УДК 510.51 Вестник СПбГУ. Сер. 1. Т. 1 (59). 2014. Вып. 3

ПОЛНАЯ АЛГОРИТМИЧЕСКАЯ ДООПРЕДЕЛИМОСТЬ ЛЮБЫХ АЛГОРИТМОВ, РАБОТАЮЩИХ НА ОГРАНИЧЕННОЙ ПАМЯТИ

Н. К. Косовский, Т. М. Косовская

С.-Петербургский государственный университет,
Российская Федерация, 199034, Санкт-Петербург, Университетская наб., 7-9

Доказывается, что по всякой машине Тьюринга M, использующей память, не превосходящую заданной конструируемой по памяти функции s от длины записи исходных данных n, можно построить применимую к любым исходным данным машину Тьюринга M_1, являющуюся продолжением машины Тьюринга M.

Теорема 1. *По всякой q-ленточной машине Тьюринга M, использующей память, не превосходящую заданной конструируемой по памяти функции s от длины записи исходных данных n, можно построить применимую к любым исходным данным $q+1$-ленточной машину Тьюринга M_1, являющуюся продолжением машины Тьюринга M одной и той же произвольной наперёд заданной константой.*

Утверждение 1. *Для каждого внешнего и внутреннего алфавита машины Тьюринга M память, используемая всюду применимой $q+1$-ленточной машиной M_1, являющаяся продолжением q-ленточной машины M из условия теоремы, не превосходит линейной функции от размера памяти, используемой машиной M.*

Классы **FP-SPACE** и **P-SPACE** расширяются до классов **pFP-SPACE** и **pP-SPACE** соответственно, то есть до классов частичных (partial) функций и предикатов, вычисляемых на машине Тьюринга с памятью, не превосходящей полинома от длины записи исходных данных.

Утверждение 3. *Для каждой функции из* **pFP-SPACE** *задача проверки её применимости к данным принадлежит* **P-SPACE**.

Однако имеет место теорема.

Теорема 3. *Пусть M — программа машины Тьюринга из класса* **pP-SPACE**, *X — её исходные данные. Задача проверки по произвольным M и X применимости M к X не принадлежит классу* **P-SPACE**.

Теорема 4. *Каждая частичная PAM программа P, работающая на битовой памяти, размер которой считается с логарифмическим весовым критерием и ограниченным сверху всюду применимой алгоритмической функцией s от длины записи исходных данных n, может быть доопределена заранее выбранной константой до всюду применимой PAM программы P_1, работающей на битовой памяти, размер которой выражается линейной функцией от $s(n)$. При этом коэффициенты этой линейной функции зависят от параметров машины P.*

Следствием этой теоремы является аналогичное утверждение относительно РАСП программ. Библиогр. 5 назв.

Ключевые слова: машина Тьюринга, PAM программа, РАСП программа, проблема остановки алгоритма, классы **P**, **FP**, **P-SPACE**,**FP-SPACE**, **LIN-SPACE**, **FLIN-SPACE**.

Введение

Широко известны теоремы об алгоритмической неразрешимости проблемы применимости алгоритмов а данным, а также о непродолжимости некоторых алгоритмов (например, универсального алгоритма) до всюду применимых [1]. Этот факт может быть сформулирован как невозможность алгоритмической проверки того, является ли алгоритм применимым к любому набору исходных данных.

Ещё на заре развития теории сложности алгоритмов стало хорошим тоном вместе с разработкой алгоритма устанавливать его вычислительную сложность. Для машины Тьюринга (классического математического понятия алгоритма) это оценки числа её шагов (временная сложность) и количество использованных ячеек (ёмкостная

сложность) как функции от длины записи исходных данных. Если получение оценок числа шагов зачастую связано с определёнными трудностями (необходимо знать, что мащина Тьюринга закончит свою работу), то оценка количества использованных ячеек, как правило, не вызывает таких трудностей.

Более того, широко известен факт, что если машина Тьюринга, работающая в алфавите $A = \{a_1, \ldots, a_l\}$ и имеющая k состояний, использует s ячеек и останавливается после совершения t шагов, то $t \leq l^s ks$. Это означает, что если машина Тьюринга, использующая s ячеек при работе с данными X, совершает по крайней мере $l^s ks + 1$ шаг, то она не применима к данным X.

Ниже в статье доказывается, что по всякой машине Тьюринга M, использующей память, не превосходящую заданной конструируемой по памяти функции s от длины записи исходных данных n, можно построить применимую к любым исходным данным машину Тьюринга M_1, являющуюся продолжением машины Тьюринга M.

1. Используемые определения

Приведём некоторые известные определения, используемые ниже в статье.

Определение [2]. *Функция $s(n)$ называется конструируемой по памяти, если некоторая детерминированная машина Тьюринга, начав работу над данными длины n, поместит специальный маркер на $s(n)$-ю клетку одной из своих лент, просмотрев не более $s(n)$ клеток на каждой ленте.*

Очевидно, что всякая конструируемая по памяти функция всюду определена.

Самые обычные функции, например, полиномы с целыми коэффициентами, 2^n, $n!$, $\lceil n \log(n+1) \rceil$ конструируемы по памяти [2].

Это определение можно обобщить на алгоритмические модели РАМ и РАСП, заменив слова «детерминированная машина Тьюринга» на «РАМ» или «РАСП» соответственно, а слова «поместит специальный маркер на $s(n)$-ю клетку одной из своих лент» на «поместит двоичное число, записанное как 1 и $s(n)$ нулей (то есть $2^{s(n)}$ в один из своих регистров».

Определение [1, 2, 3]. *Класс* **FP** *(класс* **P***) — это класс всех всюду определённых функций (соответственно предикатов), вычислимых на машине Тьюринга за число шагов, не превосходящее полинома от длины записи исходных данных.*

Определение [1, 2, 3]. *Класс* **FP-SPACE** *(класс* **P-SPACE***) — это класс всех всюду определённых функций (соответственно предикатов), вычислимых на машине Тьюринга с памятью, не превосходящей полинома от длины записи исходных данных.*

Определение [1, 2, 3]. *Класс* **FLIN-SPACE** *(класс* **LIN-SPACE***) — это класс всех всюду определённых функций (соответственно предикатов), вычислимых на машине Тьюринга с памятью, не превосходящей линейной функции от длины записи исходных данных.*

Отметим, что размер памяти многоленточной машины Тьюринга вычисляется как максимальное по всем лентам число ячеек, используемых машиной во время её работы. Память же РАМ считается как суммарное по всем регистрам число символов, записанных в этих регистрах во время её работы.

2. Алгоритмическая доопределимость машин Тьюринга, работающих на ограниченной пямяти

Определение. *Машина Тьюринга M_1 называется продолжением машины Тьюринга M одной и той же произвольной наперёд заданной константой z, если для любых исходных данных, к которым применима машина M, машина M_1 тоже применима к ним и результаты их работы совпадают. Для остальных исходных данных, к которым применима M_1, результатом её работы является константа z.*

Теорема 1. *По всякой q-ленточной машине Тьюринга M, использующей память, не превосходящую заданной конструируемой по памяти функции s от длины записи исходных данных n, можно построить применимую к любым исходным данным $q+1$-ленточной машину Тьюринга M_1, являющуюся продолжением машины Тьюринга M одной и той же произвольной наперёд заданной константой.*

Доказательство. Пусть M работает в алфавите из l ($l \geq 3$) букв и имеет k состояний. Количество слов в алфавите из l букв, которые можно записать в $s(n)$ ячейках, равно $l^{s(n)}$. Головки q-ленточной машины Тьюринга M могут находиться в $s(n)^q$ позициях. При каждой позиции головки внутри каждого слова машина Тьюринга может находиться в одном из k состояний. Следовательно, на памяти размером $s(n)$ может быть не более $l^{s(n)}s(n)^q k$ различных конфигураций.

Если M завершает работу над данными длины n, то число шагов её работы не превосходит $l^{s(n)}s(n)^q k = l^{s(n)}l^{q\log_l s(n)}k \leq m^{s(n)+\lceil q\log_l s(n)\rceil+1}$, где $m = \max\{l,k\}$. В противном случае M «зацикливается» и не применима к этим исходным данным.

Построим $(q+1)$-ленточную машину Тьюринга M_1, которая на первых лентах моделирует работу машины M, а на последних подсчитывает число шагов машины M.

Для этого на дополнительной ленте выписывается число $m^{s(n)+\lceil q\log_l s(n)\rceil+1}$ в $(m+1)$-ичной системе счисления, состоящее из $s(n)+\lceil q\log_l s(n)\rceil+1$ «цифр», каждая из которых совпадает с m. Совместно с выполнением на первых q лентах одной команды машины M на дополнительной ленте машина M_1 вычитает 1 и проверяет, равно ли оставшееся число нулю или нет. Если число не равно нулю, то выполняется команда машины M_1, соответствующая очередной команде машины M.

Машина Тьюринга M_1 приходит в заключительное состояние тогда и только тогда, когда либо машина M пришла в заключительное состояние, либо на дополнительной ленте записано число 0.

При этом если машина M пришла в заключительное состояние, то на выходной ленте машины M_1 записан результат работы исходной машины. Если на дополнительной ленте записано число 0, то машина M_1 записывает на выходную ленту заранее заданное значение, доопределяющее M. Следовательно, $(q+1)$-ленточная машина M_1 является всюду применимым продолжением машины M. ■

Следствие теоремы 1. *По всякой 1-ленточной машине Тьюринга M, использующей память, не превосходящую заданной конструируемой по памяти функции s от длины записи исходных данных n, можно построить применимую к любым исходным данным 1-ленточную машину Тьюринга M_1, являющуюся продолжением машины Тьюринга M одной и той же произвольной наперёд заданной константой.*

Доказательство. На основании теоремы 1 по машине Тьюринга M из условия следствия теоремы можно построить 2-ленточную машину Тьюринга M_2, являющуюся её всюду применимым продолжением. Поскольку по всякой многоленточной машине Тьюринга M_2 можно построить моделирующую её одноленточную машину Тьюринга M_1, то следствие теоремы доказано. ■

Очевидно, что число шагов работы так построенного всюду применимого продолжения машины Тьюринга существенно превышает число шагов исходной машины Тьюринга. Это связано с тем, что её программа, во-первых, должна исполнять программу вычисления функции s, во-вторых, на вспомогательной ленте должно быть записано слово длиной $s(n)$ и, в-третьих, после каждого выполнения команды программы машины M на первых лентах и вычитания 1 из числа на вспомогательной ленте в случае равенства 0 полученной цифры необходима проверка наличия ненулевой цифры в соседнем разряде.

Кроме того, число команд всюду применимого продолжения машины Тьюринга также увеличивается. Следующая теорема даёт ответ на то, алгритм из какого класса может строить программу всюду применимого продолжения по программе исходной машины и программе алгоритмической функции s, задающей верхнюю границу используемой памяти.

Ниже будет рассматриваться модель многоленточной машины Тьюринга [2], у которой имеется одна входная лента для записи исходных данных и одна выходная для записи результата. Остальные ленты используются для промежуточных вычислений.

Теорема 2. *Для каждого внешнего и внутреннего алфавита машины M из условия теоремы 1 существует алгоритм из класса* $\mathbf{FP} \cap \mathbf{FLIN\text{-}SPACE}$*, преобразующий запись программы машины M и запись программы для всюду применимой функции, вычисляющей $s(n)$, в запись программы машины M_1, являющейся всюду применимым продолжением машины M одной и той же произвольной заданной константой.*

Доказательство. Пусть машина Тьюринга M работает в алфавите из l букв и использует k состояний. Не умаляя общности будем считать, что программа для вычисления $s(n)$ производит вычисления в двоичной системе счисления.

При своей работе программа машины M_1 сначала производит запись числа $m^{s(n)+\lceil q \log_l s(n)\rceil+1} - 1$ в $(m+1)$-ичной системе счисления (то есть $s(n) + \lceil q \log_l s(n)\rceil + 1$ «цифр» m) на дополнительную ленту, а затем моделирует работу исходной машины Тьюринга. Для этого достаточно

a) на $q+1$-й ленте поставить маркер в $s(n) + \lceil q \log_l s(n)\rceil + 1$-й ячейке;

b) на $q+1$-й ленте записать $s(n) + \lceil \log_l s(n)\rceil + 1$ «цифр» m;

c) каждой команде q-ленточной машины Тьюринга M поставить в соответствие $m+2$ команды $q+1$-ленточной машины, которые осуществляют исходную команду, вычитают единицу из «цифры» $m+1$-ичной записи числа на $q+1$-й ленте и в случае, если полученная «цифра» равна 0, то проверяют, не равно ли нулю всё число на этой ленте.

Оценим число команд, необходимых для написания программы машины M_1 по программам машин M и s.

При написании группы команд из пункта *a* число команд равно числу команд в машине Тьюринга для вычисления s, сложенное с числом команд, не превосходящим числа команд программы, переводящей число из унарной системы счисления в двоичную, плюс число команд программы, осуществляющей сложение двух чисел в двоичной системе счисления.

Сначала переписывается программа, полученная на основе программы для вычисления s (выполняемая на $q+1$-ой ленте), которая вместо маркера записывает значение функции $s(n)$ в унарной системе счисления.

Значение унарной записи числа $\lceil \log_l s(n)\rceil$ вычисляется с помощью программы, полученной из программы перевода числа из унарной системы счисления в l-ичную, в

которой вместо каждой цифры l-ичной системы пишется цифра 1. Количество шагов написания этой части программы для M_1 не превышает $2C_1$.

Следовательно, число шагов написания части программы, реализующей пункт a программы для M_1, линейно относительно количества команд в s.

При написании группы команд для реализации пункта b число команд равно двум. При этом в программе вместо унарной единицы в программе ставится «цифра» m.

Следовательно, число шагов написания части, реализующей пункт b программы для M_1, является константой.

При написании группы команд из пункта c число команд равно числу команд в машине M, умноженное на m (здесь $m = \max(l, k)$).

Следовательно, число шагов написания той части части программы M_1, которая реализует пункт d программы для $q+1$-ленточной машины M_1, является линейным относительно длины записи исходных данных M.

Сложив оценки числа шагов написания частей программы M_1 в пунктах a, b и c получим линейную относительно числа команд в s и M верхнюю оценку числа шагов написания программы для $q+1$-ленточной машины M_1. ■

Из анализа доказательства теоремы 2 следует, что всюду применимое продолжение M_1 машины Тьюринга M использует дополнительную память только для вычисления на вспомогательной ленте двоичных чисел n и $s(n) + \lceil q \log_l s(n) \rceil + 1$, а также $l^{s(n)+\lceil q \log_l s(n) \rceil + 1} - 1$ в $m+1$-ичной системе счисления. Для этих вычислений требуется память, линейная относительно $s(n)$. Поэтому верно следующее.

Утверждение 1. *Для каждого внешнего и внутреннего алфавита машины Тьюринга M память, используемая всюду применимой $q+1$-ленточной машиной M_1, являющаяся продолжением q-ленточной машины M из условия теоремы 1, не превосходит линейной функции от размера памяти, используемой машиной M.*

Замечание. Анализ доказательства теоремы 2 позволяет утверждать, что число шагов таким образом построенной $q+1$-ленточного всюду применимого продолжения q-ленточной машины Тьюринга M, использующей память, не превосходящую заданной алгоритмической функции s от длины записи исходных данных n, экспоненциально относительно длины записи значения $s(n)$, где n — длина записи исходных данных для машины M.

Всюду применимые алгоритмы могут быть использованы для построения формул первого порядка (см., например, [4]) в выбранной сигнатуре. Если в таких формулах используются предметные переменные только для слов заранее ограниченной длины, то каждая такая формула задаёт алгоритм, являющийся предикатом. Такая конструкция, по существу использующая только ограниченные кванторы, может дополнительно прояснять логическую структуру вычисления этого предиката.

Определение. *Под* **FP-SPACE**-*формулой с ограниченными кванторами понимаем формулу, в которой все вхождения квантора Q имеют вид $Qx_{||x|| \le t}$, где t — терм, не содержащий x и содержащий только функции из класса* **FP-SPACE**, *$||x||$ — длина записи x.*

При этом $\forall x_{||x|| \le t} P \Leftrightarrow \forall x(||x|| \le t \Rightarrow P)$, $\exists x_{||x|| \le t} P \Leftrightarrow \exists x(||x|| \le t \ \& \ P)$. При замене ограниченных кванторов с помощью выписанных равносильностей получается формула, все свободные переменные которой, выписанные в лексикографическом порядке, являются аргументами предиката, задаваемого такой формулой.

Так как каждый ограниченный квантор можно реализовать в виде цикла, то очевидно утверждение о том, что размер памяти так заданного предиката не превосходит некоторого полинома от длины записи исходных данных.

Утверждение 2. *Пусть все предикаты сигнатуры принадлежат классу* **P-SPACE**, *а все функции этой сигнатуры принадлежат классу* **FP-SPACE**. *Тогда любая* **FP-SPACE**-*формула первого порядка в этой сигратуре с предметными переменными для слов, длина которых ограниченна сверху, задаёт всюду применимый предикат из класса* **P-SPACE**.

Сказанное позволяет достаточно эффективно (посредством предикатов из класса **P-SPACE**) описывать некоторые свойства предикатов из класса **P-SPACE** и функций из класса **FP-SPACE**.

Расширим классы **FP-SPACE** и **P-SPACE** до классов **pFP-SPACE** и **pP-SPACE** соответственно, то есть до классов частичных (partial) функций и предикатов, вычисляемых на машине Тьюринга с памятью, не превосходящей полинома от длины записи исходных данных.

Из только что доказанных теорем и следующей теоремы 3 следует, что задача доопределения одной частичной функции из класса **pFP-SPACE** любой наперёд заданной константой является более простой задачей, чем общая задача применимости произвольных функций из **pFP-SPACE** к произвольным исходным данным.

Утверждение 3. *Для каждой функции из* **pFP-SPACE** *задача проверки её применимости к данным принадлежит* **P-SPACE**.

Доказательство. Наличие двух всюду применимых продолжений f_{c_1} и f_{c_2} одной и той же функции f с помощью различных констант c_1 и c_2 соответственно позволяет средствами предикатов из **P-SPACE** легко описывать характеристическую функцию завершаемости работы исходной функции с помощью равносильности

$$!f(x) \Leftrightarrow f_{c_1}(x) = f_{c_2}(x).$$

Осталось применить утверждение 1. ■

Однако имеет место теорема.

Теорема 3. *Пусть M — программа машины Тьюринга из класса* **pP-SPACE**, *X — её исходные данные. Задача проверки по произвольным M и X применимости M к X не принадлежит классу* **P-SPACE**.

Доказательство. Доказательство проводится методом от противного и практически повторяет классическое доказательство алгоритмической неразрешимости задачи самоприменимости алгоритма.

Пусть задача из формулировки теоремы принадлежит **P-SPACE**. Тогда задача «По записи $\#M$ программы M проверить, что $\neg!M(\#M)$» принадлежит **P-SPACE**. То есть предикат H, решающий эту задачу, принадлежит **P-SPACE**.

$$H \in \textbf{P-SPACE} \;\&\; \forall M_{M \in \textbf{pP-SPACE}} \left(H(\#M) = 1 \Leftrightarrow \neg!M(\#M)\right).$$

Так как **P-SPACE** $\subset$ **pP-SPACE**, то верно

$$H \in \textbf{pP-SPACE} \;\&\; \forall M_{M \in \textbf{pP-SPACE}} \left(H(\#M) = 1 \Leftrightarrow \neg!M(\#M)\right).$$

Взяв в качестве M функцию H имеем

$$H \in \textbf{pP-SPACE} \;\&\; \left(H(\#H) = 1 \Leftrightarrow \neg!H(\#H)\right).$$

Учитывая, что из $H(\#H) = 1$ следует $!H(\#H)$, приходим к противоречию

$$H \in \textbf{pP-SPACE} \;\&\; \left(!H(\#H) \Leftrightarrow \neg!H(\#H)\right). \quad ■$$

3. Алгоритмическая доопределимость РАМ и РАСП программ, работающих на ограниченной пямяти

Рассмотрим более современные, чем машины Тьюринга, модели вычисления, названные РАМ (Равнодоступная Адресная машина, RAM — Random Access Machine) и РАСП (Равнодоступная Адресная машина С хранимой Программой, RASP — Random Access Stored Program) [2]. Эти модели можно рассматривать как существенное расширение понятия машины Тьюринга с неограниченным числом лент, если считать, что каждая пара регистров моделирует одну двустороннюю ленту машины Тьюринга. При этом достаточно оперировать только остатками от деления на произвольное, но одно и то же во всех операциях, натуральное число, большее двух. Команда косвенной адресации в РАМ организует взаимодействие между некоторыми лентами такого расширения понятия машины Тьюринга.

Размером памяти с логарифмическим весовым критерием РАМ программы при её работе с исходными данными X называется максимальная по всем шагам вычисления сумма длин содержимых всех регистров.

Назовём РАМ или РАСП программу частичной, если она может не завершать свою работу над какими-либо исходными данными.

Теорема 4. *Каждая частичная РАМ программа P, работающая на битовой памяти, размер которой считается с логарифмическим весовым критерием и ограниченным сверху конструируемой по памяти функцией s от длины записи исходных данных n, может быть доопределена заранее выбранной константой до всюду определённой РАМ программы P_1, работающей на битовой памяти, размер которой выражается линейной функцией от $s(n)$. При этом коэффициенты этой линейной функции зависят от параметров машины P.*

Доказательство. Пусть L — число команд в исходной программе, R — максимальное лоличество регистров, задействованных при работе программы P.

Количество различных конфигураций памяти РАМ P, с длиной записи содержимого битовых регистров не превосходящей $s(n)$, не превосходит $2^{(R+1)s(n)}$. К содержимому каждого регистра может быть применена одна из L команд программы. Таким образом, число шагов РАМ, заканчивающей работу над исходными данными длины n и битовая память которой ограничена сверху функцией $s(n)$, не может превышать $2^{(R+1)s(n)}L$. Следовательно, если РАМ P при работе с исходными данными длины n совершила $2^{(R+1)s(n)}L+1$ шаг, то она не применима к этим исходным данным (зацикливается).

РАМ программа P_1 выписывает в $R+1$-ом регистре число $2^{(R+1)s(n)}L$ и после выполнения каждой команды РАМ P вычитает единицу из этого числа и проверяет, не равно ли содержимое $R+1$-го регистра нулю.

P_1 заканчивает работу над исходными данными тогда и только тогда, когда либо P заканчивает работу над этими исходными данными (в этом случае результат работы P_1 совпадает с результатом работы P), либо содержимое $R+1$-го регистра равно нулю (в этом случае в выходной регистр записывается выбранная константа).

Так построенная РАМ P_1 требует дополнительную память для вычисления значений функции $s(n)$ и выражения $2^{(R+1)s(n)}L$, размер которой линеен относительно $s(n)$. ■

Следствие теоремы 4. *Каждая частичная РАСП программа P, работающая на битовой памяти, размер которой считается с логарифмическим весовым критерием и ограничены сверху конструируемой по памяти функцией s от длины записи*

исходных данных n, может быть доопределена заранее выбранной константой до РАСП программы P_1, работающей на битовой памяти, размер которой выражается линейной функцией от $s(n)$.

Следствие очевидно, так как по всякой РАМ программе можно построить эквивалентную ей РАСП программу, временная сложность и сложность по памяти которой отличаются от соответствующих сложностей исходной в константу раз [Ахо].

Заключение

Традиционное математическое обозначение вычисления значения функции от её аргументов (за исключением отдельных особых случаев, например, деления на ноль) подразумевает всегда успешное завершение этого вычисления.

С другой стороны, в математической теории алгоритмов обозначение значения алгоритма от исходных данных подразумевает, что алгоритм может зацикливаться (не завершаться). Поэтому обозначение $A(X)$ для алгоритма A не может использоваться для записи логико-математических формул первого порядка в сигнатуре, содержащей A, которыми описываются те или иные свойства алгоритмов и отношения между ними. Использование знака применимости и условного равенства выводит за пределы традиционной классической логики первого порядка.

Однако, как доказано в настоящей работе, для важных практически вычислимых алгоритмов, работающих в условиях функционально ограниченнной памяти (полиномиальной или превосходящей её), при линейном расширении этой памяти алгоритм может быть доопределён любой наперёд заданной константой до всюду применимого.

Такие всюду определённые продолжения алгоритмов уже можно применять в сигнатуре логико-математических формул первого порядка.

Конструкция, близкая к программе P_1 в теореме 4, была предложена в [5] для языка Паскаль.

Литература

1. *Du D. Z., Ko K. I.* Theory of Computational Complexity. A Wiley-Interscience Publication. John Wiley & Sons, Inc. 2000.

2. *Ахо А., Хопкрофт Дж., Ульман Дж.* Построение и анализ вычислительных алгоритмов. М.: Мир, 1979. (Aho A. V., Hopcroft J. E., Ullman J. D. The design and analysis of computer algorithms.)

3. *Гэри М., Джонсон Д.* Вычислительные машины и труднорешаемые задачи. М.: Мир, 1982. (Garey M. R., Johnson D. S. Computers and Intractability: A Guide to the Theory of NP-Completeness.)

4. *Косовский Н. К.* Элементы математической логики и её приложения к теории субрекурсивных алгоритмов. Л.: Изд-во Ленингр. ун-та, 1981. 192 с.

5. *Косовский Н. К., Фам Тхань Лам.* Оценка памяти, необходимой для исключения бесконечного зацикливания в Паскаль-программах, выполняемых на компьютере // Материалы XVI Международной школы-семинара «Синтез и сложность управляющих систем» (СПб., 26–30 июня 2006 г.) М.: Изд-во мех-мат ф-та МГУ, 2006. С. 53–54.

Статья поступила в редакцию 27 марта 2014 г.

Сведения об авторах

Косовский Николай Кириллович — доктор физико-математических наук, профессор; kosov@nk1022.spb.edu

Косовская Татьяна Матвеевна — доктор физико-математических наук, доцент; kosovtm@gmail.com

TOTAL ALGORITHMIC EXTENSION OF AN ALGORITHM RUNNING ON THE BOUNDED SPACE

Nikolay K. Kosovskiy, Tatiana M. Kosovskaya

St.Petersburg State University, Universitetskaya nab., 7-9, St.Petersburg, 199034, Russian Federation;
kosov@nk1022.spb.edu, kosovtm@gmail.com

It is proved that for every Turing machine M using space not greater than a done space constructible function s of the input data length n there exists a Turing machine M_1 which is a total extension of M.

Theorem 1. *For every q-tape Turing machine M using space not greater than a done space constructible function s of the input data length n there exists a q-tape Turing machine M_1 which is a total extension of M by means of a fixed constant.*

Proposition 1. *For every exterior and interior alphabet of a Turing machine M the space used by a total $q+1$-tape Turing machine M_1 from theorem 1 is not greater than a linear function of the space used by the Turing machine M.*

Classes **FP-SPACE** and **P-SPACE** are extended up to the classes **pFP-SPACE** and **pP-SPACE** respectively, i.e. up to the classes of all partial functions and predicates computable by a Turing machine with the space not greater than a polynomial under the input data length.

Proposition 3. *For every function from* **pFP-SPACE** *its stop problem belongs to* **P-SPACE**.

Nevertheless the following theorem is valid.

Theorem 3. *Let a Turing machine from the class* **pP-SPACE** *M and a word X be input data for the following problem. The problem of applicability of M to X does not belong to the class* **P-SPACE**.

The following theorem uses the fact that a RAM-program transforms nonnegative integers in binary notation.

Theorem 4. *Every partial RAM-program P with logarithmic space bounded by a done space constructible function s of the input data length n may be extended by means of a fixed constant up to a total RAM-program P_1 with logarithmic space bounded by a linear function of $s(n)$. Coefficients of such a linear function depend of the parameters of the program P.*

The analogous assertion according a RASP-program is a corollary of this theorem. Refs 5.

Keywords: Turing machine, RAM program, RASP program, stop-problem, classes **P**, **FP**, **P-SPACE**, **FP-SPACE**, **LIN-SPACE**, **FLIN-SPACE**.