ПРОГРАММИРОВАНИЕ, 2020, № 6, с. 67–72

ТЕОРЕТИЧЕСКИЕ ВОПРОСЫ ПРОГРАММИРОВАНИЯ

УДК 681.3.06

МОДЕЛИРОВАНИЕ МОГОЛЕНТОЧНЫХ МАШИН МИНСКОГО И ТЬЮРИНГА ТРЕХЛЕНТОЧНЫМИ МАШИНАМИ МИНСКОГО

© 2020 г. С. С. Марченков[a],*, С. Д. Макеев[a],**

[a] *МГУ им. М.В. Ломоносова, факультет вычислительной математики и кибернетики, ГСП-1, Ленинские горы, МГУ, д. 1, стр. 52, 119991 Москва, Россия*

**E-mail: ssmarchen@yandex.ru*

***E-mail: sergeymak98@gmail.com*

Поступила в редакцию 24.04.2020 г.
После доработки 06.05.2020 г.
Принята к публикации 17.06.2020 г.

Показано, что k-ленточную машину Минского, работающую с временем $T(n)$, можно промоделировать трехленточной машиной Минского за время, по порядку не превосходящее $T(n)^k \times \log T(n)$. Установлено, что многоленточные машины Тьюринга можно промоделировать трехленточными машинами Минского при “оптимальном” кодировании слов, записанных на лентах.

DOI: 10.31857/S0132347420060059

1. ВВЕДЕНИЕ

В теории алгоритмов широко используется моделирование одних вычислительных устройств другими вычислительными устройствами. Цели моделирования могут быть различными: сравнение вычислительных возможностей рассматриваемых устройств, доказательство существования устройств, обладающих теми или иными свойствами внешней или внутренней памяти, построение устройств с минимально возможными параметрами вычислений и другие.

Некоторые абстрактные вычислительные устройства (часто называемые машинами) в вопросах моделирования принято считать “эталонами”. К ним относятся машины Тьюринга различных типов, машины с произвольным доступом к памяти и машины Минского (см., например, [1–3]). Среди вычислительных устройств, имеющих внешнюю память в виде лент, многоленточные машины Тьюринга обладают, по-видимому, наиболее широкими вычислительными возможностями – в значительной степени за счет организации внешней памяти. Поэтому они довольно часто применяются для формализации различных алгоритмических процессов.

Машины Минского [2, 4, 5] представляют собой универсальные вычислительные устройства, способные вычислять произвольные частично-рекурсивные функции. В теории рекурсивных функций они являются весьма удобным средством для получения результатов сложностного и алгебраического характера [6–10]. Однако, в отличие от многоленточных машин Тьюринга, машины Минского обладают сравнительно небольшими возможностями по части хранения и извлечения информации из внешней памяти.

Одной из сложностных характеристик машины Минского может служить число имеющихся у нее лент (регистров). Известно [2, 4, 5], что на машинах с тремя лентами можно вычислять любые одноместные частично-рекурсивные функции (при стандартной записи аргумента на ленте). Для двухленточных машин это тоже возможно, однако аргумент и значение функции при этом следует кодировать экспоненциальным кодом. На одноленточных машинах Минского можно вычислить только некоторые тривиальные функции.

Таким образом, трехленточные машины Минского по ряду причин представляются одним из “минимальных” эталонных вычислительных устройств, которые в вычислительном отношении интересно использовать для моделирования других вычислительных устройств. Прежде всего, задача о моделировании возникает для самих многоленточных машин Минского, а также для многоленточных машин Тьюринга, вычисляющих одноместные функции. Основная проблема здесь заключается в том, чтобы найти “оптимальный” способ кодирования информации, имеющейся на лентах машины Минского или машины Тьюринга, который будет

доступен для обработки на трехленточных машинах Минского, и оценить время моделирования для такого способа кодирования.

В настоящей работе мы решаем сформулированную задачу: предлагаем некоторый “оптимальный” способ кодирования чисел, записанных на лентах многоленточной машины Минского или многоленточной машины Тьюринга; показываем, как для данного способа можно промоделировать оба вычислительных устройства трехленточными машинами Минского с “минимально возможными” потерями во времени. Для машин Минского этот переход выполняется с полиномиальным замедлением. Отметим, что несмотря на большое число работ, где одни вычислительные устройства моделируются другими вычислительными устройствами, моделирование универсальных вычислительных устройств трехленточными машинами Минского с оценками времени моделирования, насколько нам известно, не проводилось. Полученные в работе результаты можно будет использовать как при моделировании других вычислительных устройств (трехленточными машинами Минского), так и при решении специальных задач, относящихся, например, к вычислениям с полиномиальным временем.

2. ОСНОВНЫЕ ПОНЯТИЯ

Пусть $\mathbb{N} = \{0,1,\ldots\}$. Говоря далее о (натуральных) числах, мы имеем в виду числа из множества $\mathbb{N}$.

Будем рассматривать следующий вариант k-ленточной машины Тьюринга ($k \geq 1$). Машина Тьюринга $\mathcal{T}$ имеет k бесконечных в обе стороны лент, разбитых на клетки. На каждой ленте расположена одна считывающе-записывающая головка. Ленточный алфавит машины $\mathcal{T}$ состоит из символов 0, 1, λ (λ – “пустой” символ). Машина имеет множество (внутренних) состояний $Q = \{q_0, q_1, \ldots, q_r\}$, где q_1 – начальное, а q_0 – заключительное состояния. Функционирование машины $\mathcal{T}$ определяется программой, которая состоит из команд вида

$$a_1 \ldots a_k q_i \to b_1 \ldots b_k D_1 \ldots D_k q_j,$$

где $a_1, \ldots, a_k, b_1, \ldots, b_k \in \{0,1,\lambda\}$, $i \neq 0$ и $D_1, \ldots, D_k$ – символы движения на лентах: L, R, S. Предполагаем, что в программе машины $\mathcal{T}$ нет двух различных команд с одинаковыми левыми частями.

Нас будут интересовать функции одного аргумента $f(n)$, вычислимые на машине $\mathcal{T}$. Предполагаем, что в начальный момент времени на первой ленте машины записано двоичное представление числа n (слева и справа от этой записи лента пустая – заполнена символами λ), все остальные ленты пустые, машина находится в состоянии q_1, а ее головка на первой ленте – в клетке, которая раположена непосредственно справа от клетки, содержащей младший разряд двоичного представления n. Удобно также считать, что в заключительный момент времени результат $f(n)$ в двоичной записи представлен на первой ленте, слева от записи $f(n)$ лента пустая, а головка машины $\mathcal{T}$ на первой ленте находится в (пустой) клетке, примыкающей справа к записи $f(n)$.

Машина Минского представляет собой вариант нестирающей многоленточной машины Тьюринга. Машина Минского $\mathcal{M}$ имеет k односторонних (бесконечных вправо) лент, содержимое которых не меняется в процессе вычисления, и множество состояний $Q = \{q_0, q_1, \ldots, q_r\}$. Концевые клетки лент содержат символ 1, все остальные клетки – символ 0. На каждой из лент находится одна читающая головка. В процессе работы машины $\mathcal{M}$ на каждом шаге вычисления любая из головок может независимым образом сдвигаться на одну клетку влево, вправо или оставаться в той же клетке. Функционирование машины $\mathcal{M}$ определяется программой, которая состоит из команд вида

$$a_1 \ldots a_k q_i \to q_s d_1 \ldots d_k,$$

где $a_1, \ldots, a_k$ – символы 0 или 1, обозреваемые головками машины $\mathcal{M}$ на лентах, $1 \leq i \leq r$ и $d_1, \ldots, d_k$ – символы движения головок на лентах ($d_1, \ldots, d_k \in \{-1, 0, 1\}$). Программа машины $\mathcal{M}$ организована таким образом, что головки не могут сходить с концевых клеток лент.

Машина $\mathcal{M}$ вычисляет (частичную) функцию $f(n)$, если в начальный момент вычисления головка на первой ленте находится в клетке с номером n (концевые клетки имеют номер 0), остальные головки – в клетках с номером 0, а машина – в начальном состоянии q_1. Если значение $f(n)$ определено, то машина через конечное число тактов достигает заключительного состояния q_0, при этом в заключительный момент времени первая головка находится в клетке с номером $f(n)$. Если же значение $f(n)$ не определено, то машина работает неограниченно долго.

3. РЕЗУЛЬТАТЫ

При моделировании одного вычислительного устройства $\mathcal{M}_1$ другим вычислительным устройством $\mathcal{M}_2$ процесс моделирования, как правило, разбивается на три этапа. На первом этапе исход-

ные данные, записанные в языке устройства $\mathcal{M}_1$, переводятся в язык устройства $\mathcal{M}_2$ (будем говорить о кодировании данных). Это может быть, например, переход в другую позиционную систему счисления либо переход к коду, который учитывает определенные параметры моделируемого устройства $\mathcal{M}_1$.

На втором этапе устройство $\mathcal{M}_2$, работая в кодах, по шагам моделирует процесс применения устройства $\mathcal{M}_1$ к входным данным (собственно моделирование). На третьем этапе происходит перевод полученного результата из языка устройства $\mathcal{M}_2$ в язык устройства $\mathcal{M}_1$ (коротко: декодирование). Каждый из этапов имеет свои особенности. Однако первый и третий этапы идейно близки друг к другу и выполняются несколько проще, чем второй этап.

В основе моделирования машин Минского (ММ) и машин Тьюринга (МТ) трехленточными машинами Минского лежит некоторый способ кодирования наборов натуральных чисел. Этот способ не новый, он использовался другими авторами примерно для аналогичных целей (см., например, [11] или [12]).

Пусть $d, n_1, \ldots, n_k$ – натуральные числа, $d \geq 2$ и $a_{1m} \ldots a_{10}$, ..., $a_{km} \ldots a_{k0}$ – представления чисел $n_1, \ldots, n_k$ в системе счисления с основанием d (если длины d-ичных представлений некоторых чисел n_j меньше $m+1$, то соответствующие старшие разряды представлений считаем равными нулю). Набору чисел $(n_1, \ldots, n_k)$ (а точнее, набору d-ичных представлений этих чисел) сопоставим число Code $(n_1, \ldots, n_k)$ с d-ичным представлением

$$a_{km}a_{k-1,m} \ldots a_{1m}a_{k,m-1}a_{k-1,m-1} \ldots a_{1,m-1} \ldots a_{k0}a_{k-1,0} \ldots a_{10}. \quad (1)$$

Как видно, в представлении (1) разряды числа n_1 занимают позиции с номерами $1, k+1, 2k+1, \ldots$, разряды числа n_2 – позиции с номерами $2, k+2, 2k+2, \ldots$ и т.д. Мы будем говорить об i-й "дорожке", занимаемой разрядами числа n_i.

Не вдаваясь в детали, опишем в общих чертах, как трехленточная ММ может совершать некоторые простые действия с числом $n = \text{Code}(n_1, \ldots, n_k)$, опираясь при этом на представление числа n в виде (1). Будем предполагать известным, как на трехленточных ММ можно сравнивать имеющиеся на лентах числа с нулем и вычислять арифметические функции $x+d$, $x-d$ (d – натуральное число, а в случае вычитания выполняется неравенство $x \geq d$), $d \times x$, $[x/d]$, $\text{rm}(x,d)$ ($d \geq 2$ и $\text{rm}(x,d)$ и обозначает остаток от деления x на d).

Определение разряда a_{i0} $(1 \leq i \leq k)$. Число n делим с остатком на d, получаем частное $[n/d]$, которое записываем на вторую ленту, и остаток a_{10}. Если $i = 1$, умножаем $[n/d]$ на d и добавляем к результату a_{10}. Задача решена.

В противном случае записываем число 1 на третью ленту (это делается для того, чтобы далее не "потерять" возможные нулевые разряды $a_{10}, a_{20}, \ldots$), умножаем 1 на d и добавляем к результату найденное число a_{10}. Если уже проделано $j < i$ шагов и на первой ленте образовалось число n', d-ичное представление которого получается из представления (1) удалением последних j разрядов, а на третьей ленте – число n'' с d-ичным представлением $1a_{10} \ldots a_{j0}$, то делим число n' на d с остатком $a_{j+1,0}$, умножаем число n'' на d и добавляем к полученному результату $a_{j+1,0}$. Если $j+1 = i$, то искомый разряд найден. Далее меняем ролями числа n' и n'', дописывая к представлению числа n' удаленные разряды $1a_{10} \ldots a_{j0}$ и добавленный разряд 1, в конце процедуры вычитаем из полученного числа 1 (удаление "лишнего" разряда 1). Если же $j+1 < i$, то продолжаем описанную выше процедуру до вычисления разряда a_{i0}.

Обращение представления (1). Задача состоит в том, чтобы по исходному числу n с d-ичным представлением (1) получить число n', d-ичное представление которого представляет собой обращение представления (1) с добавленной в начало единицей (о роли единицы сказано выше). По существу алгоритм построения числа n' указан выше: необходимо сначала записать на третью ленту единицу, затем умножить ее на d, путем деления числа n на d с остатком, найти разряд a_{10}, добавить его на третью ленту, умножить полученный результат на d и т.д. до тех пор, пока очередное частное от деления не станет равным нулю.

Нетрудно видеть, что выполнение обращения на трехленточной ММ требует по порядку не более $n \times \log n$ шагов: при вычислении очередного разряда представления (1) машине необходимо разделить на d с остатком одно число, не превосходящее n, и умножить на d другое число, не превосходящще n. Поскольку количество разрядов в представлении (1) по порядку равно $\log n$, приходим к верхней оценке времени обращения, по порядку равной $n \times \log n$.

Перейдем к более сложным действиям, выполнимым на трехленточных ММ.

Вычитание единицы (из числа n_i в представлении (1) числа Code($n_1, \ldots, n_k$)). Предполагаем, что в начальный момент времени на первой ленте трехленточной ММ находится число $n = \text{Code}(n_1, \ldots, n_k)$ и $n_i > 0$. Выполняемая задача состоит в том, чтобы на i-й дорожке представления (1) числа n найти первый ненулевой разряд, вычесть из него 1, а все предыдущие нулевые разряды (на этой дорожке) заменить числом $d-1$. Так же, как в задаче обращения (с добавлением единицы в начало обращения), ММ путем последовательного "перекладывания" разрядов представления (1) с первой ленты на третью начинает разыскивать на i-й дорожке первый ненулевой разряд, одновременно заменяя нулевые разряды числом $d-1$. Ввиду условия $n_i > 0$ такой разряд непременно найдется. Чтобы определить, что он находится на i-й дорожке, машине необходимо выполнять действия "по модулю" k: первый разряд на i-й дорожке появится при "обработке" i-го разряда представления (1), второй разряд – при обработке $(k+i)$-го и т.д. Как только нужный разряд будет найден, машина заменит его разрядом, на единицу меньшим, а далее, как в задаче обращения, будет перекладывать разряды представления числа с третьей ленты на первую ленту. В заключение машина вычтет из полученного числа 1.

Поскольку при выполнении операции *вычитание единицы* машине в "самом худшем" случае придется обработать порядка $\log n$ разрядов представления (1), общее время выполнения данной операции по порядку не превосходит величины $n \times \log n$.

Прибавление единицы (к числу n_i в представлении (1) числа Code ($n_1, \ldots, n_k$)). Эта задача является обратной к предыдущей задаче. Поэтому остановимся лишь на некоторых отличительных особенностях. Если среди разрядов числа n_i есть хотя бы один, отличный от $d-1$, то реализуем процедуру, обратную к процедуре *вычитание единицы*. Именно, если разряды $a_{i0}, \ldots, a_{ij}$ равны $d-1$ и $a_{i,j+1} \neq d-1$, то заменяем разряды $a_{i0}, \ldots, a_{ij}$ нулями, а разряд $a_{i,j+1}$ – числом $a_{i,j+1}+1$. Так же поступаем, когда $a_{im} = 0$. Однако если $a_{im} = d-1$, то необходимо добавить нулевые разряды $a_{1,m+1}, \ldots, a_{i-1,m+1}$ и единичный разряд $a_{i,m+1}$.

Удаление младшего разряда (из представления числа n_i в представлении (1) числа Code ($n_1, \ldots, n_k$)). Задача состоит в том, чтобы в представлении (1) разряды $a_{im}, a_{i,m-1}, \ldots, a_{i1}, a_{i0}$ заменить разрядами 0, $a_{im}, \ldots, a_{i2}, a_{i1}$. Задача решается по аналогии с предыдущими задачами, однако теперь основные действия проводятся после того, как на третьей ленте будет получено обращение представления (1). В процессе "перекладывания" разрядов с третьей ленты на первую необходимо разряд a_{im} заменить нулем и запомнить. Через k позиций разряд $a_{i,m-1}$ заменяется разрядом a_{im} и запоминается до момента обработки разряда $a_{i,m-2}$ и т.д. до разряда a_{i0}, который заменяется разрядом a_{i1}.

Добавление младшего разряда (к представлению числа n_i в представлении (1) числа Code ($n_1, \ldots, n_k$)). Хотя эта задача по форме выглядит как обратная к задаче *уаление младшего разряда*, выполняется она несколько иначе. Именно необходимо сразу при получении обращения представления (1) поставить выбранное число a (котором машина хранит во внутренней памяти) вместо разряда a_{i0}, запомнить этот разряд, а далее последовательно заменять разряды $a_{i,j+1}$ разрядами a_{ij}. Исключение составляет случай, когда разряд a_{im} ненулевой. Тогда следует заменить его разрядом $a_{i,m-1}$ и затем (как в задаче *прибавление единицы*) добавить нулевые разряды $a_{1,m+1}, \ldots, a_{i-1,m+1}$ и ненулевой разряд $a_{i,m+1}$, равный a_{im}.

Так же, как при рассмотрении операции *вычитание единицы*, каждую из последних трех операций можно выполнить на трехленточной ММ за время, по порядку не превосходящее $n \times \log n$.

Теорема 1. *Пусть k-ленточная ММ работает с временем $T(n)$, причем $T(n) \geq n$. Тогда ее можно промоделировать трехленточной ММ, работающей с временем, по порядку не превосходящим $T(n)^k \times \log T(n)$.*

Доказательство. Пусть $a_m \ldots a_0$ – двоичное представление числа n. Тогда $n' = \text{Code}(n, 0, \ldots, 0)$ есть число с двоичным представлением

$$a_m 0^{k-1} a_{m-1} 0^{k-1} \ldots 0^{k-1} a_0, \qquad (2)$$

где 0^{k-1} – последовательность из $k-1$ нулей. На первом этапе моделирования трехленточная ММ $\mathcal{M}$ преобразует число n, записанное на ее первой ленте, в число n'.

Это делается примерно так же, как было описано выше при выполнении процедуры *обращение представления*. Именно сначала машина $\mathcal{M}$ с помощью последовательных делений числа n на 2 с остатком образует на третьей ленте число с двоичным представлнием

$$1 a_0 0^{k-1} a_1 0^{k-1} \ldots 0^{k-1} a_m, \qquad (3)$$

которое, за исключением единицы в начале, является обращением представления (2). Затем она (уже без измения разрядов двоичного представления) обращает представление (3) и вычитает из полученного числа 1.

Нетрудно видеть, что весь процесс получения числа Code $(n, 0, \ldots, 0)$ занимает на машине $\mathcal{M}$ время, по порядку не превосходящее n^k, т.е. не превосходящее $T(n)^k$.

Заметим, что для k-ленточной ММ, работающей с временем $T(n)$, в любой момент времени на любой из лент находится число, не превосходящее $T(n)$. Поэтому соответвующая величина Code $(n_1, \ldots, n_k)$ по порядку не превосходит $T(n)^k$. Обращаясь теперь к шагам моделирования, выполняемым машиной $\mathcal{M}$ на втором этапе, на основе указанных выше оценок для выполнения операций *вычитание единицы* и *прибавление единицы* приходим к заключению, что каждый шаг моделирования требует времени по порядку не более $T(n)^k \times \log T(n)$. Поэтому в целом весь второй этап займет по порядку не более $T(n)^{k+1} \times \log T(n)$ шагов.

Третий этап моделирования в значительной степени аналогичен первому этапу. Поэтому без подробных выкладок приводим верхнюю оценку (по порядку) его выполнения на машине $\mathcal{M}$: $T(n)^k \times \log T(n)$. Это завершает доказательство теоремы.

Ниже в теореме 2 мы предполагаем, что n есть величина аргумента (а не длина его двоичной записи).

Теорема 2. *Произвольную k-ленточную МТ, имеющую время работы $T(n)$, где $T(n) \geq n$, можно промоделировать трехленточной машиной Минского за время, по порядку не превосходящее $T(n)^2 \times 3^{2k \times T(n)}$.*

Доказательство. Пусть $\mathcal{T}$ — k-ленточная МТ с ленточным алфавитом $\{0, 1, \lambda\}$, которая работает с временем $T(n)$. Чтобы закодировать содержимое всех лент машины $\mathcal{T}$ одним числом, для любого i $(1 \leq i \leq k)$ разделим i-ю ленту машины $\mathcal{T}$ на две части: левую L_i, которая расположена непосредственно слева от клетки, обозреваемой в данный момент головкой машины, и правую R_i, крайнюю левую клетку которой в этот момент обозревает головка машины. Части L_i сопоставим число l_i, троичное представление которого находится на этой части. При этом считаем, что ленточным символам λ, 0, 1 соответствуют разряды 0, 1, 2 троичного представления l_i (предполагаем, что только конечное число клеток части L_i содержат символы 0 или 1). Аналогичное соглашение действует для части R_i и числа r_i, однако здесь по техническим причинам удобно считать, что младшие разряды части R_i располагаются слева. При этих соглашениях содержимое всех лент машины $\mathcal{T}$ будет кодироваться числом Code $(l_1, r_1, l_2, r_2, \ldots, l_k, r_k)$, в троичном представлении которого разряды числа l_i располагаются в позициях с номерами $2i - 1$, $2i + 2k - 1$, $2i + 4k - 1$, ... (дорожка с номером $2i - 1$), а разряды числа r_i — в позициях с номерами $2i$, $2i + 2k$, $2i + 4k$, ... (дорожка с номером $2i$).

Определим трехленточную ММ $\mathcal{M}$, которая будет моделировать машину $\mathcal{T}$. Первый этап работы машины $\mathcal{M}$ состоит в том, чтобы число n, записанное в начальный момент на первой ленте, перевести в число Code $(l, 0, 0, 0, \ldots, 0, 0)$, где троичная запись числа l получается из двоичной записи числа n заменой разрядов 0 и 1 соответственно разрядами 1 и 2. Это можно выполнить примерно так, как подобные процедуры были изложены выше. Именно необходимо разделить число n на 2 с остатком, к остатку добавить 1 и результат записать на третью ленту. Если $[n/2] = 0$, то машина $\mathcal{M}$ переносит результат с третьей ленты на первую, и первый этап моделирования завершен. В противном случае полученное на третьей ленте число умножается на 3^{2k-1}. Тем самым машина $\mathcal{M}$ выполнила "обработку" первого разряда двоичного представления числа n (умножение на 3^{2k-1} соответствует в данном случае постановке $2k - 1$ нулей в качестве первых разрядов чисел $r_1, l_2, r_2, \ldots, l_k, r_k$).

Далее машина $\mathcal{M}$ переходит к аналогичной обработке второго разряда двоичного представления n: делит число $[n/2]$ на 2 с остатком, умножает число с третьей ленты на 3, добавляет к нему остаток от деления плюс 1, умножает полученный результат на 3^{2k-1} и так далее. После завершения обработки последнего разряда двоичного представления n машина $\mathcal{M}$ осуществляет, как это делалось выше, преобразование полученного числа в число с "обращенным" троичным представлением.

Нетрудно видеть, что машина $\mathcal{M}$ затратит на выполнение первого этапа моделирования по порядку не более $3^{2k \cdot \log_2 n} \times \log n$ тактов.

Теперь машина $\mathcal{M}$ будет моделировать каждый шаг работы машины $\mathcal{T}$. Чтобы это выполнить, машине $\mathcal{M}$ необходимо определить все младшие разряды в троичных представлениях чисел $r_1, \ldots, r_k$. Это делается на основе процедуры *определение разряда a_{i0}*, описанной выше. Вычислив эти раз-

ряды, машина $\mathcal{M}$ определит команду машины $\mathcal{T}$, выполняемую в данный момент ("текущее" состояние машины $\mathcal{T}$ машина $\mathcal{M}$ хранит во внутренней памяти). Теперь машине $\mathcal{M}$ необходимо выполнить требуемые преобразования: замена младших разрядов в представлениях чисел $r_1,\ldots,r_k$, удаление младшего разряда из представления числа l_i и добавление младшего разряда к представлению числа r_i (если машина $\mathcal{T}$ сдвигает i-ю головку влево) и, наоборот, удаление младшего разряда из представления числа r_i и добавление младшего разряда к представлению числа l_i (если машина $\mathcal{T}$ сдвигает i-ю головку вправо).

Замена младших разрядов выполняется так же, как процедура *определение разряда* a_{i0}, описанная выше. Процедуры удаления и добавления младших разрядов также приведены выше. Нетрудно видеть, что все три процедуры машина $\mathcal{M}$ может выполнить за время, по порядку не превосходящее величины $n\times\log n$, где n – код набора $(l_1,r_1,l_2,r_2,\ldots,l_k,r_k)$.

Данная оценка времени моделирования одного шага работы машины $\mathcal{T}$ позволяет оценить в целом все время, затрачиваемое машиной $\mathcal{M}$ на втором этапе моделирования. В самом деле, если машина $\mathcal{T}$ работает с временем $T(n)$, то в течение всего вычисления длина троичной записи на любой из "полулент" также не превосходит $T(n)$. Отсюда вытекает верхняя оценка $3^{T(n)}$ для чисел, записанных на этих "полулентах". Следовательно, в процессе вычисления величина n по порядку будет не больше, чем $3^{2k\times T(n)}$. Учитывая полученную выше оценку $n\times\log n$ времени моделирования одного шага работы машины $\mathcal{T}$ и общее время $T(n)$ ее работы, приходим к верхней оценке (по порядку) вида $T(n)^2\times 3^{2k\cdot T(n)}$.

Для завершения доказательства теоремы осталось рассмотреть третий этап моделирования машины $\mathcal{T}$. По сути он аналогичен первому этапу моделирования, однако здесь надо учесть, что в начале третьего этапа на "входе" машины $\mathcal{M}$ имеется величина порядка $3^{2k\cdot T(n)}$. Так же, как на первом этапе моделирования, время работы машины $\mathcal{M}$ можно ограничить по порядку величиной $3^{2k\times T(n)}$, умноженной на логарифм этой величины, что по порядку не превосходит величины $T(n)^2\times 3^{2k\times T(n)}$.

4. БЛАГОДАРНОСТИ

Работа выполнена при частичной поддержке Российского фонда фундаментальных исследований (проект 19-01-00200).

СПИСОК ЛИТЕРАТУРЫ

1. *Марченков С.С., Савицкий И.В.* Машины в теории вычислимых функций. М.: МАКС Пресс, 2018.
2. *Мальцев А.И.* Алгоритмы и рекурсивные функции. М.: Наука, 1986.
3. *Clote P.* Computation models and functional algebras // Handbook of Computability Theory. Elsevier Science B.V., 1999. P. 589–681.
4. *Minsky M.L.* Recursive unsolvability of Post's "TAG" and topics in theory of Turing machines // Ann. Math. 1961. V. 74. P. 437–455.
5. *Минский М.* Вычисления и автоматы. М.: Мир, 1971.
6. *Марченков С.С.* Базисы по суперпозиции в классах рекурсивных функций // Математические вопросы кибернетики. Вып. 3. 1991. С. 115–139.
7. *Волков С.А.* Пример простой квазиуниверсальной функции в классе $\mathscr{E}^2$ иерархии Гжегорчика // Дискретная математика. 2006. Т. 18. № 4. С. 31–44.
8. *Волков С.А.* Конечная порождаемость некоторых групп рекурсивных перестановок // Дискретная математика. 2008. Т. 20. № 4. С. 61–78.
9. *Марченков С.С.* О сложности класса $\mathscr{E}^2$ Гжегорчика // Дискретная математика. 2010. Т. 22. № 1. С. 5–16.
10. *Марченков С.С.* Представление функций суперпозициями. М.: КомКнига, 2010.
11. *Hennie F.C., Stearns R.E.* Two-tape simulation of multitape Turing machines // Journal of ACM. 1966. V. 13. № 4. P. 533–546.
12. *Яблонский С.В.* Введение в дискретную математику. М.: Наука, 1979.