

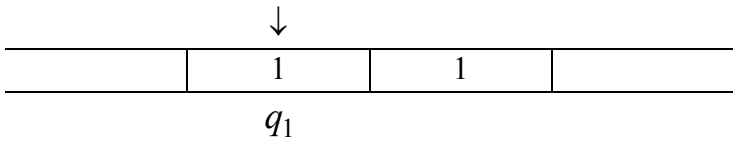


Рис. 1

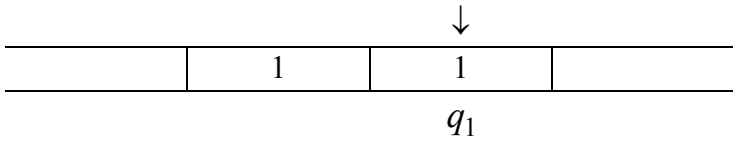


Рис. 2

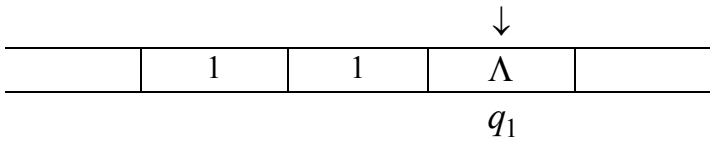


Рис. 3

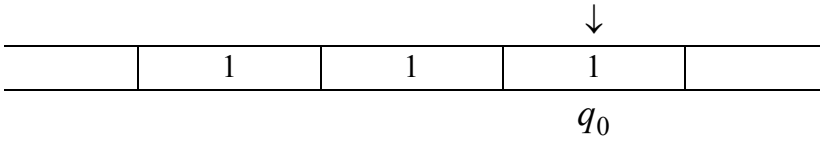


Рис. 4

Очевидно, что наилучший способ выполнить это требование состоит в следующем. После начала работы считывающая головка машины должна переместиться на одну ячейку влево, записать в пустую ячейку 1 и остановиться.

Первая команда имеет вид: $q_1 1 \to 1 П q_1$ (см. рис. 2).

Согласно этой команде, машина, находясь в состоянии q_1, читает символ 1, записанный в обозреваемой ячейке, оставляет этот символ в ячейке, остается в прежнем состоянии q_1 и сдвигается на одну ячейку ленты вправо.

Вторая команда имеет вид: $q_1 \Lambda \to 1 H q_0$ (см. рис. 3).

Таким образом, согласно этой команде, машина, находясь в состоянии q_1, читает пустой символ, записанный в обозреваемой ячейке, записывает на его место символ 1, переходит в состояние q_0 и останавливается.

Программа работы машины Тьюринга также может быть записана в виде таблицы. Столбцам таблицы соответствуют символы внешнего алфавита, а строкам – состояние машины.

В рассматриваемом примере таблицы работы машины Тьюринга имеет вид:

	Λ	1
q_1	1 Н q_0	1П q_1

Пример 6. Построить машину T_4, вычисляющую числовую функцию $f(x, y) = x + y$.

Решение. Пусть внешним алфавитом данной машины является алфавит $\{\Lambda,1\}$. Работа машины состоит из конфигураций:

$$q_1 1 \to 1 П q_1, \quad q_1 \Lambda \to 1 П q_2, \quad q_2 1 \to 1 П q_2, \quad q_2 \Lambda \to \Lambda Л q_3,$$
$$q_3 1 \to \Lambda Л q_4, \quad q_4 1 \to \Lambda Л q_0.$$

Следует отметить, что для данной машины T_4 выписаны все команды, осуществляющие вычисление функции $f(x, y) = x + y$.

Замечание. Все арифметические операции представляют собой вычислимые по Тьюрингу функции.

§ 3. Задачи и упражнения для самостоятельного решения

1. Построить машину Тьюринга, вычисляющую нуль-функцию **0** (x) = 0 в алфавите {Λ, 1}.

Указание: Взять множество $Q = \{q_0, q_1\}$, подставить вместо всех единиц Λ, а когда встретиться символ Λ, то подставить символ 1.

2. Вычисляет ли МТ в алфавите {Λ,1 }

1) с программой

	q_1	q_2	q_3
Λ	1 **Л** q_2	Λ **П** q_0	Λ **Н** q_0
1		1 **Н** q_3	Λ **Л** q_3

функцию $\overline{sign\ x} = \begin{cases} 1, & \text{если } x = 0, \\ 0, & \text{если } x \neq 0. \end{cases}$

2) с программой

	q_1	q_2	q_3	q_4
Λ	Λ **Л** q_2	Λ **П** q_0	Λ **П** q_4	Λ **П** q_4
1		1 **Л** q_3	Λ **Л** q_3	1 **Н** q_0

функцию $sign\ x = \begin{cases} 0, & \text{если } x = 0, \\ 1, & \text{если } x \neq 0. \end{cases}$

3. Построить машину Тьюринга, которая вычисляет функцию:

1) $f(x, y) = x \cdot y$; 2) $f(x) = x^2$;

3) функцию выбора аргумента $J_2^{(3)} = (x_1, x_2, x_3) = x_2$.

4*. Реализовать на МТ алгоритм вычисления функции $f(n) = n + 2$, где $n \in N$.

Указание: Взять множество состояний $Q = \{q_0, q_1, q_2\}$. Число n на ленте МТ записывается в десятичной системе счисления. Состояние q_1 заменяет последнюю цифру числа n, если эта цифра меньше 8, цифрой, на две единицы большей, и переходит в стоп-состояние. Если последняя цифра числа n равна 8, то ее заменить на 0 и перейти влево в состояние q_2. Состояние q_2 добавляет к следующему разряду 1. Если же последняя цифра числа n равна 9, то ее заменить на 1 и перейти влево в состояние q_2.

3. РЕКУРСИВНЫЕ ФУНКЦИИ

§ 1. Элементарные функции. Правила подстановки и примитивная рекурсия

Всякий алгоритм однозначно ставит в соответствие исходным данным (в случае если, он определен на них) результат. Поэтому с каждым алгоритмом однозначно связана функция, которую он вычисляет. Исследование проблемы остановки машины Тьюринга показывает, что не для всякой функции существует вычисляющий ее алгоритм. Поэтому в 30-х годах XX века была создана теория рекурсивных функций. В этой теории, как и вообще в теории алгоритмов, принят конструктивный, финитный подход, основной чертой которого является то, что все множество исследуемых объектов (в нашем случае функции) строится из конечного числа исходных объектов – базиса – с помощью простых операций, эффективная выполнимость которых достаточно очевидна. Операции над функциями в дальнейшем будем называть операторами.

Пусть имеется некоторый алгоритм α. Областью применимости алгоритма α называют совокупность тех объектов, к которым он применим.

Говорят, что алгоритм α вычисляет функцию f, если его область применимости совпадает с областью определения функции f, и алгоритм α перерабатывает всякий элемент x из своей области применимости в $f(x)$.

Американскими математиками Клини, а впоследствии Черчем были строго определены математические функции, называемые примитивно-рекурсивными. Черч высказал гипотезу о том, что множество всех рекурсивных функций совпадает с множеством всех вычислимых функций. Это предположение получило название тезиса Черча. Гипотеза Черча не может быть доказана, поскольку использует нестрогое понятие вычислимой функции.

Позже американские математики Пост и впоследствии Тьюринг ввели понятие математической машины, которую называют машиной Поста или машиной Тьюринга (см. главу 1 «Машина Тьюринга»). Тьюринг высказал гипотезу (известную также как тезис Тьюринга) о том, что для всякой вычислимой функции может быть построена машина Тьюринга. Этот тезис также не может быть доказан, так как включает нестрогое понятие вычислимой функции.

Доказано, что для всякой рекурсивной функции может быть построена машина Тьюринга и, обратно, всякая машина Тьюринга вычисляет рекурсивную функцию.

Известны также другие способы уточнения понятия алгоритма, например, нормальный алгоритм Маркова.

Практический опыт показывает, что тезисы Черча и Тьюринга являются верными, не имеется ни одного опровержения этих утверждений.

Дадим элементарное определение рекурсивных функций.

Рекурсивные функции – это функции, определенные некоторым специальным образом. Из названия следует, что их вычисление содержит обращение к самим себе (при меньших значениях аргументов). Подразумевается, что рекурсивные функции являются арифметическими функциями, т. е. область их определения и область значений является подмножеством множества N_0 или совпадает с ним.

Введем следующие правила для получения новых функций из уже имеющихся, которые упоминались в разделе 2:

1) нуль функция: $o(x) = x$ при каждом $x \in N_0$;

2) функция следования: $s(x) = x + 1$ при каждом $x \in N_0$; (3)

3) функция выбора аргумента: $I_m^{(n)}(x_1, x_2, ..., x_n) = x_m$

при всех $(x_1, ..., x_n) \in N_0^n$, $m = \overline{1,n}$, $n = 1,2,3...$

Операция суперпозиции *(подстановка)* заключается в подстановке одних рекурсивных функций вместо аргументов в другие рекурсивные функции.

Пусть даны числовые функции $f(x_1,...,x_m)$, $g_1(x_1,...,x_n)$, $g_2(x_1,...,x_n)$, …, $g_m(x_1,...,x_n)$ и пусть

$$h(x_1,...,x_n) = f(g_1(x_1,...,x_n),...,g_m(x_1,...,x_n)).$$

Тогда будем говорить, что функция h получена с помощью подстановки из функций $f(x_1,...,x_m)$, $g_1(x_1,...,x_n)$, $g_2(x_1,...,x_n)$, …, $g_m(x_1,...,x_n)$.

Например, функция $o(x_1,...,x_n) = 0$ получается с помощью подстановки из функций $o(x)$ и $I_m^{(n)}(x_1,x_2,...,x_n) = x_m$: $o(x_1,...,x_n) = o(I_m^{(n)}(x_1,x_2,...,x_n))$. А функция $s_m^{(n)}(x_1,x_2,...,x_n) = x_m + 1$ – из функций $s(x)$ и $I_m^{(n)}(x_1,x_2,...,x_n) = x_m$: $s_m^{(n)}(x_1,x_2,...,x_n) = s(I_m^{(n)}(x_1,x_2,...,x_n))$.

Рассмотрим операцию примитивной рекурсии. Эта операция строит функцию от n+1 аргументов, если имеются две числовые функции $g(x_1,...,x_n)$ и функция $h(x_1,...,x_n,x_{n+1},x_{n+2})$ (функция от n+2 аргументов, $n \geq 1$). Таким образом, если требуется построить функцию от некоторого числа аргументов, необходимо иметь две функции: одна из них g зависит от числа аргументов, которое на единицу меньше, чем число аргументов в строящейся функции f, а вторая функция h зависит от числа аргументов на единицу большего числа аргументов функции f.

Операция примитивной рекурсии определяется следующим образом:

$$\begin{cases} f(x_1,...,x_n,0) = g(x_1,...,x_n) \\ f(x_1,...,x_n,y+1) = h(x_1,...,x_n,y,f(x_1,...,x_n,y)) \end{cases}.$$

В развернутом виде имеем, когда $y = 0$:

$f(x_1,...,x_n,0) = g(x_1,...,x_n)$.

Если $y = 1$, то $f(x_1,...,x_n,1) = h(x_1,...,x_n,0,f(x_1,...,x_n,0))$.

Если $y = 2$, то $f(x_1,...,x_n,2) = h(x_1,...,x_n,1,f(x_1,...,x_n,1))$ и т. д.

Иногда операцию примитивной рекурсии обозначают $f = R(g,h)$. Заметим, что операция примитивной рекурсии фактически строит таблицу значений новой функции f.

Определение. Функция называется *примитивно-рекурсивной*, если она может быть получена из исходных функций (3) с помощью конечного числа подстановок и примитивных рекурсий.

Определение. *Функция называется примитивно-рекурсивной, если она может быть записана с помощью элементарных рекурсивных функций с использованием конечного числа операций суперпозиции и примитивной рекурсии.*

Определение. Функция называется *частичной*, если она определена не для всех значений аргументов.

Пример 7. Доказать, что функция $f(x,y) = x + y$ является примитивно-рекурсивной.

Решение. Так как заданная функция является функцией двух аргументов, то для использования операции примитивной рекурсии мы должны иметь функцию g, зависящую от одного аргумента, и функцию h, зависящую от трех аргументов. Определим эти функции.

В функции $f(x,y) = x + y$ положим $y = 0$. Тогда имеем $f(x,0) = x$ – это тоже тождественная функция. Полагая $y = 1$, получим $f(x,1) = x + 1$ – это функция следования.

Таким образом, выбираем следующие элементарные функции: тождественную $g(x) = x$ и функцию следования $h(x,y,z) = z + 1$. Заметим, что здесь $h(x,y,z) = I_3^3(x,y,s(x))$.

Используя схему примитивной рекурсии:

$f(x,0) = g(x) = x,$

$f(x,1) = h(x,0,f(x,0)) = f(x,0) + 1 = x + 1,$

$$f(x,2) = h(x,1,f(x,1)) = f(x,1) + 1 = x + 2,$$

$$f(x,y) = h(x,y-1,f(x,y-1)) = x + y - 1 + 1 = x + y.$$

Таким образом, построена функция (таблица ее значений), которая равна сумме двух слагаемых.

В дальнейшем, если нужно доказать примитивную рекурсивность некоторой функции, можно использовать не только элементарные рекурсивные функции, но и те функции, примитивная рекурсивность которых уже доказана. Например, можно использовать функцию из выше рассмотренного примера.

Так как исходные функции являются всюду определенными и операции подстановки и примитивной рекурсии сохраняют всюду определенность, то из определения примитивно-рекурсивной функции следует, что каждая примитивно рекурсивная функция является всюду определенной.

§ 2. Правило взятия μ-оператора. Классы функций

Рассмотрим еще одну операцию, называемую ***μ*-оператором.** Пусть $g(x_1,...,x_n,y)$ – произвольная числовая функция. Зафиксируем значения $x_1, x_2,..., x_n$ и через $\mu y[g(x_1,...,x_n,y) = 0]$ обозначим наименьшее число y такое, что:

1) для всех t, $0 \le t < y$, $g(x_1,...,x_n,t)$ определено и больше нуля;

2) $g(x_1,...,x_n,y)$ определено и равно нулю.

Если же одно из этих условий не выполнено, т. е. для некоторого t $g(x_1,...,x_n,t)$ не определено или же не для всех z $g(x_1,...,x_n,z)$ определено и больше нуля, будем считать, что выражение $\mu y[g(x_1,...,x_n,y) = 0]$ не определено.

Пример 8. Пусть дана функция $g(x,y)=x-y-3$. Тогда $\mu y[4-y-3=0]=1$, так как $4-0-3=1\neq 0$, $4-1-3=0$. $\mu y[3-y-3=0]=0$, а $\mu y[g(0,y)=0]$, $\mu y[g(1,y)=0]$ и $\mu y[g(2,y)=0]$ не определены, потому что $k-0-3$, где $k=0,1,2$ в области натуральных чисел не определено.

Пусть $f(x_1,...,x_n)=\mu y[g(x_1,...,x_n,y)=0]$. В этом случае говорят, что функция f получена из функции g с помощью μ-оператора.

Пример 9. Пусть $f_1(x)=\mu y[4-y-3=0]$, тогда $f_1(x)$ получается из $g(x,y)=x-y-3$ с помощью μ-оператора. Так функция $g(x,y)=x-y-3$ не является всюду определенной, то и значения $f_1(0)$, $f_1(1)$ и $f_1(2)$ не определены.

Пример 10. Функция $f_2(y)=\mu y[4-y-3=0]$ является нигде не определенной, так как для любого натурального числа m в области натуральных чисел $0-m-3$ не определено. Значит с помощью μ-оператора получена нигде не определенная функция.

Пример 11. Функция $x-y$ не является всюду определенной (например, $0-m$ не определено), но функция $f_3(x)=\mu y[x-y=0]$, полученная из нее с помощью μ-оператора, всюду определенная, причем для любого m: $f_3(m)=m$ (все разности $m-0$, $m-1$, …, $m-(m-1)$ определены и отличны от нуля, а $m-m=0$). Таким образом, с помощью μ-оператора получена всюду определенная функция.

Пример 12. Константная функция $g_1(x,y)=5$ всюду определена, а функция $f_4(x)=\mu y[g_1(x,y)=0]$ нигде не определена.

Определение. Функция называется частично рекурсивной, если она может быть получена из исходных с помощью применения конечного числа раз подстановок, примитивных рекурсий и μ-оператора. Всюду определенная частично рекурсивная функция называется общерекурсивной.

§ 3. Теорема о совпадении классов частично-рекурсивных и вычислимых по Тьюрингу функций

Частично-рекурсивная функция может быть не всюду определенной, а каждая примитивно-рекурсивная функция всюду определена. Поэтому класс примитивно-рекурсивных функций строго содержится в классе частично-рекурсивных и даже общерекурсивных.

Теорема. Функция вычислима по Тьюрингу тогда и только тогда, когда она частично рекурсивна.

Определение. Множество называется рекурсивно перечислимым, если оно совпадает с множеством значений некоторой общерекурсивной функции.

Примером могут служить множества четных и нечетных чисел.

Замечание. Каждое рекурсивное перечислимое множество порождается некоторой машиной Тьюринга, которая вычисляет соответствующую общерекурсивную функцию.

Рассмотрим характеристическую функцию множества M:

$$\chi_M(x) = \begin{cases} 0, & \text{если } x \notin M, \\ 1, & \text{если } x \in M. \end{cases}$$

Замечание. Числовое множество M называется рекурсивным, если общерекурсивна характеристическая функция этого множества.

Замечание. Каждое рекурсивное множество является рекурсивно-перечислимым.

Доказательство. Очевидно, что если множество M рекурсивно, то существует машина Тьюринга T_1, которая вычисляет его характеристическую функцию. Построим новую машину T_2, которая сначала изготавливает два экземпляра исходного слова и помнит фиксированный элемент из M. Затем она работает как машина T_1, но сохраняет один экземпляр исходного слова, т. е. слова на ленте в начальной конфигурации. Если в результате работы T_1 получается единица, машина T_2 стирает на ленте все кроме со-

храняемого экземпляра исходного слова и останавливается. Если же в результате работы машины T_1 получается нуль, T_2 стирает все буквы на ленте, пишет фиксированный элемент из M и останавливается.

Обратное включение неверно, т. е. существует рекурсивно перечислимое, но не рекурсивное множество.

Следовательно, для рекурсивного перечислимого, но не рекурсивного множества нельзя решить, принадлежит ли ему данный элемент или нет.

Вышесказанное может быть использовано при доказательстве несуществования алгоритма для решения всех задач данного класса.

Замечание. Не всякое числовое множество является рекурсивно-перечислимым.

§ 4. Задачи и упражнения для самостоятельного решения

1. Доказать, что данные функции примитивно-рекурсивные:

$f(x, y) = x(y+1)$; $f(x) = 2^x$.

2. Построить машину Тьюринга, вычисляющую функции:

$f(x, y) = x+y$; $f(x) = 2x$.

Список литературы

1. Плотников А.Д. Дискретная математика : учеб. пособие / А.Д. Плотников. – М. : Новое издание, 2005. – 288 с.

2. Мощенский В.А. Лекции по математической логике : учебное пособие для студ. ун-тов по спец. «Прикладная математика» / В.А. Мощенский. – Минск : Изд-во БГУ, 1973. – 132 с. ; ил.

3. Игошин В.И. Математическая логика и теория алгоритмов / В.И. Игошин. – Саратов : Изд-во Сарат. ун-та, 1991. – 256 с.

4. Гаврилов Г.П. Задачи и упражнения по дискретной математике : учебное пособие / Г.П. Гаврилов, А.А. Сапоженко. – 3-е изд., перераб. – М. : Физматлит, 2006. – 416 с.

5. Кузнецов О.П. Дискретная математика для инженера / О.П. Кузнецов, Г.М. Адельсон-Вельский. – 2-е изд., перераб. и доп. – М. : Энергоатомиздат, 1988. – 480 с.

СОДЕРЖАНИЕ

Учебное издание

Кацаран Татьяна Константиновна,
Строева Любовь Николаевна

МАШИНА ТЬЮРИНГА И РЕКУРСИВНЫЕ ФУНКЦИИ

Учебное пособие для вузов

Редактор И.Г. Валынкина

Подписано в печать 12.12.2008. Формат 60×84/16. Усл. печ. л. 2,03.
Тираж 50 экз. Заказ 2168.

Издательско-полиграфический центр
Воронежского государственного университета.
394000, г. Воронеж, пл. им. Ленина, 10. Тел. 208-298, 598-026 (факс)
http://www.ppc.vsu.ru; e-mail: pp_center@ppc.vsu.ru

Отпечатано в типографии Издательско-полиграфического центра
Воронежского государственного университета.
394000, г. Воронеж, ул. Пушкинская, 3. Тел. 204-133