МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
БОРИСОГЛЕБСКИЙ ФИЛИАЛ
(БФ ФГБОУ ВО «ВГУ»)

МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ ПО УЧЕБНОЙ ДИСЦИПЛИНЕ
Теория алгоритмов

1. Код и наименование направления подготовки:

44.03.01 Педагогическое образование

2. Профиль подготовки:

Информатика и информационные технологии в образовании

3. Квалификация (степень) выпускника:

Бакалавр

4. Форма обучения:

Заочная

5. Кафедра, отвечающая за реализацию дисциплины:

Кафедра прикладной математики, информатики, физики и методики их преподавания

6. Составители:

Л. В. Лободина, кандидат педагогических наук, доцент

О.Г. Ромадина, кандидат педагогических наук

7. Методические указания для обучающихся по освоению дисциплины

Приступая к изучению учебной дисциплины, целесообразно ознакомиться с учебной программой дисциплины, электронный вариант которой размещён на сайте БФ ВГУ.

Это позволит обучающимся получить четкое представление о:

– перечне и содержании компетенций, на формирование которых направлена дисциплина;

– основных целях и задачах дисциплины;

– планируемых результатах, представленных в виде знаний, умений и навыков, которые должны быть сформированы в процессе изучения дисциплины;

– количестве часов, предусмотренных учебным планом на изучение дисциплины, форму промежуточной аттестации;

– количестве часов, отведенных на контактную и самостоятельную работу;

– структуре дисциплины, основных разделах и темах;

– системе оценивания учебных достижений;

– учебно-методическом и информационном обеспечении дисциплины.

Знание основных положений, отраженных в рабочей программе дисциплины, поможет обучающимся ориентироваться в изучаемом курсе, осознавать место и роль изучаемой дисциплины в подготовке выпускника, строить свою работу в соответствии с требованиями, заложенными в программе.

Основными формами контактной работы по дисциплине являются лекции и практические занятия, посещение которых обязательно.

Подготовка к практическим занятиям ведется на основе планов практических занятий, которые размещены на сайте филиала. В ходе подготовки к практическим занятиям необходимо изучить в соответствии с вопросами для повторения конспекты лекций, основную литературу, ознакомиться с дополнительной литературой. Кроме того, следует повторить материал лекций, ответить на контрольные вопросы, изучить образцы решения задач, выполнить упражнения (если такие предусмотрены).

При подготовке к промежуточной аттестации необходимо повторить пройденный материал в соответствии с учебной программой, примерным перечнем вопросов, выносящихся на зачет. Рекомендуется использовать конспекты лекций и источники, перечисленные в списке литературы в рабочей программе дисциплины, а также ресурсы электронно-библиотечных систем.

8. Методические материалы для обучающихся по освоению теоретических вопросов дисциплины

№	Наименование раздела дисциплины	Содержание раздела дисциплины
1	Алгоритмы в математике.	Интуитивное понятие алгоритма. Необходимость уточнения понятия алгоритма. Различные подходы к определению алгоритма.
2	Машины Тьюринга и поста как модели алгоритма	Устройство и принципы работы машины Тьюринга (МТ). Решение задач синтеза и анализа на МТ. Операции с МТ. Функции, вычислимые по Тьюрингу. Тезис Черча-Тьюринга. Устройство и принципы работы машины Поста.
3	Нормальные алгоритмы Маркова (НАМ) и частично рекурсивные функции (ЧРФ) как модели алгоритма	Марковские подстановки. НАМы и их применение к словам. Нормально вычислимые функции и принцип нормализации Маркова. Операторы подстановки, примитивной рекурсии, минимизации. Классы ЧРФ, общерекурсивных и примитивно рекурсивных функций и их взаимосвязь. Тезис Черча для ЧРФ.
4	Равносильность различных определений алгоритма. Нумерация алгоритмов	Понятие вычислимой функции. Разрешимые и перечислимые множества. Характеристические функции. Понятие об эффективной нумерации. Нумерация машин Тьюринга.
5	Алгоритмически неразрешимые проблемы	Теорема Клини. Недетерминированная машина Тьюринга (НДТ). Основные алгоритмически неразрешимые проблемы. Алгоритмическая сводимость.

	Классы сложности P и NP.	Характеристики сложности вычислений. Полиномиально и экспоненциально решаемые задачи. Полиномиальная сводимость. Теорема Кука. Основные NP-полные задачи
6	Оптимизационные задачи теории алгоритмов.	Общая характеристика «жадных» алгоритмов. Условия оптимальности «жадных» алгоритмов.

9. Методические материалы для обучающихся по подготовке к практическим занятиям

Машина Тьюринга

В 1937 году английский инженер и математик А. Тьюринг сформулировал понятие алгоритма в виде абстрактной математической машины, которая по имени своего создателя получила название *машины Тьюринга (МТ).*

Машину Тьюринга можно представить себе в виде некоторого автоматического устройства, через считывающую головку которого проходит бесконечная в обе стороны лента, разбитая на ячейки. В каждый момент времени устройство, находясь в определенном состоянии, обозревает содержимое только одной ячейки, в которой записано не больше одного символа некоторого заданного алфавита. Рабочий шаг машины заключается в том, что устройство стирает символ, записанный в обозреваемой ячейке, заменяет его на новый символ и перемещается на соседнюю слева или справа ячейку в новом состоянии. Шаг осуществляется в соответствии с *командой.* Набор команд задает *программу* данной МТ.

Чтобы задать программу МТ, необходимо задать:

1. Внешний алфавит $A = \{a_0, a_1, ..., a_n\}$, символы которого записываются в ячейки ленты. Символ a_0 называют «пустым»; он означает, что в ячейке ничего не записано и вводится для удобства рассуждений.
2. Алфавит внутренних состояний $Q = \{q_0, q_1, ..., q_m\}$, в каждом из которых может находиться данная МТ. Среди всех внутренних состояний машины выделяют два: q_0 - состояние остановки, попав в которое, машина прекращает работу, и q_1 - начальное состояние, находясь в котором, машина начинает работу.
3. Набор команд, каждая из которых имеет вид:

$$q_i a_j \to q_k a_l K, \ \text{где} \begin{cases} 1 \le i \le m, \quad 0 \le j \le n, \\ 0 \le k \le m, \quad 0 \le l \le n, \\ K \in \{Л, П, С\}. \end{cases}$$

Буквы Л, П, С, записанные в правой части команды означают, что после замены символа a_j на a_l, машина передвигается на одну ячейку влево (Л), вправо (П) или остается на месте (С), переходя из состояния q_i в состояние q_k.

Так как работа МТ полностью определяется ее состоянием в данный момент - q_i и содержимым обозреваемой в этот момент ячейки - a_j, то для каждой пары символов (q_i, a_j), программа МТ должна содержать одну и только одну команду, начинающуюся этими символами. Очевидно, ни одна команда не может начинаться с символа q_0, поскольку он означает состояние остановки. Таким образом, программа МТ с внешним алфавитом $A = \{a_0, a_1, ..., a_n\}$ и алфавитом внутренних состояний $Q = \{q_0, q_1, ..., q_m\}$ должна содержать $m \cdot (n + 1)$ команд. Программа МТ чаще всего записывается в виде таблицы, в строках которой записаны символы внешнего алфавита, в столбцах – символы алфавита внутренних состояний, а на пересечении строки с символом q_i и столбца с символом a_j записана правая часть команды $q_k a_l$ К:

A / Q	$\mathbf{a}_0$	$\mathbf{a}_1$	…	$\mathbf{a}_j$	…	a_n
$\mathbf{q}_1$						
⋮						
$\mathbf{q}_i$				$\mathbf{q}_k \mathbf{a}_l \mathbf{K}$		
⋮						
q_m						

Словом в алфавите A или Q или $A \cup Q$ называется любая конечная последовательность букв соответствующего алфавита.
k-той конфигурацией называется слово в алфавите A, записанное на ленте к началу k-го шага работы МТ, с указанием того, какая ячейка обозревается на этом шаге и в каком состоянии находится машина.
Конфигурация называется *заключительной,* если состояние, в котором находится МТ – заключительное, т.е. состояние остановки q_0.

Непустое слово α в алфавите $A \setminus \{a_0\}$ *воспринимается МТ в стандартном положении,* если оно записано в последовательных ячейках ленты, все другие ячейки пусты, и МТ обозревает крайнюю справа ячейку из тех, в которых записано слово α. Если при этом МТ находится в состоянии q_1, то стандартное положение называется *начальным,* если в состоянии q_0 - то *заключительным.*

Слово α перерабатывается МТ в слово β, если от слова α, воспринимаемого машиной в начальном состоянии, после выполнения конечного числа команд МТ приходит к слову β, воспринимаемому в положении остановки.
Задача переработки слов МТ с заданной программой называется *задачей применения МТ к словам.*

ПРИМЕР. Применить машину Тьюринга с данной программой:

A / Q	$\mathbf{a}_0$	**1**	*
$\mathbf{q}_1$	$\mathbf{q}_1 \mathbf{a}_0 \mathbf{П}$	$\mathbf{q}_3 \mathbf{a}_0 \mathbf{Л}$	$q_0 a_0$
$\mathbf{q}_2$	$\mathbf{q}_2 \mathbf{a}_0 \mathbf{Л}$	$\mathbf{q}_4 \mathbf{a}_0 \mathbf{П}$	$q_4 a_0 П$
$\mathbf{q}_3$	$\mathbf{q}_2 \mathbf{a}_0 \mathbf{П}$	$\mathbf{q}_3 \mathbf{1Л}$	q_3*Л
q_4	$q_1 a_0 Л$	$q_4 1П$	q_4*П

к слову α = 111*1, воспринимаемому в начальном стандартном положении.

РЕШЕНИЕ:

Так как слово α = 111*1 воспринимается МТ в начальном стандартном положении, то в начальный момент времени машина обозревает крайнюю правую ячейку, видит в ней символ 1, находясь в начальном состоянии q_1:

$$\underbrace{111*1}_{q_1} \quad .$$

Находим в программе МТ команду, которая начинается сочетанием символов $q_1 1$. Для этого выбираем в таблице строку с символом q_1 и столбец с символом 1. На пересечении этих строки и столбца записана вторая часть команды: $q_3 a_0 Л$, выполняя которую машина стирает 1, записывает вместо нее пустой символ a_0 и переходит в соседнюю слева ячейку в состоянии q_3:

$$1\,1\,\underbrace{1}_{q_3}\,*\,a_0\,.$$

В ячейке слева машина обозревает символ * в состоянии q_3. На пересечении строки q_3 и столбца * записана команда q_3*Л, по которой машина оставляет * без изменения и в том же состоянии q_3 передвигается еще на одну ячейку влево:

$$1\,\underbrace{1}_{q_3}\,1\,*\,a_0\,.$$

Теперь машина обозревает 1 в состоянии q_3. Следующий шаг она делает согласно команде: $q_3 1 \to q_3 1Л$:

$$\underbrace{1}_{q_3}\,1\,1\,*\,a_0\,.$$

Очевидно, что эта команда будет повторяться до тех пор, пока машина будет «видеть» 1, находясь в состоянии q_3:

$$\underbrace{1}_{q_3}\,1\,1\,*\,a_0 \quad \text{и} \quad \underbrace{a_0}_{q_3}\,1\,1\,1\,*\,a_0\,.$$

«Пересчитав» все единицы, считывающее устройство дойдет до пустой ячейки, в которой по договоренности записан пустой символ a_0. Обозревая a_0 в состоянии q_3, машина должна выполнить следующую команду: $q_3 a_0 \to q_2 a_0 П$,

согласно которой машина перейдет к ячейке с 1 в новом состоянии q_2:

$$a_0\,\underbrace{1}_{q_2}\,1\,1\,*\,a_0\,.$$

Далее по команде $q_2 1 \to q_4 a_0 П$ машина стирает 1 и переходит уже на соседнюю справа ячейку в новом состоянии q_4:

$$a_0\,a_0\,\underbrace{1}_{q_4}\,1\,*\,a_0\,.$$

По команде $q_4 1 \to q_4 1П$ машина, оставив без изменений следующую справа 1 и оставаясь в состоянии q_4, дойдет в этом состоянии до ячейки с символом *:

$$a_0\,1\,\underbrace{1}_{q_4}\,*\,a_0 \quad \text{и} \quad a_0\,1\,1\,\underbrace{*}_{q_4}\,a_0\,.$$

Далее по команде q_4*$\to q_4$*П, машина, оставив * без изменения, передвинется еще на одну ячейку вправо, где нет никакого символа, то есть, по договоренности, мы считаем, что там записан пустой символ a_0:

$$a_0\,1\,1\,*\,\underbrace{a_0}_{q_4}\,a_0\,.$$

По команде $q_4 a_0 \to q_1 a_0 Л$, «проверив», что единиц справа на ленте больше нет, машина, ничего не вписав в пустую ячейку, возвращается назад, влево, перейдя в состояние q_1:

$$a_0\ 11\ \underbrace{*}_{q_1}\ a_0\ .$$

Наконец, по команде $q_1 * \to q_0 a_0$ машина «стирает» символ * и останавливается.

Таким образом, слово α = 111*1 перерабатывается данной МТ в слово β = 11.
Можно сказать, что данная МТ вычитает из числа единиц, записанных слева от *, число единиц, записанных справа от нее.

Конструирование машин Тьюринга

Задача написания программы машины Тьюринга, которая решала бы некоторую проблему, получила название *задачи конструирования машины Тьюринга.* Для конструирования машины Тьюринга необходимо задать внешний алфавит из n символов, алфавит внутренних состояний из m символов и программу, содержащую m(n + 1) команду. Формат допустимых команд описан в задании 1. Задача конструирования является более сложной, чем задача применения готовой машины Тьюринга к словам. Здесь не может существовать единого алгоритма, описывающего процесс конструирования программы. Выбор символов внешнего алфавита и состояний внутреннего алфавита обуславливается данными конкретной задачи, для решения которой строится программа.

ПРИМЕР. Сконструировать машину Тьюринга, которая вычисляла бы функцию f(x) = 2x для чисел в десятичной системе счисления.

РЕШЕНИЕ:

В условии сказано, что числа представлены в десятичной системе счисления, поэтому, кроме обязательного пустого символа a_0 во внешний алфавит необходимо включить цифры от 0 до 9:

$$A = \{a_0, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9\} \quad .$$

В алфавит внутренних состояний обязательно входят начальное состояние q_1 и состояние остановки q_0. Иногда ими можно и ограничиться, а иногда необходимы новые состояния. В нашем случае, чтобы определить это, проведем ряд рассуждений.
При удвоении натурального числа, записанного в десятичной системе счисления, реализуется один из двух случаев.

1. Если все цифры исходного числа не превосходят 4, то каждая из цифр заменяется цифрой, в два раза большей – получаем удвоенное число. Для осуществления этой операции достаточно состояния q_1 и передвижения считывающего устройства влево на соседнюю ячейку.
2. Если же одна или несколько цифр превышают 4, то потребуется новое состояние q_2. Оно должно обеспечить не только замену последней цифры, но и изменение предыдущих цифр числа.

Получаем следующий набор команд:

1. $q_1 0 \to q_1 0Л$;
2. $q_1 1 \to q_1 2Л$;
3. $q_1 2 \to q_1 4Л$;
4. $q_1 3 \to q_1 6Л$;
5. $q_1 4 \to q_1 8Л$.

В первом случае, после удвоения каждой цифры числа, считывающее устройство доходит до пустой ячейки в состоянии q_1. Так как число удвоено, то машина должна остановиться. Получаем пятую команду:

$$6.\ q_1 a_0 \to q_0 a_0.$$

Во втором случае, когда последняя (или вообще, какая-либо из цифр) числа есть одна из цифр от 5 до 9, получаем следующие команды:

$$7.\ q_1 5 \to q_2 0Л;$$
$$8.\ q_1 6 \to q_2 2Л;$$
$$9.\ q_1 7 \to q_2 4Л;$$
$$10.\ q_1 8 \to q_2 6Л;$$
$$11.\ q_1 9 \to q_2 8Л.$$

Состояние q_1 в правой части этих команд использовать нельзя, так как в этом случае машина, перейдя влево, удвоит предыдущую цифру и мы получим неверный результат. Например, число 15 при использовании только состояния q_1 перешло бы в число 20, а не в 30, как должно быть.

Теперь осталось задать команды для случая, когда машина в состоянии q_2 обозревает ячейку с одной из цифр или с пустым символом. Получаем последние 11 команд, которые завершают программу.

$$12.\ q_2 0 \to q_1 1Л; \qquad 17.\ q_2 5 \to q_2 1Л;$$
$$13.\ q_2 1 \to q_1 3Л; \qquad 18.\ q_2 6 \to q_2 3Л;$$
$$14.\ q_2 2 \to q_1 5Л; \qquad 19.\ q_2 7 \to q_2 5Л;$$
$$15.\ q_2 3 \to q_1 7Л; \qquad 20.\ q_2 8 \to q_2 7Л;$$
$$16.\ q_2 4 \to q_1 9Л; \qquad 21.\ q_2 9 \to q_2 9Л;$$
$$22.\ q_2 a_0 \to q_0 1.$$

Команды с 12 по 16 обеспечивают правильное изменение цифр числа от 0 до 4, если эти цифры не являются последними в десятичной записи числа; для цифр от 5 до 9 ту же задачу решают команды с 17 по 21. Последняя команда программы срабатывает в том случае, когда машина «добирается» до пустой ячейки в состоянии q_2.

Таким образом, алфавит внутренних состояний сконструированной машины содержит три состояния:

$$Q = \{q_0, q_1, q_2\}.$$

Проверим работу программы на числах **99108** и **1043**.

$$9910\underbrace{8}_{q_1} \xrightarrow{10} 991\underbrace{0}_{q_2}6 \xrightarrow{12} 99\underbrace{1}_{q_1}16 \xrightarrow{2} 9\underbrace{9}_{q_1}216 \xrightarrow{11}$$

$$\xrightarrow{11} a_0\underbrace{9}_{q_2}8216 \xrightarrow{21} \underbrace{a_0}_{q_2}98216 \xrightarrow{22} \underbrace{1}_{q_0}98216.$$

Действительно, **99108•2 = 198216.**

$$104\underbrace{3}_{q_1} \xrightarrow{4} 10\underbrace{4}_{q_1}6 \xrightarrow{5} 1\underbrace{0}_{q_1}86 \xrightarrow{1} \underbrace{1}_{q_1}086 \xrightarrow{2}$$

$$\xrightarrow{2} \underbrace{a_0}_{q_1}2086 \xrightarrow{21} \underbrace{a_0}_{q_0}2086.$$

Действительно, **1043•2 = 2086.**

Машина произвольного доступа

Очередная попытка сформулировать понятие алгоритма в виде абстрактной математической машины была предпринята в 70-е годы 20 века. Эта машина получила название *машины произвольного доступа (МПД).*

МПД состоит из бесконечного числа регистров $R_1, R_2, \ldots, R_n, \ldots$, в каждом из которых может быть записано натуральное число или 0. Будем обозначать число, записанное в регистре R_n через r_n.

Состоянием машины или *конфигурацией* будем называть последовательность чисел $r_1, r_2, \ldots, r_n, \ldots$. Работа МПД заключается в изменении ее конфигураций путем выполнения команд в порядке их написания. Машина имеет следующие типы команд.

1. *Команды обнуления.* Для всякого натурального числа n имеется команда Z(n), действие которой заключается в замене содержимого регистра R_n на число 0. Содержимое других регистров при этом не меняется.

Обозначение: Z(n) - $r_n := 0$.

2. *Команды прибавления единицы.* Для всякого натурального числа n имеется команда S(n), действие которой заключается в увеличении содержимого регистра R_n на 1. Содержимое других регистров при этом не меняется.

Обозначение: S(n) - $r_n := r_n + 1$.

3. *Команды переадресации.* Для всяких натуральных чисел n и m имеется команда T(m, n), действие которой заключается в замене содержимого регистра R_n числом r_m, хранящимся в регистре R_m. Содержимое других регистров при этом не меняется.

Обозначение: T(m, n) - $r_n := r_m$ или $r_m \to R_n$.

4. *Команды условного перехода.* Для всяких натуральных n, m и q имеется команда J(m, n, q), действие которой заключается в следующем:
   - сравнивается содержимое регистров R_n и R_m, затем:
   - если $r_n = r_m$, то МПД переходит к выполнению команды с номером q;
   - если $r_n \neq r_m$, то МПД переходит к выполнению команды, следующей по списку.

Пусть P – программа МПД, K_0 – начальная конфигурация. Применение программы P к начальной конфигурации $P(K_0)$ называется *вычислением.* Будем обозначать команды программы P через $I_1, I_2, \ldots, I_S$. В качестве начальных конфигураций будем рассматривать только такие, в которых имеется лишь *конечное число* ненулевых элементов и вместо $(a_1, a_2, \ldots, a_n, 0, \ldots)$ будем писать $(a_1, a_2, \ldots, a_n)$.

Опишем класс функций, вычислимых на МПД. Множество натуральных чисел с нулем обозначим N_0. Будем рассматривать частичные функции

$$f : N_0 \times N_0 \times \ldots \times N_0 \to N_0$$

Будем говорить, что программа P вычисляет функцию f с результатом b ($b \in N$), если она применима к начальной конфигурации $(a_1, a_2, \ldots, a_n)$, f определена на этой конфигурации и $f(a_1, a_2, \ldots, a_n) = b$.

ПРИМЕР.

Записать программу и построить блок-схему МПД, которая вычисляла бы функцию f(x, y) = x + y.

РЕШЕНИЕ:

В качестве начальной конфигурации возьмем (x, y, 0). Тогда содержимое первого регистра до начала работы программы будет равно x, второго - y, а третьего – 0:

$$R_1 : r_1 : = x,\ R_2 : r_2 : = y,\ R_3 : r_3 : = 0.$$

Функция f(x, y) = x + y может быть вычислена следующей программой, состоящей из пяти команд:

I_1 = J(3, 2, 5) – команда условного перехода, которая сравнивает содержимое третьего и второго регистров и, если числа, записанные в них равны, то осуществляется переход к пятой команде I_5, а если не равны, то к следующей по списку команде I_2.

I_2 = S(1) - r_1: = r_1 + 1 – команда прибавления единицы к содержимому первого регистра.

I_3 = S(3) - r_3: = r_3 + 1 - прибавления единицы к содержимому третьего регистра.

I_4 = J(1, 1, 1) – команда, обеспечивающая цикл. Так как содержимое первого регистра всегда равно самому себе, то эта команда возвращает машину к выполнению первой команды до тех пор, пока число в третьем регистре не станет равно числу во втором регистре. Так прибавление по единице идет последовательно к содержимому третьего, а затем первого регистров, то когда в третьем регистре получим число y, то в первом, соответственно, получим x + y = f(x, y).

I_5 - останов. - **r_1: = x + y = f(x, y).**

Нумерация машин Тьюринга

Под нумерацией алгоритмов понимают эффективные кодирования натуральными числами множества всех алгоритмов для каждой из рассматриваемых моделей алгоритмов. Данные результаты относятся к числу фундаментальных, так как они используются, в частности, для установления невычислимости ряда конкретных функций.

Будем считать, что символы внешних алфавитов и алфавитов внутренних состояний машин Тьюринга берутся из двух множеств символов:

$$\{a_0, a_1, \ldots, a_i, \ldots\} \text{ и } \{q_0, q_1, \ldots, q_j, \ldots\}.$$

При этом будем считать, что a_0 принадлежит всем внешним алфавитам машин Тьюринга и означает пустой символ, а символы q_0 и q_1 принадлежат всем алфавитам внутренних состояний и означают начальное состояние и состояние остановки соответственно. Назовем следующий набор символов *стандартным алфавитом*:

$$\mathbf{A} = \{Л, П, С, a_0, a_1, \ldots, a_i, \ldots, q_0, q_1, \ldots, q_j, \ldots\}.$$

Каждому символу стандартного алфавита поставим в соответствие двоичный набор – *код* – согласно таблице 1.

Таблица 1

	Символ	**Код**	**Число нулей в коде**
Символы сдвига	**П**	**10**	**1**
	Л	**100**	**2**
	С	**1000**	**3**
Символы внешнего алфавита	**a_0**	**10000**	**4**
	a_1	**1000000**	**6**
	…	**…**	**…**
	a_i	**100…00**	**2i + 4**

	…	…	…
Символы алфавита состояний	$\mathbf{q}_0$	**100000**	**5**
	$\mathbf{q}_1$	**10000000**	**7**
	…	…	…
	$\mathbf{q}_j$	**100…000**	**2i + 5**
	…	…	…

Если команда машины Тьюринга I имеет формат:

$$I =: qa \to q'a'X, \quad X \in \{Л, П, С\},$$

то ей ставится двоичный набор по правилу:

$$Код(I) = Код(q)Код(a)Код(q')Код(a')Код(X) \quad (1)$$

Пусть Θ - произвольная машина Тьюринга. Упорядочим все ее команды в соответствии с лексикографическим порядком левых частей команд:

$$q_1a_0, q_1a_1, \ldots, q_1a_n, q_2a_0, q_2a_1, \ldots, q_2a_n, \ldots, q_ma_0, q_ma_1, \ldots, q_ma_n .$$

Всего программа машины Тьюринга с внешним алфавитом из n символов и алфавитом внутренних состояний из m символов содержит m(n + 1) команд, которые по указанному выше правилу оказались лексикографически упорядочены следующим образом: $I_1, I_2, \ldots, I_{m(n+1)}$. Данной последовательности команд поставим в соответствие двоичный набор вида:

$$Код(\Theta) = Код(I_1)\ Код(I_2)\ldots\ Код(I_{m(n+1)}) \quad (2).$$

Из указанной процедуры следует, что машина Θ переводит конфигурацию вида $q_1a_1^x$ в конфигурацию вида $q_0a_1^x$, поэтому, представляя натуральное число n в виде a_1^{n+1}, получаем, что Θ вычисляет функцию

$$f(x) = x.$$

Очевидно, что такое кодирование является алгоритмической процедурой. Зная код машины, можно однозначно восстановить ее программу. Для этого нужно выделить все подслова, начинающиеся единицей с каким-либо числом нулей, до следующей единицы. Пятерка таких подслов образует команду.

Номером машины Тьюринга называется натуральное число, которое получается при переводе в десятичную систему счисления двоичный код этой машины. Очевидно, что так как все коды начинаются с единицы, то разным кодам соответствуют разные натуральные числа. Тогда все мыслимые машины Тьюринга можно упорядочить по возрастанию их номеров:

$$T_0, T_1, \ldots, T_n, \ldots \quad (3).$$

Указанное упорядочение является эффективным в том смысле, что существует алгоритм, который по номеру n машины T_n выдает ее код, и, обратно, существует алгоритм, который по заданному коду выдает номер машины.

Нумерация МПД – программ

Определим нумерацию команд, которые может выполнять машина произвольного доступа. Для этого зададим некоторую функцию α, вычисляющую номера команд МПД.

$$\alpha(Z(n)) = 4(n – 1);$$
$$\alpha(S(n)) = 4(n – 1) – 1;$$
$$\alpha(T(m, n)) = 4p(m - 1, n – 1) + 2; \quad (4)$$

$$\alpha(J(m, n, q)) = 4\pi(m, n, q) + 3,$$

где $p(x, y) = 2^x(2y + 1) - 1$ и $\pi(x, y, z) = p(p(x - 1, y - 1), z - 1)$.

Можно показать, что функция α и обратная к ней эффективно вычислимы. Определим теперь номер программы произвольной МПД.

Пусть программа имеет вид $P = I_1 \ldots I_s$. Номер программы $\gamma(P)$ вычисляется следующим образом:

$$\gamma(P) = \tau(\alpha(I_1), \ldots, \alpha(I_s)) \qquad (5),$$

где $\tau(x_1, \ldots, x_s) = 2^{x_1} + 2^{x_1 + x_2 + 1} + 2^{x_1 + x_2 + x_3 + 2} + \ldots + 2^{x_1 + \ldots + x_s + s - 1} - 1$ (6).

Присвоим каждой программе МПД ее номер $\gamma(P)$ и упорядочим номера по возрастанию. Указанное упорядочение также является эффективным, так как по каждой программе позволяет найти ее номер и обратно.

ПРИМЕР.

1. Занумеровать машину Тьюринга с программой Θ:

$$\Theta: \quad q_1 a_0 \to q_2 a_1 Л$$
$$q_1 a_1 \to q_2 a_0 C$$
$$q_2 a_0 \to q_1 a_1 П$$
$$q_2 a_1 \to q_0 a_1 C.$$

2. Занумеровать программу МПД:

$$P: \quad I_1 = S(2)$$
$$I_2 = Z(3)$$
$$I_3 = T(1, 3)$$
$$I_4 = J(1, 1, 1).$$

РЕШЕНИЕ:

1. В программе Θ 4 команды. Сначала найдем коды этих команд согласно таблице 1 правилу (1).

Первая команда $q_1 a_0 \to q_2 a_1 Л$ имеет код $10^7 10^4 10^9 10^6 10^2$; вторая команда $q_1 a_1 \to q_2 a_0 C$ - код $10^7 10^6 10^9 10^4 10^3$; третья команда $q_2 a_0 \to q_1 a_1 П$ - код $10^9 10^4 10^7 10^6 10^1$, и, наконец, последняя команда - $q_2 a_1 \to q_0 a_1 C$ - код $10^9 10^6 10^5 10^6 10^3$. Тогда код машины будет иметь вид:

$$\mathbf{10^7 10^4 10^9 10^6 10^2 10^7 10^6 10^9 10^4 10^3 10^9 10^4 10^7 10^6 10^1 10^9 10^6 10^5 10^6 10^3}.$$

2. Согласно формулам (4), рассчитаем значение α-функции от каждой команды программы P.

$$\alpha(I_1) = \alpha(S(2)) = 4(2 - 1) - 1 = 4 - 1 = \underline{\mathbf{3}};$$
$$\alpha(I_2) = \alpha(Z(3)) = 4(3 - 1) = 4 \cdot 2 = \underline{\mathbf{8}};$$
$$\alpha(I_3) = \alpha(T(1, 3)) = 4p(1 - 1, 3 - 1) + 2 = 4p(0, 2) + 2,$$

так как $p(0, 2) = 2^0(2 \cdot 2 + 1) - 1 = 4$, то $\alpha(I_3) = \alpha(T(1, 3)) = 4 \cdot 4 + 2 = \underline{\mathbf{18}}$.

$$\alpha(I_4) = \alpha(J(1, 1, 1)) = 4\pi(1, 1, 1) + 3.$$

Т. к. $\pi(1, 1, 1) = p(p(1 - 1, 1 - 1), 1 - 1) = p(p(0, 0), 0) = p([2^0(2 \cdot 0 + 1) - 1], 0) =$
$= p(0, 0) = 2^0(2 \cdot 0 + 1) - 1 = 0$, то:

$$\alpha(I_4) = \alpha(J(1, 1, 1)) = 4 \cdot 0 + 3 = \underline{\mathbf{3}}.$$

Согласно формуле (5), номер программы P равен:

$$\gamma(P) = \tau(\alpha(I_1), \alpha(I_2), \alpha(I_3), \alpha(I_4)) = \tau(3, 8, 18, 3).$$

Значение функции τ рассчитаем по формуле (6):

$\tau(3, 8, 18, 3) = 2^3 + 2^{3+8+1} + 2^{3+8+18+2} + 2^{3+8+18+3+3} - 1 = 2^3 + 2^{12} + 2^{31} + 2^{35} - 1.$

Следовательно, $\gamma(P) = 2^3 + 2^{12} + 2^{31} + 2^{35} - 1 = 36507226119.$

Частично-рекурсивные функции

Класс частично рекурсивных функций также был введен в качестве еще одного уточнения понятия алгоритма. Данный класс определяется путем указания конкретных исходных функций, называемых *базисными* и фиксированного множества операций получения новых функций из заданных.

Обозначим через N_0 множество натуральных чисел с нулем. Тогда в качестве базисных функций берутся следующие функции.

1). *Нуль-функция 0(x)*:

$$(\forall x \in N_0) \quad 0(x) = 0 \qquad (1).$$

2). *Функция следования s(x)*:

$$(\forall x \in N_0) \quad s(x) = x + 1 \qquad (2).$$

3). *Функция выбора аргумента* $I_m^n(x_1, x_2, \ldots, x_n)$:

$$(\forall x \in N_0) \quad I_m^n(x_1, x_2, \ldots, x_n) = x_m, \ m \in \{1, 2, \ldots, n\} \qquad (3).$$

Допустимыми операциями над функциями являются операции *суперпозиции* или *подстановки, примитивной рекурсии* и *минимизации.*

Операция суперпозиции.

Пусть заданы одна n-местная функция $g(x_1, x_2, \ldots, x_n)$ и n m-местных функций, зависящих от одних и тех же переменных $f_1(x_1, x_2, \ldots, x_m)$, $f_2(x_1, x_2, \ldots, x_m)$, …, $f_n(x_1, x_2, \ldots, x_m)$. (Этого можно добиться введением фиктивных переменных).

Суперпозицией функций g и $f_1, f_2, \ldots, f_n$ называется функция

$$h(x_1, x_2, \ldots, x_m) = g(f_1(x_1, x_2, \ldots, x_m), f_2(x_1, x_2, \ldots, x_m) \qquad (4),$$

Очевидно, что функция h определена тогда и только тогда, когда определены все функции $f_1, f_2, \ldots, f_n$. Операцию суперпозиции обозначают:

$$h = S(g, f_1, f_2, \ldots, f_n) \qquad (5).$$

Операция примитивной рекурсии.

Пусть заданы n-местная функция $g(x_1, x_2, \ldots, x_n)$ и (n + 2)-местная функция h($x_1, x_2, \ldots, x_n$, y, z). Определим (n + 1)-местную функцию f индуктивно с помощью соотношений:

$$\begin{cases} f(x_1, x_2, \ldots, x_n, 0) = g(x_1, x_2, \ldots, x_n); \\ f(x_1, x_2, \ldots, x_n, y+1) = h(x_1, x_2, \ldots, x_n, y, f(x_1, x_2, \ldots, x_n, y)) \end{cases} \qquad (6).$$

Про функцию f говорят, что она получена рекурсией из функций g и h и обозначают:

$$f = R(g, h).$$

Операция минимизации.

Пусть задана n-местная функция $g(x_1, x_2, \ldots, x_{n-1}, y)$. Зафиксируем набор $x_1, x_2, \ldots, x_{n-1}, x_n$ и рассмотрим уравнение относительно y:

$$g(x_1, x_2, \ldots, x_{n-1}, y) = x_n \qquad (7).$$

Будем решать данное уравнение, последовательно вычисляя

$g(x_1, x_2, \ldots, x_{n-1}, 0)$, $g(x_1, x_2, \ldots, x_{n-1}, 1)$, $g(x_1, x_2, \ldots, x_{n-1}, 2)$ и т.д.

и сравнивая с x_n. Наименьшее y, для которого выполнено (7), обозначим

$$\mu_y = (g(x_1, x_2, \ldots, x_{n-1}, y) = x_n) \qquad (8).$$

Про функцию f говорят, что она получена минимизацией из функции g и обозначают:

$f = \mu_y(g)$.

Функция называется *частично рекурсивной,* если она может быть получена из базисных функций применением конечного числа раз операций суперпозиции, примитивной рекурсии и минимизации.

ПРИМЕР.

Выразить функцию f(x, y) = x + y через базисные: нуль-функцию - 0(x) = 0, функцию следования – s(x) = x + 1 и функцию выбора аргументов - $I_m^n(x_1, x_2, \ldots, x_n) = x_m$.

РЕШЕНИЕ:

Для любых чисел x и y выполняются соотношения:

$x + 0 = x = I_1^2(x, y)$,

$x + (y + 1) = (x + y) + 1$.

Пусть $I_1^2(x, y) = g(x)$, s(z) = z + 1 = (x + y) + 1 = h(x, y, z), т.е. заданы одноместная функция g(x) и трехместная функция h(x, y, z).

Тогда функцию f(x, y) = x + y можно рассматривать как двуместную функцию, удовлетворяющую соотношениям:

$$f(x, 0) = x + 0 = I_1^2(x, y) = g(x),$$
$$f(x, y + 1) = s(z) = h(x, y, f(x, 1)).$$

Следовательно, функция f(x, y) = x + y получена с помощью операции примитивной рекурсии из функций g(x) и h(x, y, z), или, с учетом вышесказанного, из функций $I_1^2(x, y)$ - функция выбора аргумента и s(z) - функция следования.

10. Тематика рефератов

1. Алгоритмы вокруг нас.
2. Основатели теории алгоритмов – Клини, Черч, Пост, Тьюринг.
3. Тезис Черча.
4. Проблема вычислимости математической логике.
5. Нормальные алгоритмы Маркова..
6. Методы разработки алгоритмов.
7. Средства и языки описания (представления) алгоритмов.
8. История формирования «понятия алгоритмов».
9. Известнейшие алгоритмы в истории математики.
10. Нормальные алгоритмы Маркова.
11. Принцип нормализации Маркова.
12. Эквивалентность различных теорий алгоритмов.
13. Алгоритмические проблемы.
14. Теорема Гёделя о неполноте формальной арифметики.