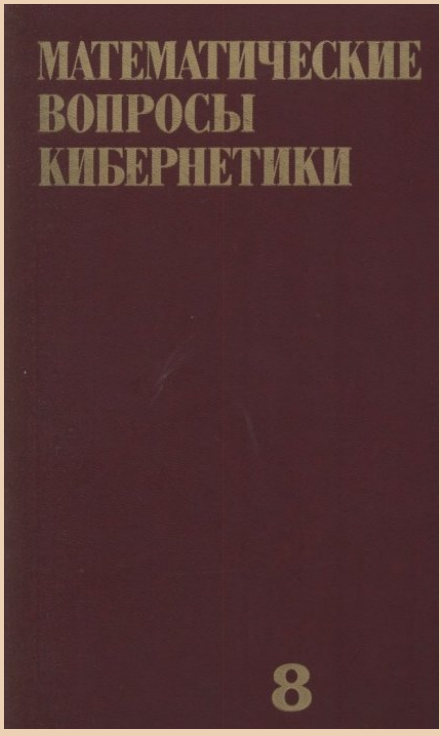
ИПМ им. М.В. Келдыша РАН • Электронная библиотека
Математические вопросы кибернетики • Выпуск 8



Ю. В. Рогожин

Универсальные вычисления

Рекомендуемая форма библиографической ссылки: Рогожин Ю. В. Универсальные вычисления // Математические вопросы кибернетики. Вып. 8. — М.: Наука, 1999. — С. 147–190. URL: http://library.keldysh.ru/mvk.asp?id=1999-147

УНИВЕРСАЛЬНЫЕ ВЫЧИСЛЕНИЯ

Ю. В. РОГОЖИН
(КИШИНЕВ)

§ 1. Введение

В настоящем обзоре рассмотрены классическая Тьюринговская модель вычислений и современные модели биомолекулярных вычислений, основанные на операции рекомбинации молекул (*splicing*-операции).

Создание и изучение различных формальных моделей компьютеров и вычислительных алгоритмов очень важно для понимания истинной природы универсальных вычислений. Из этого анализа могут возникнуть новые подходы для построения будущих компьютеров (квантовых, генетических, молекулярных или каких-либо других). Исследование ограничений этих формальных моделей позволяет глубже понять возможности современных и будущих компьютеров.

Одну из первых теоретических моделей компьютера в 1936 г. предложил А. Тьюринг [48], один из основателей информатики и вычислительной техники. Она называется, в его честь, (одноленточной) машиной Тьюринга (МТ) и является базовой для других моделей компьютеров и вычислений. Основная причина этого заключается в ее простоте, элегантности и гибкости и в том факте, что шаг вычисления машиной Тьюринга элементарен как с операционной, так и с коммуникационной точек зрения.

Одним из основных результатов информатики и вычислительной техники является возможность построения конкретной машины Тьюринга, которая при подходящем кодировании может выполнять работу любой другой машины Тьюринга. Универсальная машина, построенная Тьюрингом, содержала несколько сот команд. Поиски универсальной машины Тьюринга минимального размера начались в 1956 г. с работы К. Шеннона [47]. С тех пор многие ученые пытались строить минимальные универсальные машины Тьюринга. Основными мотивами здесь были теоретический и практический научный интерес, поиск принципов, обуславливающих мощь генетических информационных процессов, и стремление уменьшать размер и повышать производительность компьютеров.

Мы будем рассматривать обычные одноголовочные машины Тьюринга с одной одномерной бесконечной в обе стороны лентой.

Как уже говорилось, К. Шеннон в 1956 г. сформулировал проблему построения минимальной универсальной машины Тьюринга и в качестве меры сложности предложил использовать произведение mn числа состояний (m) и числа символов (n) машины Тьюринга, т. е. максимально возможное количество команд в программе машины Тьюринга. В качестве меры сложности можно рассматривать также число команд, реально используемых в программе машины Тьюринга. Одной из целей, преследуемых в данной работе,

и является построение малых универсальных машин Тьюринга, позволяющих существенно приблизить решение задачи Шеннона нахождения минимальной по объему универсальной машины Тьюринга.

В литературе имеется несколько различных вариантов определения универсальной машины Тьюринга (УМТ) (см. [13, 1, 14, 33, 34, 43] и др.). В § 3 охарактеризованы два существенно различных подхода к определению понятия универсальной машины Тьюринга, которые можно назвать *функциональным* (универсальная машина — это такая машина, которая способна вычислять любую частично рекурсивную функцию) и *имитационным* (универсальная машина — это такая машина, которая способна моделировать работу любой машины Тьюринга, нормального алгорифма Маркова, таг-системы или других подобных алгоритмических систем). Отметим, что мы не рассматриваем «нетрадиционные» варианты определения УМТ, например, с «динамическим остановом» (т. е. «универсальные» машины, которые не останавливаются в процессе работы, а результат считывается с ленты в случае выполнения некоего условия для последовательных конфигураций машины), а также определения, допускающие «универсальную» машину, результатом работы которой может быть конечное множество значений. В теореме 1 устанавливается эквивалентность соответствующих этим подходам определений универсальной машины Тьюринга. Отметим, что доказательство этого результата достаточно просто, а сам результат характеризует позицию, занимаемую автором в вопросе определения понятия УМТ.

В § 4 изучается возможность существования универсальных машин Тьюринга в зависимости от ограничений, накладываемых на операционные возможности и на мощности алфавита символов и алфавита состояний машин Тьюринга. Соответствующую иерархию подклассов машин Тьюринга впервые предложил П. Фишер [18]; ее изучение продолжили С. Аандераа и П. Фишер [8]. Автором эта иерархия была дополнена и изучена (см. [5]); полученные в § 4 результаты дают для этого круга вопросов в некотором смысле окончательную картину.

Машина Тьюринга может за один такт выполнить три операции:

a — записать символ в рассматриваемую ячейку (может быть, тот же самый, что был там ранее),

δ — сдвинуть головку на одну ячейку ленты вправо или влево, или оставить ее на месте,

q — перейти в новое состояние (которое, вообще говоря, может совпадать со старым).

Пусть $X, Y \in \{a, \delta, q\}$, запись XY обозначает, что за один такт могут выполняться операции X и Y, запись $X \vee Y$ обозначает, что за один такт может выполняться либо операция X, либо Y (но не обе одновременно). Тогда, например, условие $qa \vee a\delta$ обозначает, что за один такт машина может либо записать символ и перейти в новое состояние, либо записать символ и сдвинуть головку. Используя эти обозначения, определим 9 классов машин Тьюринга (табл. 1).

Указанными классами исчерпываются все логические возможности (мы рассматриваем только классы машин Тьюринга, которые могут выполнять все три операции: a, δ и q). Классы SM, DM и UM ввел П. Фишер, классы AM, BM, CM и FM введены автором. Соотношения между этими класса-

Таблица 1

Класс	Условие	Класс	Условие	Класс	Условие
TM	$qa\delta$	DM	$q\delta \vee a\delta$	BM	$qa \vee \delta$
FM	$qa \vee q\delta \vee a\delta$	PM	$qa \vee q\delta$	CM	$q\delta \vee a$
SM	$qa \vee a\delta$	AM	$q \vee a\delta$	UM	$q \vee a \vee \delta$

ми показаны на рис. 1. Обозначим иерархию этих подклассов машин Тьюринга через $\mathcal{FTM}$ (*formalisms for Turing machines* — формализмы для машин Тьюринга).

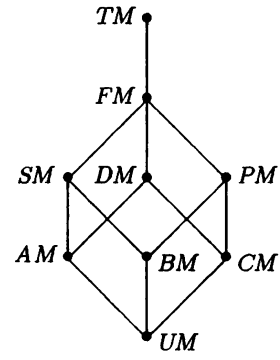


Рис. 1. Иерархия $\mathcal{FTM}$ подклассов машин Тьюринга

Класс TM — это класс обычных машин Тьюринга. Класс PM (называемый также классом машин Поста) — это подкласс машин Тьюринга, которые за один такт работы не могут одновременно выполнить операции a и δ, т. е. записать символ и сдвинуть головку. Машины Поста, наряду с машинами Тьюринга, широко используются в литературе по теории алгоритмов и информатике. Класс DM — это подкласс машин Тьюринга, которые не могут одновременно выполнить операции q и a. Класс SM — это подкласс машин Тьюринга, которые не могут одновременно выполнить операции q и δ. Класс UM — это подкласс машин Тьюринга, которые за один такт работы выполняют только одну из трех операций: a, q или δ. Наконец, FM, AM, BM и CM — это промежуточные классы, которые позволили автору дополнить и завершить иерархию $\mathcal{FTM}$.

Как показал К. Шеннон, универсальные машины существуют как в классе TM с 2 состояниями, так и в классе TM с 2 символами (известно также, что УМТ с одним состоянием невозможны, как невозможны УМТ с одним символом). Будем обозначать это утверждение через $TM(2, 2)$; здесь первый аргумент — минимальное число состояний универсальной машины в классе TM, а второй — минимальное число символов универсальной машины в этом же классе.

С. Аандераа, П. Фишер и П. Хупер доказали утверждения $PM(3, 2)$, $DM(2, 2)$ и $SM(2, 3)$. Здесь представляется важным тот факт, что запрет на одновременное выполнение операций a и δ влечет невозможность построения УМТ с 2 состояниями, а запрет на одновременное выполнение операций q и δ — невозможность построения УМТ с 2 символами.

Автору удалось показать, что в классе UM универсальную машину можно построить как с 3 состояниями, так и с 3 символами, т. е. доказать утверждение $UM(3, 3)$ (теорема 2). Все известные и полученные автором результаты суммированы на рис. 2.

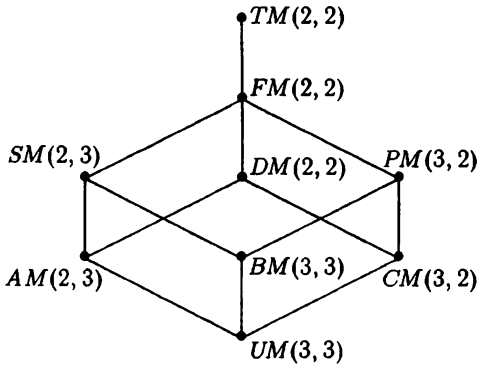


Рис. 2. Универсальность подклассов машин Тьюринга в иерархии $\mathcal{FTM}$

Далее мы рассматриваем вопрос о существовании универсальных машин в классе TM в зависимости от числа символов и состояний. Класс универсальных машин Тьюринга с m состояниями и n символами обозначим $\mathcal{UTM}(m, n)$.

В § 5 описываются принципы построения малых универсальных машин Тьюринга, моделирующих так называемые таг-системы. Универсальные машины Тьюринга можно строить по-разному, в частности, напрямую моделируя другие машины Тьюринга. Однако последний способ требует сложного кодирования и декодирования программы моделируемой машины и, соответственно, объем программы универсальной машины, полученной таким образом, будет значителен (см., например, [49]). М. Минский [33] предложил в качестве очень простого алгоритмического формализма использовать таг-системы и построил универсальную машину Тьюринга с 7 состояниями и 4 символами, которая моделировала таг-системы с $m=2$. Универсальная машина Минского в клас-

се $\mathcal{UTM}(7, 4)$ имеет неразрешимую проблему остановки, но имеет также серьезный недостаток, который был отмечен автором в 1982 г.: эта машина перед остановкой повреждает результат на ленте, и его восстановить нельзя [6]. Р. Робинсон в 1991 г. также отметил этот дефект универсальной машины Минского [42] и предложил свой правильный вариант универсальной машины в классе $\mathcal{UTM}(7, 4)$.

Также в § 5 представлен усовершенствованный автором метод Минского построения универсальных машин, свободный от вышеупомянутого недостатка. С помощью этого усовершенствованного метода и удалось построить наименьшие по объему из известных универсальные машины Тьюринга.

Отметим, что поиски универсальных алгоритмов, минимальных по объему (т. е. по количеству команд в программе), ведутся и в других алгоритмических формализмах. Например, П. Хупер построил универсальные машины Тьюринга с 2 состояниями, 3 символами и 2 лентами, а также с 1 состоянием, 2 символами и 4 лентами [22]. Л. Призе рассматривал задачу Шеннона для двумерных машин Тьюринга [40], а М. Марженстерн — для так называемых нестирающих машин Тьюринга [27]. Результаты по малым регистровым машинам представлены И. Корец [25]. Хорошие обзоры по данной проблематике опубликовал М. Марженстерн [28, 29]. Обзор по универсальным полиномам представлен Ю. В. Матиясевичем [32]. Построение минимальной универсальной машины Тьюринга также рассматривалось в книгах Й. Грушки [19] и Г. Пэуна, Г. Розенберга, А. Саломаа [38].

В § 6 строятся семь универсальных машины Тьюринга, моделирующих таг-системы, и тем самым доказывается теорема 3, согласно которой универсальные машины Тьюринга существуют в следующих классах: $\mathcal{UTM}(22, 2)$, $\mathcal{UTM}(10, 3)$, $\mathcal{UTM}(7, 4)$, $\mathcal{UTM}(5, 5)$, $\mathcal{UTM}(4, 6)$, $\mathcal{UTM}(3, 10)$, $\mathcal{UTM}(2, 18)$.

Как уже говорилось, М. Минский построил универсальную машину Тьюринга в классе $\mathcal{UTM}(7, 4)$ с помощью моделирования таг-систем. Мы также используем этот метод, но представленные в настоящей работе универсальные машины в классах $\mathcal{UTM}(5, 5)$ и $\mathcal{UTM}(4, 6)$ будут моделировать некоторые специальные подклассы таг-систем.

Р. Робинсон рассматривал также число команд, реально используемых в программе машины Тьюринга [42]. Его универсальная машина из класса $\mathcal{UTM}(7, 4)$ использует 27 команд, в то время как аналогичная машина автора использует 26 команд [44]. Отметим также, что представленная автором универсальная машина из класса $\mathcal{UTM}(5, 5)$ использует 23 команды, а из класса $\mathcal{UTM}(4, 6)$ — 22 команды, что является наименьшим (на настоящее время) числом команд универсальной машины Тьюринга.

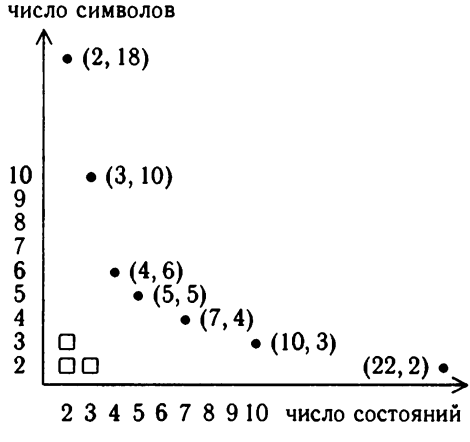


Рис. 3. Таблица малых универсальных машин Тьюринга. Обозначения: • — УМТ существует; □ — УМТ не существует

Как показала Л. М. Павлоцкая, классы $\mathcal{UTM}(3, 2)$ и $\mathcal{UTM}(2, 3)$ универсальных машин пусты [3, 4]. Используя другие методы, В. Дейкерт и М. Кудлек [16, 26] установили аналогичный результат для класса $\mathcal{UTM}(2, 2)$.

Из теоремы 3 и результатов Л. М. Павлоцкой следует, что пока не исследованными (пусты они или нет) остаются 49 классов $\mathcal{UTM}(m, n)$, см. рис. 3.

В §§ 7–12 рассматриваются математические модели молекулярных вычислений (*DNA-computing*), основанные на операции рекомбинации молекул (*splicing*-операции).

Молекулярные вычисления (называемые также биомолекулярными вычислениями, биовычислениями или ДНК-вычислениями) привлекают в настоящее время большое внимание многих специалистов в области химии, биологии и информатики. Они являются новым подходом к вычислениям и используют биомолекулярные операции для решения вычислительных проблем. Первые успешные эксперименты в этом направлении были проведены Л. Адлеманом [9, 10] в решении задачи нахождения гамильтонова пути в графе. Основная идея заключалась в кодировании данных последовательностями ДНК и применении к ним технологий молекулярной биологии для выполнения вычислительных операций. Как показали первые эксперименты и теоретические исследования, молекулярные вычисления обладают большим потенциалом, чтобы превзойти вычисления на обычных электронных компьютерах. Ожидается, что для определенного класса задач (которые хорошо распараллеливаются) ДНК-компьютеры будут иметь огромное преимущество перед ныне существующими по скорости, энергетической эффективности и экономному хранению информации [23, 24].

Однако, несмотря на достигнутый прогресс, существуют серьезные препятствия различного рода (технологические, теоретические) на пути создания реального биокомпьютера. Исследования в этом направлении только начаты, а потому проводятся в двух направлениях:

изучение различных технологий молекулярной биологии для вычислительных целей,

поиск удобных формальных моделей для биовычислений.

В последнем направлении разрабатываются формальные модели и дается их математическое обоснование (например, способность осуществлять универсальные вычисления или порождать любой рекурсивно перечислимый язык), а также разрабатываются молекулярные алгоритмы решения известных трудных задач (ограниченной проблемы соответствия Поста, проблемы выполнимости для булевских формул и т. п.).

Т. Хед в статье [20] описал связи между молекулярными вычислениями и теорией формальных языков. В своей модели (называемой H-системами или системами Хеда) он рассматривал молекулы как слова над некоторым конечным алфавитом, а энзимы*) — как правила рекомбинации молекул. Правило рекомбинации может применяться к двум молекулам: они разрываются в некоторых фиксированных местах, определяемых правилом рекомбинации, и рекомбинируются, т. е. начальный отрезок одной молекулы соединяется с конечным отрезком другой. Идеи Т. Хеда привлекли внимание многих ученых, занимающихся теорией формальных языков. Так, например, Г. Пэун, Г. Розенберг, А. Саломаа [37] интересовались тем, какие классы формальных языков порождаются молекулярными компьютерами, если первоначальные молекулы и энзимы выбраны из заданных классов формальных языков. Э. Чухай-Варью, Л. Кари, Г. Пэун [12] модифицировали концепцию Т. Хеда и предложили систему n пробирок (*test-tube*) — n-tt-систему, другое название — *communicating distributed H-system*. Здесь любая пробирка есть система Хеда (H-система) с дополнительным фильтром-алфавитом. За один макрошаг работы каждая пробирка порождает новые молекулы согласно своим стартовым молекулам и своему множеству правил рекомбинации. После этого результаты каждой пробирки проходят через фильтры остальных пробирок — молекулы, целиком состоящие из букв алфавита-фильтра i-й пробирки, $1 \leqslant i \leqslant n$, удаляются из исходных пробирок

*) Ферменты, расщепляющие белковые молекулы. — *Ред.*

и переходят в i-ю, образуя стартовый набор молекул для следующего макрошага этой пробирки.

Процесс фильтрации результатов одной пробирки в другую увеличивает вычислительные возможности молекулярного компьютера. Назовем n-tt-систему *конечной*, если первоначально каждая пробирка содержит (бесконечно много) копий молекул из конечного множества молекул и обладает конечным набором правил рекомбинации. Известно, что конечные 1-tt-системы порождают только регулярные множества молекул [39].

Тем не менее конечная 2-tt-система может порождать не только регулярные множества молекул. К. Ферретти, Ж. Маури и К. Зандрон показали [17], что любое рекурсивно перечислимое множество молекул порождается соответствующей конечной 9-tt-системой (или конечной 6-tt-системой, если допустить простое кодирование порожденных молекул).

Напомним, что проблема вхождения неразрешима для рекурсивно перечислимых языков. Это означает, что не существует алгоритма $\mathcal{A}$, который по любому данному слову w и данному языку L определяет, принадлежит ли слово w языку L. Более того, найдется такой язык U, что не существует алгоритма $\mathcal{A}$, распознающего принадлежность произвольного слова w языку U. Тривиальное следствие из этого результата следующее: не существует алгоритма, вычисляющего, какие молекулы порождает конечная 6-tt-система. Иначе говоря, результаты работы конечной 6-tt-системы не могут быть алгоритмически предсказаны (определены).

В § 7 приведен пример моделирования H-системой праволинейной грамматики. В § 8 показано, что результаты конечной 3-tt-системы не могут быть алгоритмически предсказаны (теорема 4). В § 9 показано, что конечная расширенная 3-tt-система может порождать любой «чистый» рекурсивно перечислимый язык (теорема 5). Отметим, что позже Г. Пэун представил другое доказательство этого результата [36]. Его работа содержит также хороший обзор по данной проблематике. В § 10 построена универсальная расширенная 3-tt-система (теорема 6). Вопросы «существует ли конечная универсальная 2-tt-система?» и «любой ли рекурсивно перечислимый язык порождают конечные 2-tt-системы?» остаются открытыми; однако известно, что конечные 1-tt-системы порождают только регулярные языки.

Г. Пэун [35] предложил другую модификацию концепции H-систем — так называемые TVDH-системы (*time-varying distributed H-systems*) степени n, $n \geqslant 1$.

TVDH-система степени n, $n \geqslant 1$, есть H-система, которая имеет специальные свойства: в различные моменты времени она использует различные множества правил рекомбинации (эти множества правил называются компонентами TVDH-системы, а количество компонент — степенью TVDH-системы), переход от одного множества правил к другому цикличен. Как показал Г. Пэун [36], TVDH-системы степени 7 и выше порождают любой рекурсивно перечислимый язык (другими словами, вычислительная мощь таких TVDH-систем совпадает с вычислительной мощью машин Тьюринга). Этот результат нами был улучшен: в § 11 показано (теорема 7), как с помощью TVDH-систем степени не менее 2 порождать любой рекурсивно перечислимый язык, а в § 12 построены универсальные TVDH-системы степени не ниже 2 (теорема 8).

Вопросы «можно ли построить универсальную TVDH-систему степени 1?» и «любой ли рекурсивно перечислимый язык порождают TVDH-системы степени 1?» остаются открытыми.

Результаты §§ 7–10 получены совместно с Л. Призе (Кобленц, Германия) и М. Марженстерном (Мец, Франция). Результаты §§ 11, 12 получены совместно с М. Марженстерном.

§ 2. Основные используемые понятия и обозначения

Укажем основные используемые понятия и обозначения. Мы рассматриваем обычные *детерминированные машины Тьюринга* (МТ) с одной одномерной лентой*) и одной головкой, а также используем понятие *конфигурации* МТ (см., например, [15] или [1]). *Программа* машины Тьюринга состоит из конечного числа *команд* (*инструкций*) вида**) $\langle q_i, a_j, a_l, \delta, q_k\rangle$, где $i, k \in \{1, \ldots, n\}$; $j, l \in \{1, \ldots, m\}$; $\delta \in \{R, L, N\}$.

Множество $\{q_i\}$, $i = 1, \ldots, n$, называется *алфавитом состояний*, а множество $\{a_j\}$, $j = 1, \ldots, m$, — *алфавитом символов* МТ.

Как обычно, требуется, чтобы для каждой пары $\langle q_i, a_j\rangle$ программа МТ имела не более одной команды, начинающейся с этой пары.

Выполнение команды машиной Тьюринга интерпретируется следующим образом: если МТ находится в состоянии q_i и ее головка расположена на ленте над ячейкой, содержащей символ a_j, то за один такт работы МТ заменяет символ в этой ячейке на a_l, сдвигает головку влево или вправо или же оставляет на месте в зависимости значения δ (L, R или N) и переходит в состояние q_k. После этого МТ готова выполнить следующую соответствующую команду. Если в процессе работы встретится ситуация, когда для пары $\langle q_i, a_j\rangle$ (где q_i — текущее состояние МТ, а a_j — рассматриваемый символ) в программе нет соответствующей команды (или есть специальная команда остановки вида $\langle q_i, a_j, \mathrm{STOP}\rangle$), то МТ прекращает вычисления (останавливается).

Конфигурацией машины Тьюринга называется совокупность, образованная последовательностью символов, содержащихся в ячейках ленты МТ, состоянием, в котором находится МТ, и номером (или положением на ленте) ячейки, над которой находится головка.

Конфигурацию машины Тьюринга будем иногда обозначать Aq_ia_jB или A, B (возможно, с дополнительными индексами), где q_i — текущее состояние МТ, a_j — рассматриваемый символ, A — содержимое ячеек ленты до рассматриваемой ячейки (не включая ее), B — содержимое ячеек ленты после рассматриваемой ячейки; тем самым A и B — бесконечные последовательности символов (если предполагать, как это обычно принято, что первоначально лишь конечное число ячеек не пусты, т. е. содержат символы, отличные от специального символа «пробел», то эти последовательности можно считать словами в алфавите символов, не включая в них бесконечные «хвосты» из пустых ячеек).

Через $\beta\downarrow$ будем обозначать тот факт, что β есть заключительная (финальная) конфигурация. Пусть $\beta_{i_1} \overset{M}{\to} \beta_{i_2}$ обозначает, что машина Тьюринга M переходит от конфигурации β_{i_1} к β_{i_2} за один шаг. Запись $\beta_{i_1} \overset{M}{\Rightarrow} \beta_{i_j}$ обозначает возможность перехода от конфигурации β_{i_1} к β_{i_j} за несколько шагов $(\beta_{i_1} \overset{M}{\to} \beta_{i_2} \overset{M}{\to} \ldots \overset{M}{\to} \beta_{i_j})$.

Обозначим через $\mathfrak{B}$ множество всех конфигураций всех машин Тьюринга, а через $\mathfrak{B}_M$ — множество конфигураций фиксированной машины M. Хорошо известно, что множества $\mathfrak{B}$ и $\mathfrak{B}_M$ являются рекурсивно перечислимыми. Определим функцию F_M: $\mathfrak{B}_M \to \mathfrak{B}_M$ следующим образом:

$$F_M(\alpha) = \beta \iff \alpha \overset{M}{\Rightarrow} \beta, \qquad \alpha, \beta \in \mathfrak{B}_M, \quad \beta\downarrow.$$

---

*) Предполагается, что лента бесконечна в обе стороны и ее ячейки занумерованы целыми числами. — *Ред.*

**) Иногда мы будем записывать команды проще, в виде $q_ia_ja_l\delta q_k$, т. е. опуская скобки и запятые.

Область определения и множество значений функции f обозначим через $\mathrm{Def}(f)$ и $\mathrm{Val}(f)$.

С каждой машиной Тьюринга можно стандартным образом связать вычисляемую ею функцию. Обычно предполагают, что алфавит символов содержит символы «пробел» и «1», целое неотрицательное число n кодируется последовательностью из $n+1$ ячеек, содержащих символ «1», информация на ленте машины Тьюринга записывается подряд, аргументы отделены друг от друга пустыми ячейками, все ячейки после последнего аргумента также пустые, машина начинает работу в состоянии q_1, а ее головка расположена над первой непустой ячейкой. Когда машина Тьюринга останавливается, ее головка опять должна оказаться над первой непустой ячейкой, а результатом вычисления считается информация на ленте до ближайшей (вперед) пустой ячейки. Известно, что класс функций, вычислимых машинами Тьюринга, совпадает с классом частично рекурсивных функций.

Напомним определение частично рекурсивных функций. Пусть $\mathbb{Z}_+$ — множество целых неотрицательных чисел. Отображения вида $f\colon M \to \mathbb{Z}_+$, где $M \subseteq \mathbb{Z}_+^n$, называем *частичными* функциями. При $M = \mathbb{Z}_+^n$ функцию называют *всюду определенной*. Введем базовые функции (здесь $x_0, x_1, \ldots \in \mathbb{Z}_+$):

$$Z(x_0) = 0,$$
$$S(x_0) = x_0 + 1,$$
$$U_i^{n+1}(x_0, x_1, \ldots, x_n) = x_i, \qquad n \in \mathbb{Z}_+, \quad 0 \leqslant i \leqslant n.$$

Класс частично рекурсивных функций можно определить как наименьший класс функций, содержащий базовые функции и замкнутый относительно следующих трех операций.

1. *Рекурсивная схема* (*рекурсия*) определяет по n-местной функции g и $(n+2)$-местной функции h новую $(n+1)$-местную функцию f:

$$f(0, x_1, \ldots, x_n) = g(x_1, \ldots, x_n),$$
$$f(i+1, x_1, \ldots, x_n) = h(f(i, x_1, \ldots, x_n), i, x_1, \ldots, x_n), \qquad i > 0.$$

2. Операция *композиции* функций h и $g_0, \ldots, g_k$ определяет функцию f следующим образом: $f(x_0, x_1, \ldots, x_n) = h(g_0(x_0, \ldots, x_n), \ldots, g_k(x_0, \ldots, x_n))$.

3. Операция *минимизации* строит по $(n+2)$-местной функции g новую $(n+1)$-местную функцию f следующим образом [1]: значение $f(x_0, \ldots, x_n)$ определено и равно целому неотрицательному числу y в том и только том случае, когда $g(y, x_0, \ldots, x_n) = 0$, причем при всех z, для которых $0 \leqslant z < y$, значения $g(z, x_0, \ldots, x_n)$ определены и отличны от нуля; в противном случае функцию f считаем не определенной на наборе $(x_0, \ldots, x_n)$.

Функция f называется *частично рекурсивной*, если она является базовой функцией либо за конечное число шагов получается из базовых применением операций композиции, рекурсивной схемы и минимизации.

Всюду определенная частично рекурсивная функция называется *общерекурсивной* функцией.

В информатике принят *тезис Тьюринга — Чёрча*, который говорит о том, что любой «интуитивный алгоритм» можно представить подходящей машиной Тьюринга, а любая «интуитивно вычислимая функция» есть частично рекурсивная функция.

Рассмотрим произвольную систему $\mathcal{G}$ числовых частичных одноместных функций. Частичная функция $\psi(\cdot, \cdot)$ называется *универсальной* для семейства $\mathcal{G}$, если выполнены следующие два условия:

для каждого фиксированного числа i одноместная функция $\psi(i, \cdot)$ принадлежит $\mathcal{G}$;

для каждой функции f из $\mathcal{G}$ существует такое число i, что для всех x выполняется равенство $\psi(i, x) = f(x)$.

Иначе говоря, функция $\psi(\cdot, \cdot)$ универсальна для семейства $\mathcal{G}$, если последовательность $\psi(0, \cdot), \psi(1, \cdot), \psi(2, \cdot), \ldots$ содержит (возможно, с повторениями) все функции этого семейства.

Наряду с числовыми частично рекурсивными функциями мы будем рассматривать словарные частично рекурсивные функции [1]. Таковыми являются введенная выше функция $F_M(\alpha)$ и определяемые ниже кодирующие и декодирующие функции, причем кодирующие и декодирующие функции являются также и рекурсивными словарными функциями.

Одним из главных результатов информатики является существование универсальной машины Тьюринга, т. е. конкретной машины Тьюринга U, которая по кодам любой другой произвольной машины Тьюринга P и входных данных для нее a выдает тот же, что и машина P, результат (или код этого результата), см. [48].

Этот результат можно перефразировать для частично рекурсивных функций. Например, существует частично рекурсивная функция двух переменных $u(\cdot, \cdot)$, универсальная для семейства всех одноместных частично рекурсивных функций, т. е. существует последовательность $u(0, \cdot)$, $u(1, \cdot), \ldots, u(i, \cdot), \ldots$, содержащая (возможно, с повторениями) все одноместные частично рекурсивные функции.

Тезис Тьюринга — Чёрча можно перефразировать и так: не существует универсального компьютера, превосходящего по вычислительным возможностям универсальную машину Тьюринга.

Зафиксируем некоторое стандартное эффективное перечисление (нумерацию) всевозможных наборов команд (программ) машин Тьюринга, т. е. каждому целому неотрицательному числу x взаимно однозначно поставим в соответствие набор команд, стоящий на $(x+1)$-м месте в этом перечислении. Процесс перечисления понимаем в двух смыслах: как алгоритм для перехода от произвольного числа x к соответствующему набору команд T_x (машине Тьюринга T_x) и как алгоритм перехода от набора команд машины Тьюринга T к такому числу x, что $T = T_x$. Число x называется *индексом* или *гёделевым номером* набора команд T_x (машины Тьюринга T_x), а соответствующее перечисление — *гёделевой нумерацией*. Хорошо известно, что результаты теории рекурсивных функций не зависят от выбранной гёделевой нумерации.

Множество $\mathcal{A}$, $\mathcal{A} \subseteq \mathbb{Z}_+$, называется *рекурсивно перечислимым*, если оно является множеством значений соответствующей частично рекурсивной функции. Множество $\mathcal{A}$ называется *рекурсивным*, если существует алгоритм, который по любому числу n может определить, принадлежит ли оно этому множеству (т. е. решает проблему вхождения для множества $\mathcal{A}$).

Числовое множество $\mathcal{A}$ называется *m-сводимым* к числовому множеству $\mathcal{B}$, если существует такая общерекурсивная функция $f(x)$, что

$$x \in \mathcal{A} \iff f(x) \in \mathcal{B}$$

для всех значений x. Множество $\mathcal{U}$ называется *m-универсальным*, если выполнены следующие два условия: 1) $\mathcal{U}$ рекурсивно перечислимо; 2) каждое рекурсивно перечислимое множество m-сводится к $\mathcal{U}$.

Мы будем использовать следующие стандартные обозначения из формальной теории языков. *Алфавит* есть конечное непустое множество, элементы которого называются *буквами*. *Слово* в некотором алфавите Σ есть конечная (возможно, пустая, т. е. длины 0) последовательность букв из Σ. Слово длины 0 называется *пустым словом* и обозначается ε. Множество всех слов в алфавите Σ обозначается через Σ^*. *Языком* (в алфавите Σ) называется произвольное подмножество множества Σ^*. Пусть a и b — слова в алфавите Σ, имеющие длины n и m, т. е. $a = a_1 \ldots a_n$, $b = b_1 \ldots b_m$. Запись $a \cdot b$ или ab обозначает *конкатенацию* этих слов, т. е. слово

$a_1 \dots a_n b_1 \dots b_m$ длины $n+m$. (Легко видеть, что для любых слов a, b, c выполнены соотношения $a\varepsilon = \varepsilon a = a$ и $a(bc) = (ab)c$.)

Формальная грамматика G есть четверка $G = (N, T, R, S)$, состоящая из алфавита N *нетерминальных* букв, алфавита T *терминальных* букв, где $N \cap T = \varnothing$, *начальной буквы* S, $S \in N$, и конечного множества R *правил* вида $u \to v$, где $u, v \in (N \cup T)^*$.

Наличие правила $u \to v$ понимается как возможность строить по слову w, $w \in (N \cup T)^*$, новые слова, заменяя любое вхождение слова u на слово v. На множестве слов в алфавите $N \cup T$ введем бинарное отношение $\Rightarrow_G$, полагая утверждение $w \Rightarrow_G w'$ истинным в том и только том случае, когда существует такое правило $u \to v$ из R и такие слова w_1, w_2 в алфавите $N \cup T$, что $w = w_1 u w_2$ и $w' = w_1 v w_2$. Бинарное отношение $\Rightarrow^*_G$ определим как рефлексивное и транзитивное замыкание отношения $\Rightarrow_G$, т. е. утверждение $w \Rightarrow^*_G w'$ обозначает наличие такого числа n и таких слов $w_1, \dots, w_n$, что $w = w_1$, $w' = w_n$ и $w_i \Rightarrow_G w_{i+1}$, $1 \leqslant i < n$. Допустим также случай $n = 1$, так что утверждение $w \Rightarrow^*_G w$ всегда верно. Если грамматика G однозначно понятна из контекста или ее выбор несущественен, мы часто будем опускать индекс G и вместо $\Rightarrow_G$ писать просто $\Rightarrow$.

Последовательность $w_1 \Rightarrow w_2 \Rightarrow \dots \Rightarrow w_n$ также называется *вычислением* (говорим, что слово w_n есть результат вычисления длины $n-1$ из слова w_1). *Терминальное слово* есть слово в алфавите T. Все терминальные слова, вычислимые из начальной буквы*) S, образуют язык $L(G)$, порожденный грамматикой G. Более формально, $L(G) := \{w \in T^*;\ S \Rightarrow^* w\}$. Язык называется *рекурсивно перечислимым*, если существует машина Тьюринга, порождающая его. Одним из фундаментальных фактов в информатике является следующий: язык L является рекурсивно перечислимым в том и только том случае, когда существует такая формальная грамматика G, что $L = L(G)$.

Под (абстрактной) *молекулой* мы будем понимать слово в некотором алфавите. *Абстрактный энзим* (другие названия — *правило рекомбинации* или *splicing-правило*) есть четверка (u_1, u_2, u'_1, u'_2) слов, которые часто записываются в виде двумерной таблицы $\begin{array}{c|c} u_1 & u_2 \\ \hline u'_1 & u'_2 \end{array}$.

Говорим, что правило рекомбинации $r = (u_1, u_2, u'_1, u'_2)$ применимо к молекулам m_1 и m_2, если $m_1 = w_1 u_1 u_2 w_2$ и $m_2 = w'_1 u'_1 u'_2 w'_2$, где w_1, w_2, w'_1 и w'_2 — некоторые (возможно, пустые) слова в том же алфавите; результатом применения правила рекомбинации являются новые молекулы m'_1 и m'_2, где $m'_1 = w_1 u_1 u'_2 w'_2$ и $m'_2 = w'_1 u'_1 u_2 w_2$. В этом случае мы пишем $\{m_1, m_2\} \vdash_r \{m'_1, m'_2\}$. (Иногда мы будем также использовать сокращенную запись $m_1 \vdash_r m'_1$.)

H-система (система Хеда) есть тройка $H = (\Sigma, M, E)$, состоящая из алфавита Σ, множества M первоначальных молекул в алфавите Σ и множества E правил рекомбинации, $E \subseteq (\Sigma^*)^4$. H-система называется конечной, если конечны множества M и E. Для любого множества молекул L, $L \subseteq \Sigma^*$, определим

$$\sigma_H(L) := \{w \in \Sigma^*:\ (\exists w_1, w_2 \in L)(\exists w' \in \Sigma^*)(\exists r \in E)\ \{w_1, w_2\} \vdash_r \{w, w'\}\},$$

т. е. $\sigma_H(L)$ есть множество всех молекул, порождаемых из L однократным применением правил рекомбинации. Далее,

$$\sigma^0_H(L) := L, \qquad \sigma^{i+1}_H(L) := \sigma^i_H(L) \cup \sigma_H(\sigma^i_H(L)), \qquad \sigma^*_H(L) := \bigcup_{i \geqslant 0} \sigma^i_H(L).$$

*) Точнее, из однобуквенного слова S; здесь и ниже мы не различаем букву и однобуквенное слово, состоящее только из этой буквы.

Результатом H-системы H является множество $\sigma(H)$, определяемое как $\sigma(H) := \sigma^*_H(M)$, т. е. множество всех молекул, получающихся в результате итеративного применения правил рекомбинации, начиная с множества M первоначальных молекул.

Пробирка (*test-tube*) T есть пара $T = (H, F)$, состоящая из H-системы $H = (\Sigma, M, E)$ и алфавита $F \subseteq \Sigma$, называемого *фильтром*.

H-система с n пробирками, или просто *n-tt-система*, есть набор $\mathscr{H} = (\Sigma, T_1, \ldots, T_n)$, состоящий из алфавита Σ и n пробирок $T_i = (H_i, F_i) = ((\Sigma, M_i, E_i), F_i)$, $1 \leqslant i \leqslant n$.

Для языков $L_1, \ldots, L_n$, $L'_1, \ldots, L'_n$ в алфавите Σ мы пишем $(L_1, \ldots, L_n) \vdash_{\mathscr{H}} (L'_1, \ldots, L'_n)$, если верны равенства

$$L'_i = \bigcup_{j=1}^{n} (\sigma^*_{T_j}(L_j) \cap F_i^*) \cup \left(\sigma^*_{T_i}(L_i) \cap \left(\Sigma^* \setminus \bigcup_{j=1}^{n} F_j^*\right)\right), \qquad i = 1, \ldots, n$$

(под $\sigma^*_{T_j}(L_j)$ мы понимаем $\sigma^*_{H_j}(L_j)$). Иначе говоря, для получения множества L'_i порождаются результаты $\sigma^*_{T_j}(L_j)$ всех пробирок T_j, считая L_j первоначальным множеством молекул, и в L'_i помещаются все те результаты $\sigma^*_{T_j}(L_j)$, которые проходят через фильтр F_i. Добавим, что в L'_i остаются все те молекулы, произведенные в T_i из L_i (т. е. молекулы множества $\sigma^*_{T_i}(L_i)$), которые не могут пройти никакой фильтр F_j системы Хеда T_j. Обозначим через $\vdash^*_{\mathscr{H}}$ рефлексивное и транзитивное замыкание отношения $\vdash_{\mathscr{H}}$.

Результатом $\rho(\mathscr{H})$ рассматриваемой n-tt-системы $\mathscr{H}$ считаем все то, что находится в первой пробирке:

$$\rho(\mathscr{H}) := \{L_1 \subseteq \Sigma^* : (\exists L_2, \ldots, L_n \subseteq \Sigma^*)\ (M_1, \ldots, M_n) \vdash^*_{\mathscr{H}} (L_1, \ldots, L_n)\},$$

т. е. молекулы, порожденные в первой пробирке T_1, если первоначально пробирки T_i, $1 \leqslant i \leqslant n$, содержали соответственные множества молекул M_i.

Очевидно, что H-система $H = (\Sigma, M, E)$ и 1-tt-система $T = (H, F)$ определяют один и тот же результат.

Расширенная n-tt-система $\mathscr{H} = (\Sigma, T_1, \ldots, T_n, \Gamma)$ есть n-tt-система $\mathscr{H}' = (\Sigma, T_1, \ldots, T_n)$, дополненная *терминальным алфавитом* Γ, $\Gamma \subseteq \Sigma$. Ее *результат* $\rho(\mathscr{H})$ определяется как $\rho(\mathscr{H}) := \rho((\Sigma, T_1, \ldots, T_n)) \cap \Gamma^*$, т. е. множество всех порождаемых в первой пробирке слов-молекул, содержащих только буквы алфавита Γ.

TVDH-система (*time-varying distributed H-system*) степени n, $n \geqslant 1$, есть набор $\Gamma = (V, T, A, R_1, R_2, \ldots, R_n)$, где V — алфавит, T — его подмножество (терминальный алфавит), A — конечное подмножество множества V^* (множество аксиом), $R_1, \ldots, R_n$ — конечные множества правил рекомбинации в алфавите V. Множества R_i называются *компонентами* TVDH-системы, а их количество n — ее *степенью*.

В каждый момент $k = nj + i$, где $j \geqslant 0$, $1 \leqslant i \leqslant n$, компонента R_i использует для операций рекомбинации доступные в настоящий момент молекулы. А именно, мы определяем

$$L_1 = A, \qquad L_{k+1} = \sigma_i(L_k), \quad i \equiv k \pmod{n}, \quad k \geqslant 1,$$

где $\sigma_i(L)$, $1 \leqslant i \leqslant n$, обозначает однократное применение правил рекомбинации R_i к множеству молекул L, причем молекулы из L, к которым не применимо ни одно правило из R_i, удаляются. Таким образом, от k-го шага к $(k+1)$-му передается только результат правил рекомбинации, применяемых к молекулам множества L_k согласно правилам из R_i для $i \equiv k \pmod{n}$; молекулы множества L_k, которые не могут участвовать в операциях рекомбинации, удаляются. Язык, порожденный системой Γ, определяется соотношением $L(\Gamma) = T^* \cap \bigcup_{k \geqslant 1} L_k$.

§ 3. Эквивалентность определений УМТ

Доказательство эквивалентности двух определений УМТ будет проведено менее формально, чем доказательства в других разделах данной работы. Это сделано с целью упрощения изложения результатов, поскольку формальное доказательство можно без труда восстановить.

Определение 1. Пусть $\psi(\cdot, \cdot)$ — некоторая универсальная частично рекурсивная функция для множества всех частично рекурсивных функций одной переменной. Машина Тьюринга U называется *универсальной*, если существуют такие рекурсивные функции $\rho(\cdot, \cdot)$ (*кодирующая функция*), где $\mathrm{Val}(\rho) \subseteq \mathscr{B}_U$, и $\lambda(\cdot)$ (*декодирующая функция*), что для всех n и x выполнено равенство $\lambda(F_U(\rho(n, x))) = \psi(n, x)$.

Замечание 1. Это определение УМТ отличается от предложенного М. Дэвисом [14] лишь несколькими несущественными деталями.

Определение 2. Машина Тьюринга M *моделирует* машину Тьюринга T, если существуют такие рекурсивные функции $\tilde{\rho}(\cdot)$ (*кодирующая функция*), где $\mathrm{Val}(\tilde{\rho}) \subseteq \mathscr{B}_M$, и $\tilde{\lambda}(\cdot)$ (*декодирующая функция*), где $\mathrm{Val}(\tilde{\lambda}) \subseteq \mathscr{B}_T$, что для всех α из $\mathscr{B}_T$ имеет место равенство $\tilde{\lambda}(F_M(\tilde{\rho}(\alpha))) = F_T(\alpha)$.

Замечание 2. Это определение основано на определении моделирования одной абстрактной вычислительной машины другой, предложенном Г. Германом [21].

Пусть мы выбрали некоторое стандартное перечисление (гёделеву нумерацию) программ всех машин Тьюринга. Пусть в этом перечислении T_n есть обозначение для машины Тьюринга с гёделевым номером n.

Определение 3. *Машина Тьюринга U называется универсальной*, если она может моделировать каждую машину Тьюринга T, причем кодирующая и декодирующая функции могут быть эффективно определены по программе моделируемой машины, т. е. существуют такие рекурсивные функции $\tilde{\rho}(\cdot, \cdot)$ (*кодирующая функция*), где $\mathrm{Val}(\tilde{\rho}) \subseteq \mathscr{B}_U$, и $\tilde{\lambda}(\cdot, \cdot)$ (*декодирующая функция*), где $\mathrm{Val}(\tilde{\lambda}) \subseteq \mathscr{B}$, что для всех n из $\mathbb{Z}_+$ и всех α из $\mathscr{B}$ имеет место равенство

$$\tilde{\lambda}(n, F_U(\tilde{\rho}(n, \alpha))) = F_{T_n}(\alpha). \tag{1}$$

Замечание 3. Легко показать, что декодирующая функция $\tilde{\lambda}$ может быть функцией одного аргумента (не зависящей от n), и выражение (1) можно переписать в виде $\tilde{\lambda}(F_U(\tilde{\rho}(n, \alpha))) = F_{T_n}(\alpha)$.

Теорема 1. *Определения* 1 *и* 3 *универсальной машины Тьюринга эквивалентны.*

Доказательство теоремы 1 основано на следующих трех леммах.

Лемма 1. *Пусть M является универсальной машиной Тьюринга в смысле определения* 1. *Тогда она вычисляет любую бинарную частично рекурсивную функцию.*

Доказательство очевидно.

Пусть, как определено выше, T_n — машина Тьюринга с гёделевым номером n. Пусть $G(\cdot)$ — некоторая гёделева нумерация множества $\mathscr{B}$ всевозможных конфигураций машин Тьюринга. С каждой машиной Тьюринга M свяжем числовую функцию φ_M:

$$\varphi_M(x) = y \iff (x \in G^{-1}(\mathscr{B}_M)) \,\&\, (y \in G^{-1}(\mathscr{B}_M)) \,\&\, (F_M(G(x)) = G(y)). \tag{2}$$

Эта функция — частично рекурсивная, поскольку, очевидно, $\mathscr{B}_M$ есть рекурсивное множество.

Пусть значение $F_M(\alpha)$ не определено, если $\alpha \notin \mathscr{B}_M$. Из (2) следует, что

$$(\forall x \in \mathbb{Z}_+)\ \varphi_M(x) = G^{-1} \circ F_M \circ G(x). \tag{3}$$

Лемма 2. *Машина Тьюринга, универсальная в смысле определения* 1, *универсальна также и в смысле определения* 3.

Доказательство. Очевидно, что функция $t(\cdot, \cdot)$, заданная соотношением $t(n, x) = \varphi_{T_n}(x)$, является частично рекурсивной.

Пусть машина Тьюринга M универсальна в смысле определения 1. Тогда согласно лемме 1 существуют такие рекурсивные функции λ и ρ, что для всех n и x имеет место равенство $\lambda \circ F_M \circ \rho(n, x) = \varphi_{T_n}(x)$. Пользуясь соотношением (3), запишем $(\forall n, x \in \mathbb{Z}_+)\ \varphi_{T_n}(x) = G^{-1} \circ F_{T_n} \circ G(x)$. С учетом предыдущего соотношения получаем $\lambda \circ F_M \circ \rho(n, x) = G^{-1} \circ F_{T_n} \circ G(x)$, откуда $G \circ \lambda \circ F_M \circ \rho(n, x) = F_{T_n} \circ G(x)$.

Пусть $x = G^{-1}(\alpha)$, где $\alpha \in \mathscr{B}$. Тогда $(\forall \alpha \in \mathscr{B})\ G \circ \lambda \circ F_M \circ \rho(n, G^{-1}(\alpha)) = F_{T_n}(\alpha)$.

Пусть $\widetilde{\lambda}(\beta) = G \circ \lambda(\beta)$ для всех β из $\mathscr{B}_M$ и $\widetilde{\rho}(n, \alpha) = \rho(n, G^{-1}(\alpha))$ для всех α из $\mathscr{B}$. Тогда $\widetilde{\lambda} \circ F_M \circ \widetilde{\rho}(n, \alpha) = F_{T_n}(\alpha)$ для всех α из $\mathscr{B}$ и всех n из $\mathbb{Z}_+$. Следовательно, M есть универсальная (в смысле определения 3) машина Тьюринга. Лемма 2 доказана.

Лемма 3. *Машина Тьюринга, универсальная в смысле определения* 3, *универсальна также и в смысле определения* 1.

Доказательство. Пусть машина Тьюринга M универсальна в смысле определения 3. Тогда в соответствии с замечанием 3 найдутся такие рекурсивные функции $\widetilde{\lambda}(\cdot)$ и $\widetilde{\rho}(\cdot, \cdot)$, что для всех целых неотрицательных n и всех α из $\mathscr{B}$ имеет место равенство $\widetilde{\lambda} \circ F_M \circ \widetilde{\rho}(n, \alpha) = F_{T_n}(\alpha)$.

Пусть машина Тьюринга T_{n_0} универсальна в смысле определения 1. Тогда, в частности, $\widetilde{\lambda} \circ F_M \circ \widetilde{\rho}(n_0, \alpha) = F_{T_{n_0}}(\alpha)$, и согласно определению 1 существуют такие рекурсивные функции $\lambda(\cdot)$ и $\rho(\cdot, \cdot)$, что для всех n и x выполнено равенство $\lambda \circ F_{T_{n_0}} \circ \rho(n, x) = \psi(n, x)$, где $\psi(\cdot, \cdot)$ есть некоторая универсальная частично рекурсивная функция для класса всех одноместных частично рекурсивных функций. Из двух последних равенств следует, что $\lambda \circ \widetilde{\lambda} \circ F_M \circ \widetilde{\rho}(n_0, \rho(n, x)) = \lambda \circ F_{T_{n_0}} \circ \rho(n, x) = \psi(n, x)$.

Пусть $\lambda_1(\alpha) = \lambda \circ \widetilde{\lambda}(\alpha)$ и $\rho_1(n, x) = \widetilde{\rho}(n_0, \rho(n, x))$. Тогда имеем $\lambda_1 \circ F_M \circ \rho_1(n, x) = \psi(n, x)$, что и требовалось доказать.

Определение 4 (А. И. Мальцев [1]). Машина Тьюринга T называется *универсальной*, если $\mathrm{Def}(F_T)$ есть m-универсальное множество.

Замечание 4. Принадлежащее А. И. Мальцеву определение универсальной машины Тьюринга в действительности шире, чем приведенные выше два определения, так как возможна ситуация, когда машина T имеет m-универсальное множество $\mathrm{Def}(F_T)$, а результатом ее работы является конечное множество значений (в частности, множество значений может состоять из одного элемента). Определение А. И. Мальцева для *строгой* универсальной машины Тьюринга эквивалентно приведенному выше определению 3.

§ 4. Ограничения, накладываемые на машины Тьюринга, и универсальность

В этом параграфе изучается возможность существования универсальных машин Тьюринга в зависимости от ограничений, накладываемых на их операционные возможности и на мощности алфавита символов и алфавита состояний. Полученные автором результаты дают в некотором смысле окончательную картину в этом круге вопросов.

Рассмотрим подклассы машин Тьюринга из иерархии $\mathcal{FTM}$, введенной в § 1. Программы машин из различных классов могут быть нескольких типов. В табл. 2 для каждого из 9 классов иерархии $\mathcal{FTM}$ указаны возможные типы команд. (В этой таблице, как и в § 2, q_i — начальное состояние МТ, a_j — исходный символ, a_l — новый символ, δ — перемещение головки, q_k — новое состояние.)

Утверждение 1. *Для каждой машины A из класса TM с n состояниями и m символами можно построить моделирующую машину B из класса AM, имеющую n состояний и не более $m(n+1)$ символов.*

Доказательство. Пусть $q_1, \ldots, q_n$ — состояния машины A, а $a_1, \ldots, a_m$ — элементы ее алфавита символов. Построим машину B с теми же состояниями, а в качестве ее алфавита символов возьмем множество $\{a_1, \ldots, a_m, a_{11}, \ldots, a_{1m}, a_{21}, \ldots, a_{2m}, \ldots, a_{n1}, \ldots, a_{nm}\}$.

Составим программу для машины B. Если программа машины A содержит команду $\langle q_i, a_j, a_l, \delta, q_k\rangle$, где $i \neq k$, введем в программу машины B три команды: $\langle q_i, a_j, a_{i,j}, N\rangle$, $\langle q_i, a_{i,j}, q_k\rangle$ и $\langle q_k, a_{i,j}, a_l, \delta\rangle$; если же $i = k$ — всего одну команду: $\langle q_i, a_j, a_l, \delta\rangle$.

Опишем кодирующую и декодирующую функции, с помощью которых машина B моделирует машину A.

Каждой конфигурации $\alpha q_i a_j \beta$ машины A соответствует та же конфигурация $\alpha q_i a_j \beta$ машины B. Каждой конфигурации $\alpha q_i a_j \beta$ машины B (где α и β — слова в алфавите символов машины A) соответствует та же конфигурация $\alpha q_i a_j \beta$ машины A.

Теперь нетрудно убедиться, что машина B моделирует машину A. Пусть, например, $\alpha q_i a_j a_t \beta \xrightarrow{A} \alpha a_l q_k a_t \beta$, где $i \neq k$. Тогда согласно построению машины B имеем $\alpha q_i a_j a_t \beta \xrightarrow{B} \alpha q_i a_{i,j} a_t \beta \xrightarrow{B} \alpha q_k a_{i,j} a_t \beta \xrightarrow{B} \alpha a_l q_k a_t \beta$.

Следствие 1. *В классе AM существует универсальная машина с двумя состояниями.*

Доказательство. Это следует из утверждения 1 и существования УМТ с 2 состояниями [47].

Таблица 2

Форматы команд машин Тьюринга из различных классов

Класс МТ	Возможные форматы команд	Класс МТ	Возможные форматы команд	Класс МТ	Возможные форматы команд
TM	$\langle q_i, a_j, a_l, \delta, q_k\rangle$	*DM*	$\langle q_i, a_j, a_l, \delta\rangle$ $\langle q_i, a_j, \delta, q_k\rangle$	*BM*	$\langle q_i, a_j, a_l, q_k\rangle$ $\langle q_i, a_j, \delta\rangle$
FM	$\langle q_i, a_j, a_l, q_k\rangle$ $\langle q_i, a_j, a_l, \delta\rangle$ $\langle q_i, a_j, \delta, q_k\rangle$	*PM*	$\langle q_i, a_j, a_l, q_k\rangle$ $\langle q_i, a_j, \delta, q_k\rangle$	*CM*	$\langle q_i, a_j, \delta, q_k\rangle$ $\langle q_i, a_j, a_l\rangle$
SM	$\langle q_i, a_j, a_l, q_k\rangle$ $\langle q_i, a_j, a_l, \delta\rangle$	*AM*	$\langle q_i, a_j, a_l, \delta\rangle$ $\langle q_i, a_j, q_k\rangle$	*UM*	$\langle q_i, a_j, a_l\rangle$ $\langle q_i, a_j, \delta\rangle$ $\langle q_i, a_j, q_k\rangle$

Утверждение 2. *Для каждой машины A из класса TM с n состояниями и m символами можно построить моделирующую машину B из класса CM, имеющую m символов и не более $n(m+1)$ состояний.*

Доказательство. Пусть $q_1, \ldots, q_n$ — состояния машины A, а $a_1, \ldots, a_m$ — элементы ее алфавита символов. Построим машину B с теми же символами, а в качестве ее алфавита состояний возьмем множество $\{q_1, \ldots, q_n, q_{11}, \ldots, q_{1m}, q_{21}, \ldots, q_{2m}, \ldots, q_{n1}, \ldots, q_{nm}\}$.

Составим программу для машины B. Если программа машины A содержит команду $\langle q_i, a_j, a_l, \delta, q_k\rangle$, где $j \neq l$, введем в программу машины B три команды: $\langle q_i, a_j, N, q_{i,j}\rangle$, $\langle q_{i,j}, a_j, a_l\rangle$ и $\langle q_{i,j}, a_l, \delta, q_k\rangle$; если же $j = l$ — всего одну команду $\langle q_i, a_j, \delta, q_k\rangle$.

Опишем кодирующую и декодирующую функции, с помощью которых машина B моделирует машину A. Каждой конфигурации $\alpha q_i a_j \beta$ машины A соответствует та же конфигурация $\alpha q_i a_j \beta$ машины B (и обратно).

Нетрудно убедиться, что машина B моделирует машину A. Пусть, например, $\alpha q_i a_j a_t \beta \xrightarrow{A} \alpha a_l q_k a_t \beta$. Тогда согласно построению машины B имеем $\alpha q_i a_j a_t \beta \xrightarrow{B} \alpha q_{i,j} a_j a_t \beta \xrightarrow{B} \alpha q_{i,j} a_l a_t \beta \xrightarrow{B} \alpha a_l q_k a_t \beta$.

Следствие 2. *В классе CM существует универсальная машина с двумя символами.*

Доказательство. Это следует из утверждения 2 и существования УМТ с 2 состояниями [47].

Перейдем теперь к исследованию класса *UM*.

Утверждение 3. *Для каждой машины A из класса TM с n состояниями и m символами можно построить моделирующую машину B из класса UM, имеющую 3 состояния и не более $4m(n+1)$ символов.*

Доказательство. Пусть $q_1, \ldots, q_n$ — состояния машины A, а $a_1, \ldots, a_m$ — элементы ее алфавита символов. Состояния машины B обозначим q, q_R и q_L, а ее символы — $a_{w,j,\sigma}$, $b_{i,j,\sigma}$ и $a_{j,\sigma}$, где $w = 1, \ldots, n+1$, $i = 1, \ldots, n$, $j = 1, \ldots, m$ и $\sigma \in \{R, L\}$.

Программа машины B строится следующим образом. Каждой команде $\langle q_i, a_j, a_l, N, q_k\rangle$ машины A в программе машины B соответствуют две команды $\langle q, a_{i,j,\sigma}, a_{k,l,\sigma}\rangle$ для $\sigma = R$ и $\sigma = L$. Команде $\langle q_i, a_j, a_l, \delta, q_k\rangle$, где $\delta \in \{R, L\}$, также соответствует пара команд $\langle q, a_{i,j,\sigma}, b_{n-k+1,l,\delta}\rangle$ для $\sigma = R$ и $\sigma = L$. Остальная часть программы машины B зависит только от n и m:

$\langle q, b_{i,j,\delta}, q_\delta\rangle$,	$i = 1, \ldots, n$;	$\langle q_\delta, a_{n,j,\delta}, a_{n+1,j,\delta}\rangle$;	
$\langle q, a_{n+1,j,\delta}, a_{j,\delta}\rangle$;		$\langle q_\delta, a_{n+1,j,\delta}, q\rangle$;	
$\langle q, a_{j,\delta}, \delta\rangle$;		$\langle q_\delta, b_{i,j,\delta}, a_{i,j,\delta}\rangle$,	$i = 1, \ldots, n$;
$\langle q_\delta, a_{i,j,\delta}, \delta\rangle$,	$i = 1, \ldots, n$;	$\langle q_\delta, b_{i,j,\delta}, q_\delta\rangle$,	$i = 1, \ldots, n$;
$\langle q_\delta, a_{i,j,\delta}, b_{i+1,j,\delta}\rangle$,	$i = 1, \ldots, n-1$;	$\langle q_\delta, a_{j,\sigma}, b_{1,j,\delta}\rangle$,	$\sigma \in \{R, L\}$

(всюду также подразумевается, что $j = 1, \ldots, m$ и $\delta \in \{R, L\}$).

Опишем кодирующую и декодирующую функции, с помощью которых машина B моделирует машину A.

Каждой конфигурации $\alpha q_i a_j \beta$ машины A соответствует конфигурация $\alpha' q a_{i,j,\sigma} \beta'$ машины B, где слова α' и β' получаются из α и β соответственно заменой символов a_r на $a_{r,R}$ (или на $a_{r,L}$, что безразлично), а σ есть R (или L, что также несущественно). Обратно, каждой конфигурации $\alpha q a_{i,j,\sigma} \beta$ машины B, где $\alpha, \beta \in \{a_{r,\gamma},\ r = 1, \ldots, m,\ \gamma \in \{R, L\}\}^*$, соответствует конфигурация $\alpha' q_i a_j \beta'$ машины A, где слова α' и β' получаются из α и β соответственно заменой символов $a_{r,\gamma}$ на a_r.

Поясним идею построения машины B. Всякий раз, когда эта машина находится в конфигурации $\alpha q a_{i,j,\sigma} \beta$, где α и β — слова в алфавите $\{a_{r,\gamma},$

$r = 1, \ldots, m,\ \gamma \in \{R, L\}\}$, это означает, что моделируемая машина A находится в конфигурации $\alpha' q_i a_j \beta'$, где слова α' и β' получаются из α и β соответственно заменой символов $a_{r,\gamma}$ на a_r.

$\alpha' q a_{i,j,\sigma} a_{t,\gamma} \beta'$
$\alpha' q b_{n-k+1,l,R} a_{t,\gamma} \beta'$
$\alpha' q_R b_{n-k+1,l,R} a_{t,\gamma} \beta'$
$\alpha' q_R a_{n-k+1,l,R} a_{t,\gamma} \beta'$
$\alpha' a_{n-k+1,l,R} q_R a_{t,\gamma} \beta'$
$\alpha' a_{n-k+1,l,R} q_R b_{1,t,L} \beta'$
$\alpha' a_{n-k+1,l,R} q_L b_{1,t,L} \beta'$
$\alpha' a_{n-k+1,l,R} q_L a_{1,t,L} \beta'$
$\alpha' q_L a_{n-k+1,l,R} a_{1,t,L} \beta'$
$\alpha' q_L b_{n-k+2,l,R} a_{1,t,L} \beta'$
$\alpha' q_R b_{n-k+2,l,R} a_{1,t,L} \beta'$
$\alpha' q_R a_{n-k+2,l,R} a_{1,t,L} \beta'$
$\alpha' a_{n-k+2,l,R} q_R a_{1,t,L} \beta'$
$\alpha' a_{n-k+2,l,R} q_R b_{2,t,L} \beta'$
$\alpha' a_{n-k+2,l,R} q_L b_{2,t,L} \beta'$
$\alpha' a_{n-k+2,l,R} q_L a_{2,t,L} \beta'$
$\alpha' q_L a_{n-k+2,l,R} a_{2,t,L} \beta'$
. . . . . .
$\alpha' q_L a_{n,l,R} a_{k,t,L} \beta'$
$\alpha' q_L a_{n+1,l,R} a_{k,t,L} \beta'$
$\alpha' q a_{n+1,l,R} a_{k,t,L} \beta'$
$\alpha' q a_{l,R} a_{k,t,L} \beta'$
$\alpha' a_{l,R} q a_{k,t,L} \beta'$

Рис. 4

Информацию о переходе моделируемой машины A в новое состояние q_k, записи нового символа a_l и сдвиге головки машины A в направлении δ моделирующая машина B кодирует записью в рассматриваемую ячейку символа $b_{n-k+1,l,\delta}$. Затем она переходит в состояние q_δ, заменяет символ $b_{n-k+1,l,\delta}$ на $a_{n-k+1,l,\delta}$ и сдвигает головку в направлении δ. Далее происходит «процесс перекачки информации». Всякий раз, когда машина B находится в состоянии q_σ и рассматриваемым символом является $a_{i,j,\bar{\sigma}}$, она заменяет его на $b_{i+1,j,\bar{\sigma}}$, переходит в состояние $q_{\bar{\sigma}}$, затем заменяет $b_{i+1,j,\bar{\sigma}}$ на $a_{i+1,j,\bar{\sigma}}$ и совершает сдвиг $\bar{\sigma}$. «Перекачка информации» закончится, когда B окажется в состоянии q_δ и текущим символом будет $a_{n,l,\delta}$; рано или поздно это случится, поскольку первый индекс растет.

В этот момент в ячейке, соседней (в направлении δ) с текущей, будет находится нужный символ $a_{k,t,\delta}$. Теперь машина B заменяет символ $a_{n,l,\delta}$ на $a_{l,\delta}$, переходит в состояние q и перемещает головку к ячейке, содержащей символ $a_{k,t,\delta}$.

Поясним сказанное примером. Пусть $\alpha q_i a_j a_t \beta \xrightarrow{A} \xrightarrow{A} \alpha a_l q_k a_t \beta$. Тогда $\alpha' q a_{i,j,\sigma} a_{t,\gamma} \beta' \overset{B}{\Rightarrow} \alpha' a_{l,R} q a_{k,t,L} \beta'$, где $\sigma, \gamma \in \{R, L\}$ и слова α и β получаются из α' и β' соответственно заменой символов $a_{r,\gamma}$ на a_r (на рис. 4 работа машины B показана такт за тактом).

Утверждение 4. *Для каждой машины A из класса TM с n состояниями и m символами можно построить моделирующую машину B из класса UM, имеющую $m+1$ символов и не более $7n(m+1)$ состояний.*

Доказательство. Пусть $q_1, \ldots, q_n$ — состояния машины A, а $a_1, \ldots, a_m$ — элементы ее алфавита символов. Пусть машина B имеет все эти же символы и дополнительный символ b, а также все указанные состояния и новые состояния $q^u_{i,j}$, где $i = 1, \ldots, n$, $j = 1, \ldots, m$, $u = 1, \ldots, 7$.

Зададим программу машины B. Каждой команде $\langle q_i, a_j, a_l, N, q_k \rangle$ машины A соответствует в машине B группа из 5 команд: $\langle q_i, a_j, q^1_{i,j} \rangle$, $\langle q^1_{i,j}, a_j, b \rangle$, $\langle q^1_{i,j}, b, q^2_{i,j} \rangle$, $\langle q^2_{i,j}, b, a_l \rangle$ и $\langle q^2_{i,j}, a_l, q_k \rangle$. Команде вида $\langle q_i, a_j, a_l, \delta, q_k \rangle$, где $\delta \in \{R, L\}$, соответствуют три группы команд машины B:

(i)	(ii)	(iii)		
$\langle q_i, a_j, q^1_{i,j} \rangle$;	$\langle q^2_{i,j}, a_l, q^3_{i,j} \rangle$;	$\langle q^2_{i,j}, a_r, q^6_{i,j} \rangle$,	$r = 1, \ldots, m,$	$r \neq l$;
$\langle q^1_{i,j}, a_j, b \rangle$;	$\langle q^3_{i,j}, a_l, a_{l-1} \rangle$;	$\langle q^6_{i,j}, a_r, \bar{\delta} \rangle$,	$r = 1, \ldots, m,$	$r \neq l$;
$\langle q^1_{i,j}, b, q^2_{i,j} \rangle$;	$\langle q^3_{i,j}, a_{l-1}, \delta \rangle$;	$\langle q^6_{i,j}, b, a_l \rangle$;		
$\langle q^2_{i,j}, b, \delta \rangle$;	$\langle q^3_{i,j}, b, q^4_{i,j} \rangle$;	$\langle q^6_{i,j}, a_l, q^7_{i,j} \rangle$;		
	$\langle q^4_{i,j}, b, a_l \rangle$;	$\langle q^7_{i,j}, a_l, \delta \rangle$;		
	$\langle q^4_{i,j}, a_l, \delta \rangle$;	$\langle q^7_{i,j}, a_r, q_k \rangle$,	$r = 1, \ldots, m,$	$r \neq l$.
	$\langle q^4_{i,j}, a_{l-1}, q^5_{i,j} \rangle$;			
	$\langle q^5_{i,j}, a_{l-1}, a_l \rangle$;			
	$\langle q^5_{i,j}, a_l, q_k \rangle$;			

(В записи команд группы (ii) полагаем $a_0 = a_m$.)

Опишем кодирующую и декодирующую функции, с помощью которых машина B моделирует машину A. Каждой конфигурации $\alpha q_i a_j \beta$ машины A соответствует та же конфигурация $\alpha q_i a_j \beta$ машины B. Каждой конфигурации $\alpha q_i a_j \beta$ машины B, где α и β — слова в алфавите символов машины A, соответствует та же конфигурация $\alpha q_i a_j \beta$ машины A.

Поясним идею, лежащую в основе построения машины B.

В случае моделирования сдвига головки ($\delta \in \{R, L\}$) машине B придется в конце концов покинуть ячейку с новым символом a_l и сдвинуть головку (не меняя — в силу операционных возможностей машины B — своего состояния), а потому мы должны быть уверены в том, что машина B не встретит в той ячейке, куда сдвигается головка, такой же символ a_l. (Если это случится, машина B проскочит нужную ячейку и продолжит движение в направлении δ.) Поэтому машина B проверяет содержимое ячейки, куда потом предстоит сдвинуться головке. Если в этой ячейке находится символ a_l (см. группу команд (ii)), машина B заменит его на a_{l-1} (напомним, что $a_0 = a_m$). Затем машина B возвратит головку к первоначально рассмотренной ячейке, запишет там символ a_l и сдвинется к ячейке с символом a_{l-1}. Здесь машина B восстановит первоначальный символ a_l. После этого машина B готова выполнить следующий шаг своей работы.

$\alpha q_i a_j a_l \beta$
$\alpha q^1_{i,j} a_j a_l \beta$
$\alpha q^1_{i,j} b a_l \beta$
$\alpha q^2_{i,j} b a_l \beta$
$\alpha b q^2_{i,j} a_l \beta$
$\alpha b q^3_{i,j} a_l \beta$
$\alpha b q^3_{i,j} a_{l-1} \beta$
$\alpha q^3_{i,j} b a_{l-1} \beta$
$\alpha q^4_{i,j} b a_{l-1} \beta$
$\alpha q^4_{i,j} a_l a_{l-1} \beta$
$\alpha a_l q^4_{i,j} a_{l-1} \beta$
$\alpha a_l q^5_{i,j} a_{l-1} \beta$
$\alpha a_l q^5_{i,j} a_l \beta$
$\alpha a_l q_k a_l \beta$

Рис. 5

Если в ячейке, куда надо сдвинуть головку машины B, находится символ a_r, $r \neq l$, то (см. группу команд (iii)) машина B возвращается к ранее рассмотренной ячейке и записывает туда символ a_l. В ячейке, к которой должна быть перемещена головка, находится символ a_r, отличный от a_l, а потому машина B просто сдвигает туда головку и после этого готова выполнить следующий шаг своей работы.

Поясним сказанное примером. Пусть $\alpha q_i a_j a_l \beta \xrightarrow{A} \alpha a_l q_k a_l \beta$. Тогда $\alpha q_i a_j a_l \beta \overset{B}{\Rightarrow} \alpha a_l q_k a_l \beta$. За работой машины B можно проследить на рис. 5 (применяются команды из групп (i) и (ii)).

Теорема 2. *В классе UM существуют как универсальные машины с* 3 *состояниями, так и универсальные машины с* 3 *символами.*

Доказательство этой теоремы следует из утверждений 3 и 4.

Соответствующие результаты для классов AM, BM, CM и FM следуют из указанных и отражены на рис. 1.

§ 5. Общие принципы построения универсальных машин Тьюринга, моделирующих таг-системы

В § 6 мы построим универсальные машины Тьюринга, которые моделируют таг-системы. Пусть m — натуральное число, A — алфавит $\{a_1, \ldots, a_n, a_{n+1}\}$. Преобразование слов в этом алфавите *m-таг-системой* можно описать следующим образом: первые m букв слова удаляются, а к результату справа приписывается слово, зависящее от первой (самой левой) из удаленных букв. Этот процесс повторяется вновь и вновь, а заканчивается в двух случаях: если очередной результат будет короче m букв или если первой его буквой окажется a_{n+1}. Дадим формальное определение.

Определение 5. *Таг-система* есть тройка $T = (m, A, P)$, где m — натуральное число, A — конечный алфавит, $A = \{a_1, \ldots, a_{n+1}\}$, P — отображение, ставящее в соответствие буквам из множества $\{a_1, \ldots, a_n\}$ слова в алфавите A, а букве a_{n+1} — особое значение STOP.

Таг-система $T = (m, A, P)$ называется *m-таг-системой*. Слова α_i, $i = 1, \ldots, n$, где $\alpha_i = P(a_i)$, называются *продукциями* таг-системы T. Буква a_{n+1} называется *остановочным символом*. Продукции часто записываются следующим образом:

$$\begin{cases} a_i \to \alpha_i, & i \in \{1, \ldots, n\}; \\ a_{n+1} \to \text{STOP}. \end{cases} \tag{4}$$

Вычисление таг-системы T, $T = (m, A, P)$, над словом β в алфавите A есть конечная или бесконечная последовательность $\beta_0, \beta_1, \ldots$ слов в этом же алфавите, в которой $\beta_0 = \beta$ и $\beta_0 \xrightarrow{T} \beta_1 \xrightarrow{T} \beta_2 \xrightarrow{T} \ldots$, т. е. каждое следующее слово β_{k+1} образуется из β_k удалением первых m букв и присоединением справа к результату слова α_i — в том случае, если слово β_k начинается с буквы a_i, $1 \leqslant i \leqslant n$. Если слово β_k имеет длину менее m или начинается с буквы a_{n+1}, вычисление прекращается (останавливается). В этом случае говорят, что вычисление конечно и состоит из k шагов.

Транзитивное замыкание отношения $\xrightarrow{T}$ будем обозначать $\overset{T}{\Rightarrow}$.

Пример. Пусть 2-таг-система T_1 задана продукциями $a_1 \to a_2 a_1 a_3$, $a_2 \to a_1$ и $a_3 \to \text{STOP}$. Для начального слова $\beta = a_2 a_1 a_1$ вычисление этой таг-системы есть $a_2 a_1 a_1 \to a_1 a_1 \to a_2 a_1 a_3 \to a_3 a_1$.

М. Минский доказал существование универсальных 2-таг-систем [33]. Поэтому мы будем, учитывая это доказательство, использовать только 2-таг-системы со следующими двумя дополнительными свойствами:

1) остановка таг-системы происходит только из-за слов, начинающихся с остановочного символа a_{n+1};

2) продукции α_i, $i \in \{1, \ldots, n\}$, не пусты.

Универсальная машина Тьюринга U моделирует таг-системы *) следующим образом. Пусть T — некоторая таг-система с алфавитом A, $A = \{a_1, \ldots, a_{n+1}\}$, и продукциями $a_i \to \alpha_i$. С каждой буквой a_i мы связываем натуральное число N_i и коды A_i и $\widetilde{A}_i$ (возможно равенство $A_i = \widetilde{A}_i$), имеющие вид $A_i = u^{N_i} = \underbrace{uu \ldots u}_{N_i}$ и $\widetilde{A}_i = \widetilde{u}^{N_i}$, где u и $\widetilde{u}$ — слова в алфавите символов машины U.

Коды A_i (или $\widetilde{A}_i$) на ленте машины U разделяются метками (метка также есть слово в алфавите символов машины U Тьюринга). Мы для сокращения записи обычно будем опускать метки, явно выделяя их только при особой необходимости или чтобы избежать разночтений.

Продукция $a_i \to \alpha_i$ таг-системы T, где $\alpha_i = a_{i1} a_{i2} \ldots a_{im_i}$, m_i — длина слова α_i, $i = 1, \ldots, n$, кодируется словом P_i вида **) $P_i = A_{im_i} A_{im_i - 1} \ldots A_{i2} A_{i1}$.

Перерабатываемое таг-системой T начальное слово β, $\beta = a_r a_s a_t \ldots a_w$, кодируется словом S, где $S = A_r A_s A_t \ldots A_w$ (или $S = \widetilde{A}_r \widetilde{A}_s \widetilde{A}_t \ldots \widetilde{A}_w$).

Начальная лента УМТ имеет вид $Q_L \underbrace{P_{n+1} P_n \ldots P_1 P_0}_{P} \underbrace{A_r A_s A_t \ldots A_w}_{S} Q_R$, где Q_L и Q_R — бесконечные (соответственно влево и вправо) части ленты, содержащие только пустые символы, P_{n+1} — код остановочного символа a_{n+1}, P_0 — добавочный код, содержащий несколько меток, в начале работы машина находится в состоянии q_1, причем головка помещена на самую левую букву части S (в случае универсальной машины из ***) $\mathcal{UTM}(2, 18)$ — на самую правую букву кода P_0).

---

*) Здесь и в дальнейшем под таг-системами будем понимать 2-таг-системы.

**) В этой записи опущены метки, как было оговорено чуть выше. — *Ред.*

***) Напомним, что $\mathcal{UTM}(m, n)$ означает класс универсальных машин Тьюринга с m состояниями и n символами.

Пусть T — произвольная таг-система, S_1 и S_2 — коды слов β_1 и β_2 соответственно. Пусть $\beta_1 \xrightarrow{T} \beta_2$. Тогда УМТ U должна выполнять преобразование

$$Q_L P_{n+1} P_n \dots P_1 P_0 S_1 Q_R \overset{U}{\Rightarrow} Q_L P_{n+1} P_n \dots P_1 P_0 R S_2 Q_R$$

(отрезок R соответствует ячейкам, содержащим коды удаленных первых двух символов).

Работу УМТ можно разделить на три этапа:

1) УМТ ищет код P_r, соответствующий коду A_r, и затем удаляет коды A_r и A_s (т. е. удаляет метку между ними);

2) УМТ пишет код P_r (в обратном порядке) на участок Q_R ленты;

3) затем УМТ восстанавливает свою ленту для нового цикла моделирования.

Число N_i, соответствующее символу a_i таг-системы, $1 \leqslant i \leqslant n+1$, обладает тем свойством, что на каждом цикле моделирования имеется в точности N_r меток между кодами P_r и A_r (в случае машины из $\mathcal{UTM}(2, 18)$ имеется $N_r + 1$ меток, но дополнительная метка в P_0 удаляется немедленно в начале первого этапа).

На первом этапе моделирования головка УМТ проходит в части P столько меток, сколько символов u содержит код A_r. После этого этапа лента выглядит следующим образом:

$$Q_L P_{n+1} P_n \dots P_{r+1} P_r P'_{r-1} \dots P'_1 P'_0 R' A'_r A_s A_t \dots A_w Q_R;$$

головка находится на метке между A'_r и A_s. Здесь P'_k, A'_r, R' и ниже P''_k, R'' обозначают соответственно варианты кодов P_k, A_r и измененное содержимое отрезка R. Затем УМТ удаляет эту метку (стирает коды первых двух символов перерабатываемого слова) и переходит ко второму этапу моделирования. После него лента выглядит так:

$$Q_L P_{n+1} P_n \dots P_{r+1} P''_r P''_{r-1} \dots P''_1 P''_0 R'' A_t \dots A_w A_{r1} A_{r2} \dots A_{rm_r} Q_R;$$

головка находится на левом конце P''_r, и машина переходит к третьему этапу моделирования. После него лента имеет вид

$$Q_L P_{n+1} P_n \dots P_1 P_0 R A_t \dots A_w A_{r1} A_{r2} \dots A_{rm_r} Q_R,$$

головка находится на правом конце участка R и начинается новый цикл моделирования.

В § 6 строится серия из 7 универсальных машин Тьюринга и тем самым доказывается

Теорема 3. *Классы* $\mathcal{UTM}(22, 2)$, $\mathcal{UTM}(10, 3)$, $\mathcal{UTM}(7, 4)$, $\mathcal{UTM}(5, 5)$, $\mathcal{UTM}(4, 6)$, $\mathcal{UTM}(3, 10)$ *и* $\mathcal{UTM}(2, 18)$ *универсальных машин Тьюринга не пусты.*

Как было отмечено выше, из этой теоремы и результатов Л. М. Павлоцкой [3, 4] следует, что пока не исследованными (пусты они или нет) остаются 49 классов $\mathcal{UTM}(m, n)$, см. рис. 3.

§ 6. Универсальные машины Тьюринга

Пусть $a_1, a_2, \dots, a_k$, $b_1, b_2, \dots, b_k$ — символы машины Тьюринга. Пусть запись $a_1 a_2 \dots a_k R b_1 b_2 \dots b_k$ (соответственно, $a_1 a_2 \dots a_k L b_1 b_2 \dots b_k$) означает, что при движении головки УМТ вправо (влево) группа символов $a_1 a_2 \dots a_k$ заменяется на $b_1 b_2 \dots b_k$.

Запись $R a_1 a_2 \dots a_k (b_1 b_2 \dots b_k) L$ (соответственно, $L a_1 a_2 \dots a_k (b_1 b_2 \dots b_k) R$) означает, что головка УМТ проходит группу символов $a_1 a_2 \dots a_k$ слева направо (справа налево), заменяя их на $b_1 b_2 \dots b_k$, после чего начинается движение влево (вправо).

Ниже такие записи мы будем обычно использовать как числитель условной «дроби», указывая в знаменателе последовательность выполняемых при этом команд машины Тьюринга.

6.1. УМТ с 22 состояниями и 2 символами (см. [46]). Предъявим машину из $\mathcal{UTM}(22, 2)$, имеющую состояния $q_1, \ldots, q_{22}$ и символы 0 (пустой символ) и 1; ее программа показана в табл. 3.

Пусть $N_1 = 1$, $N_{k+1} = N_k + m_k + 1$, $k = 1, \ldots, n$. Пусть продукция $\alpha_i = a_{i1}a_{i2}\ldots a_{im_i}$ таг-системы, $i = 1, \ldots, n$, кодируется словом P_i, где $P_i = 011001(01)^{N_{im_i}}101001(01)^{N_{im_i-1}}\ldots 101001(01)^{N_{i1}}101001$, т. е. $A_j = (01)^{N_j}$, $j = 1, \ldots, n+1$, а 0110 и 011010 — метки. Пусть, кроме того, $P_0 = 0110$ и $P_{n+1} = 0001$. Пусть начальное слово $\beta = a_r a_s a_t \ldots a_w$, преобразуемое таг-системой, имеет код S, где $S = (01)^{N_r}11(01)^{N_s}11(01)^{N_t}\ldots 11(01)^{N_w}$, т. е. $\tilde{A}_j = (01)^{N_j}$, $j = 1, \ldots, n+1$, и пара 11 — также метка.

Таг-система моделируется следующим образом. На первом этапе имеем $\dfrac{01L00}{q_1 10 L q_2\ q_2 00 L q_1}$, $\dfrac{00L00}{q_1 00 L q_2\ q_2 00 L q_1}$, $\dfrac{L0110(0010)R}{q_1 00 L q_2\ q_2 11 L q_3\ q_3 10 R q_4\ q_4 11 R q_5\ q_5 00 R q_4}$, $\dfrac{0010L0010}{q_1 00 L q_2\ q_2 11 L q_3\ q_3 00 L q_2\ q_2 00 L q_1}$, $\dfrac{L011010(001010)R}{q_1 00 L q_2\ q_2 11 L q_3\ q_3 00 L q_2\ q_2 11 L q_3\ q_3 10 R q_4\ q_3 10 R q_4\ q_4 11 R q_5\ q_5 00 R q_4\ q_4 11 R q_5\ q_5 00 R q_4}$, $\dfrac{00R00}{q_4 00 R q_7\ q_7 00 R q_4}$, $\dfrac{10R10}{q_4 11 R q_5\ q_5 00 R q_4}$ и $\dfrac{R01(00)L}{q_4 00 R q_7\ q_7 10 L q_2\ q_2 00 L q_1}$.

Когда головка УМТ, двигаясь вправо, встречает метку 11, первый этап моделирования заканчивается: $\dfrac{R11(01)L}{q_4 11 R q_5\ q_5 11 L q_6\ q_6 10 L q_8}$, и начинается второй. На втором этапе имеем $\dfrac{00L01}{q_8 00 L q_{13}\ q_{13} 00 R q_{18}\ q_{18} 01 L q_{22}\ q_{22} 00 L q_8}$, $\dfrac{10L10}{q_8 00 L q_{13}\ q_{13} 11 L q_8}$ и $\dfrac{11L11}{q_8 11 L q_9\ q_9 11 L q_8}$.

Если УМТ находится в состоянии q_8 и слева от текущей позиции находится 0101, происходят шаги $\dfrac{L0101(0100)R}{q_8 11 L q_9\ q_9 00 L q_{10}\ q_{10} 11 R q_{17}\ q_{17} 00 R q_{18}\ q_{18} 10 R q_{17}}$ и УМТ пишет пару 01 в Q_R. В этом случае имеем $\dfrac{01R00}{q_{17} 00 R q_{18}\ q_{18} 10 R q_{17}}$, $\dfrac{11R11}{q_{17} 11 R q_{19}\ q_{19} 11 R q_{17}}$, $\dfrac{10R10}{q_{17} 11 R q_{19}\ q_{19} 00 R q_{17}}$, $\dfrac{R00(01)L}{q_{17} 00 R q_{18}\ q_{18} 01 L q_{22}\ q_{22} 00 L q_8}$.

Если слева от текущей позиции находится 101001, УМТ выполняет шаги $\dfrac{L101001(101000)R}{q_8 11 L q_9\ q_9 00 L q_{10}\ q_{10} 00 L q_{11}\ q_{11} 11 L q_{12}\ q_{12} 00 R q_{14}\ q_{14} 11 R q_{16}\ q_{16} 00 R q_{14}\ q_{14} 00 R q_{15}\ q_{15} 10 R q_{14}}$ и пишет метку 11 в Q_R. В этом случае $\dfrac{01R00}{q_{14} 00 R q_{15}\ q_{15} 10 R q_{14}}$, $\dfrac{11R11}{q_{14} 11 R q_{16}\ q_{16} 11 R q_{14}}$, $\dfrac{10R10}{q_{14} 11 R q_{16}\ q_{16} 00 R q_{14}}$ и $\dfrac{R00(11)L}{q_{14} 00 R q_{15}\ q_{15} 01 L q_6\ q_6 01 L q_8}$.

Когда головка, двигаясь влево, встречает группу $0001 = P_{n+1}$, УМТ останавливается, выполняя команды $q_8 11 L q_9$, $q_9 00 L q_{10}$, $q_{10} 00 L q_{11}$ и $q_{11} 0$ —. А если головка, двигаясь влево, встречает группу 011001, второй этап моделирования заканчивается: $\dfrac{L011001(011001)R}{q_8 11 L q_9\ q_9 00 L q_{10}\ q_{10} 00 L q_{11}\ q_{11} 11 L q_{12}\ q_{12} 11 R q_{20}\ q_{20} 11 R q_{21}\ q_{21} 00 R q_{20}\ q_{20} 00 R q_{12}\ q_{12} 11 R q_{20}}$, и начинается третий. На третьем этапе имеем $\dfrac{10R10}{q_{20} 11 R q_{21}\ q_{21} 00 R q_{20}}$ и $\dfrac{01R01}{q_{20} 00 R q_{12}\ q_{12} 11 R q_{20}}$.

Таблица 3

Программа УМТ с 22 состояниями и 2 символами

$q_1 00 L q_2$	$q_4 00 R q_7$	$q_7 00 R q_4$	$q_{10} 00 L q_{11}$	$q_{13} 00 R q_{18}$	$q_{16} 00 R q_{14}$	$q_{19} 00 R q_{17}$	$q_{22} 00 L q_8$
$q_1 10 L q_2$	$q_4 11 R q_5$	$q_7 10 L q_2$	$q_{10} 11 R q_{17}$	$q_{13} 11 L q_8$	$q_{16} 11 R q_{14}$	$q_{19} 11 R q_{17}$	$q_{22} 10 R q_7$
$q_2 00 L q_1$	$q_5 00 R q_4$	$q_8 00 L q_{13}$	$q_{11} 0$ —	$q_{14} 00 R q_{15}$	$q_{17} 00 R q_{18}$	$q_{20} 00 R q_{12}$	
$q_2 11 L q_3$	$q_5 11 L q_6$	$q_8 11 L q_9$	$q_{11} 11 L q_{12}$	$q_{14} 11 R q_{16}$	$q_{17} 11 R q_{19}$	$q_{20} 11 R q_{21}$	
$q_3 00 L q_2$	$q_6 01 L q_8$	$q_9 00 L q_{10}$	$q_{12} 00 R q_{14}$	$q_{15} 01 L q_6$	$q_{18} 01 L q_{22}$	$q_{21} 00 R q_{20}$	
$q_3 10 R q_4$	$q_6 10 L q_8$	$q_9 11 L q_8$	$q_{12} 11 R q_{20}$	$q_{15} 10 R q_{14}$	$q_{18} 10 R q_{17}$	$q_{21} 10 L q_{22}$	

Когда головка УМТ, двигаясь вправо, встречает метку 11 (перед кодом $\widetilde{A}_t$), третий этап (и вместе с ним — весь цикл моделирования) заканчивается. УМТ удаляет метку 11, и начинается новый цикл моделирования: $\dfrac{11R00}{q_{20}11Rq_{21}\ q_{21}10Lq_{22}\ q_{22}10Rq_7\ q_700Rq_4}$.

6.2. УМТ с 10 состояниями и 3 символами (см. [7, 45]). Предъявим машину из $\mathcal{UTM}(10,3)$, имеющую состояния $q_1,\ldots,q_{10}$ и символы 0 (пустой символ), 1 и b. Ее программа показана в табл. 4. Пусть

$$N_1=2,\qquad N_{k+1}=N_k+m_k+t_k,\quad t_k=\begin{cases}2, & m_k \text{ четно},\\ 1, & m_k \text{ нечетно},\end{cases}\qquad k=1,\ldots,n.$$

(Очевидно, что все числа $N_1,\ldots,N_{n+1}$ четны.) Пусть продукция $\alpha_i=a_{i1}a_{i2}\ldots a_{im_i}$ таг-системы, $i=1,\ldots,n$, кодируется словом P_i, где

$$P_i=\begin{cases}b0b00^{N_{im_i}}00b00^{N_{im_i-1}}00b\ldots00b00^{N_{i1}}0, & m_i \text{ нечетно},\\ b0b0b00^{N_{im_i}}00b00^{N_{im_i-1}}00b\ldots00b00^{N_{i1}}0, & m_i \text{ четно};\end{cases}$$

т. е. $A_j=0^{N_j}$, $j=1,\ldots,n+1$, а слово $b0$ есть метка. Пусть, кроме того, $P_0=b0b0b$ и $P_{n+1}=10$. Пусть начальное слово $\beta=a_r a_s a_t\ldots a_w$, преобразуемое таг-системой, имеет код S, где $S=1^{N_r}bb1^{N_s}bb1^{N_t}\ldots bb1^{N_w}$, т. е. $\widetilde{A}_j=1^{N_j}$, $j=1,\ldots,n+1$, и пара символов bb — также метка.

Моделирование таг-системы протекает следующим образом. На первом этапе имеем $\dfrac{b1Lb0}{q_210Lq_2\ q_2bbLq_2}$, $\dfrac{1L0}{q_210Lq_2}$, $\dfrac{00L00}{q_200Lq_3\ q_300Lq_2}$, $\dfrac{Lb0(b1)R}{q_200Lq_3\ q_3bbRq_1}$, $\dfrac{b0Rb1}{q_1bbRq_4\ q_401Rq_1}$, $\dfrac{0R1}{q_101Rq_1}$, $\dfrac{R1(0)L}{q_110Lq_2}$.

Первый этап моделирования завершен, когда головка УМТ, двигаясь вправо, встречает метку bb. Она заменяется на 11: $\dfrac{Rbb(11)L}{q_1bbRq_4\ q_4b1Lq_4}$, затем УМТ пишет метку bb в Q_R и начинается второй этап моделирования. На этом этапе происходит следующее: $\dfrac{bbLbb}{q_6bbLq_6}$, $\dfrac{b1Lb1}{q_611Lq_6\ q_6bbLq_6}$, $\dfrac{1L1}{q_611Lq_6}$.

Если УМТ находится в состоянии q_6 и слева от текущей позиции находится слово 00, происходят шаги $\dfrac{L00(01)R}{q_601Lq_7\ q_700Rq_8\ q_811Rq_8}$ и УМТ пишет символ 1 в Q_R. В этом случае $\dfrac{bRb}{q_8bbRq_8}$, $\dfrac{1R1}{q_811Rq_8}$ и $\dfrac{R0(1)L}{q_801Lq_6}$.

Если же слева от текущей позиции находится 00b0, происходят шаги $\dfrac{L00b0(01b1)R}{q_601Lq_7\ q_7bbLq_9\ q_901Lq_{10}\ q_{10}00Rq_5}$ и УМТ пишет метку bb в Q_R. В этом случае $\dfrac{bRb}{q_5bbRq_5}$, $\dfrac{1R1}{q_511Rq_5}$ и $\dfrac{R10(bb)L}{q_511Rq_5\ q_50bLq_3\ q_31bLq_6}$.

Когда головка, двигаясь влево, встречает пару $10=P_{n+1}$, УМТ останавливается (q_601Lq_7, q_71 —). А если головка, двигаясь влево, встречает группу $b0b0$, второй этап моделирования заканчивается: $\dfrac{Lb0b0(b0b0)R}{q_601Lq_7\ q_7bbLq_9\ q_901Lq_{10}\ q_{10}bbRq_9\ q_910Rq_{10}}$, и начинается третий. На третьем этапе происходят следующие действия: $\dfrac{b1Rb0}{q_{10}bbRq_9\ q_910Rq_{10}}$, $\dfrac{1R0}{q_{10}10Rq_{10}}$.

Таблица 4

Программа УМТ с 10 состояниями и 3 символами

q_101Rq_1	q_601Lq_7
q_110Lq_2	q_611Lq_6
q_1bbRq_4	q_6bbLq_6
q_200Lq_3	q_700Rq_8
q_210Lq_2	q_71 —
q_2bbLq_2	q_7bbLq_9
q_300Lq_2	q_801Lq_6
q_31bLq_6	q_811Rq_8
q_3bbRq_1	q_8bbRq_8
q_401Rq_1	q_901Lq_{10}
q_411Rq_5	q_910Rq_{10}
q_41bLq_4	q_9b0Lq_4
q_50bLq_3	$q_{10}00bRq_5$
q_511Rq_5	$q_{10}10Rq_{10}$
q_5bbRq_5	$q_{10}bbRq_9$

Когда головка УМТ, двигаясь вправо, встречает метку bb, третий этап (и вместе с ним — весь цикл моделирования) заканчивается. УМТ удаляет метку bb, и начинается новый цикл моделирования: $\dfrac{Rbb(10)L}{q_{10}bbRq_9\ q_9b0Lq_4\ q_4b1Lq_4}$. Через несколько шагов головка окажется на левом конце участка S.

6.3. УМТ с 7 состояниями и 4 символами (см. [6, 45]). Предъявим машину из $\mathcal{UTM}(7,4)$, имеющую состояния $q_1, \ldots, q_7$ и символы 0 (пустой символ), 1, b и c. Ее программа приведена в табл. 5.

Таблица 5

Программа УМТ с 7 состояниями и 4 символами

q_100Lq_1	q_201Rq_2	q_301Lq_4	q_401Lq_7	q_50cLq_4	q_600Rq_5	q_700Rq_3
q_110Lq_1	q_210Lq_1	q_311Rq_3	q_411Lq_4	q_511Rq_5	q_610Rq_6	q_71 —
q_1bcRq_2	q_2bcRq_2	q_3bcRq_3	q_4bcLq_4	q_5bcRq_5	q_6bbRq_6	q_7bbLq_6
q_1cbLq_1	q_2c1Rq_5	q_3cbRq_3	q_4cbLq_4	q_5cbRq_5	q_6c0Rq_1	q_7c —

Пусть $N_1 = 1$ и $N_{k+1} = N_k + m_k + 1$, $k = 1, \ldots, n$. Пусть продукция $\alpha_i = a_{i1}a_{i2}\ldots a_{im_i}$ таг-системы, $i = 1, \ldots, n$, кодируется словом P_i, где $P_i = bb00^{N_{im_i}}b00^{N_{im_i-1}}\ldots b00^{N_{i2}}b00^{N_{i1}}$, т. е. $A_j = 0^{N_j}$, $j = 1, \ldots, n+1$, символ b — метка. Пусть также $P_0 = b0$, $P_{n+1} = 10$. Пусть начальное слово $\beta = a_r a_s a_t \ldots a_w$, преобразуемое таг-системой, имеет код S, где $S = 1^{N_r}c1^{N_s}c1^{N_t}\ldots c1^{N_w}$, т. е. $\widetilde{A}_j = 1^{N_j}$, $j = 1, \ldots, n+1$, символ c — также метка.

На первом этапе моделирования таг-системы имеем $\frac{c1Lb0}{q_110Lq_1\ q_1cbLq_1}$, $\frac{1L0}{q_110Lq_1}$, $\frac{0L0}{q_100Lq_1}$, $\frac{b0Rc1}{q_2bcRq_2\ q_201Rq_2}$, $\frac{Lb0(c1)R}{q_100Lq_1\ q_1bcRq_2\ q_201Rq_2}$, $\frac{0R1}{q_201Rq_2}$ и $\frac{R1(0)L}{q_210Lq_1}$.

Первый этап моделирования завершен, когда головка УМТ, двигаясь вправо, встречает метку c. Она заменяется на символ 1: $\frac{cR1}{q_2c1Rq_5}$, затем УМТ пишет метку c в Q_R и начинается второй этап моделирования. На втором этапе происходит следующее: $\frac{cLb}{q_4cbLq_4}$, $\frac{bLc}{q_4bcLq_4}$, $\frac{1L1}{q_411Lq_4}$.

Если УМТ находится в состоянии q_4 и слева от текущей позиции находится слово 00, выполняются шаги $\frac{L00(01)R}{q_401Lq_7\ q_700Rq_3\ q_311Rq_3}$ и УМТ пишет символ 1 в Q_R. В этом случае $\frac{cRb}{q_3cbRq_3}$, $\frac{bRc}{q_3bcRq_3}$, $\frac{1R1}{q_311Rq_3}$ и $\frac{R0(1)L}{q_301Lq_4}$.

Если же слева от текущей позиции находится 0b0, происходят шаги $\frac{L0b0(0c1)R}{q_401Lq_7\ q_7bbLq_6\ q_600Rq_5\ q_5bcRq_5\ q_511Rq_5}$ и УМТ пишет метку c в Q_R. В этом случае $\frac{cRb}{q_5cbRq_5}$, $\frac{bRc}{q_5bcRq_5}$, $\frac{1R1}{q_511Rq_5}$ и $\frac{R0(c)L}{q_50cLq_4}$.

Когда головка, двигаясь влево, встречает пару $10 = P_{n+1}$, УМТ останавливается (q_401Lq_7, q_71 —).

Если же головка, двигаясь влево, встречает группу $bb0$, второй этап моделирования заканчивается: $\frac{Lbb0(bb0)R}{q_401Lq_7\ q_7bbLq_6\ q_6bbRq_6\ q_610Rq_6}$, и начинается третий. На третьем этапе имеем $\frac{b1Rb0}{q_6bbRq_6\ q_610Rq_6}$ и $\frac{1R0}{q_610Rq_6}$.

Когда головка УМТ, двигаясь вправо, встречает метку c, третий этап (и вместе с ним — весь цикл моделирования) заканчивается. УМТ удаляет метку c, и начинается новый цикл моделирования: $\frac{cR0}{q_6c0Rq_1}$.

6.4. УМТ с 5 состояниями и 5 символами (см. [6, 45]). Предъявим машину из класса $\mathcal{UTM}(5,5)$, которая моделирует следующий класс таг-систем:

$$\begin{cases} a_i \to b\alpha_i a, & i \in \{1, \ldots, n\}; \\ a \to \varepsilon; \\ b \to \varepsilon; \\ a_{n+1} \to \text{STOP} \end{cases} \tag{5}$$

(здесь $\alpha_i = a_{i1}a_{i2}\ldots a_{im_i}$ есть непустое конечное слово в алфавите $\{a_1, \ldots, a_{n+1}\}$, ε — пустое слово).

Замечание 5. Процесс преобразования слова βa таг-системой T' типа (5) описывается последовательностью $\beta a \xrightarrow{T'} \beta_1 a \xrightarrow{T'} \ldots \xrightarrow{T'} \beta_j a \xrightarrow{T'} \ldots$, причем слова β_j, которые не начинаются с символов a и b, имеют вид $\beta_{j1} ab \beta_{j2} ab \ldots \beta_{jk_j - 1} ab \beta_{jk_j}$, где подслова β_{jr} при $r = 1, \ldots, k_j$ не содержат символов a и b, $\beta_{j1} \neq \varepsilon$ и $\beta_{jk_j} \neq \varepsilon$.

Универсальность таг-систем типа (5) доказывает следующая

Лемма 4. *Для каждой таг-системы T типа* (4) *существует моделирующая ее таг-система T' типа* (5).

Доказательство. Формально преобразуем данную таг-систему T к типу (5), добавляя новые буквы a и b к правой части каждого правила вывода $a_i \to \alpha_i$, а также вводя два новых правила: $a \to \varepsilon$ и $b \to \varepsilon$.

Покажем, что если $\beta \overset{T}{\Rightarrow} \gamma$, то $\beta a \overset{T'}{\Rightarrow} \gamma' a$, где первая буква в слове γ' отлична от a и b, а слово γ получается из γ' удалением всех вхождений символов a и b. Применим индукцию по числу шагов преобразования $\beta \overset{T}{\Rightarrow} \gamma$.

База индукции. Пусть $a_i a_j \beta \xrightarrow{T} \beta \alpha_i$. Тогда $a_i a_j \beta a \xrightarrow{T} \beta a b \alpha_i a$. Если $\beta = \varepsilon$, то $a_i a_j a \xrightarrow{T'} a b \alpha_i a \xrightarrow{T'} \alpha_i a$. Но $\alpha_i \neq \varepsilon$, а потому все доказано.

Индуктивное предположение. Пусть $\beta \overset{T}{\Rightarrow} \gamma_t \xrightarrow{T} \gamma$ и $\beta a \overset{T'}{\Rightarrow} \gamma_t' a$, где первая буква слова γ_t' отлична от a и b, а слово γ_t получается из γ_t' удалением всех вхождений букв a и b.

Шаг индукции. Рассмотрим два случая.

1. Пусть $\gamma_t' = a_i a_j \delta_t'$, где $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, n+1$. Тогда $\gamma_t = a_i a_j \delta_t$, где слово δ_t получается из δ_t' удалением всех вхождений символов a и b, и $a_i a_j \delta_t \xrightarrow{T} \delta_t \alpha_i$ ($\gamma = \delta_t \alpha_i$), но $a_i a_j \delta_t' a \xrightarrow{T'} \delta_t' a b \alpha_i a$.

Если $\delta_t = \varepsilon$, то с учетом замечания 5 и $\delta_t' = \varepsilon$. Тогда $\delta_t' a b \alpha_i a = a b \alpha_i a \xrightarrow{T'} \xrightarrow{T'} \alpha_i a$, и требуемое γ' — это α_i.

Если $\delta_t \neq \varepsilon$, то с учетом замечания 5 либо первая буква слова δ_t' отлична от a и b (тогда $\gamma' = \delta_t' a b \alpha_i$), либо $\delta_t' = ab\delta_t''$, где первая буква слова δ_t'' отлична от a и b, и δ_t получается из δ_t'' удалением всех вхождений букв a и b. В последнем случае $ab\delta_t'' a b \alpha_i a \xrightarrow{T'} \delta_t'' a b \alpha_i a$, и требуемое γ' есть $\delta_t'' a b \alpha_i$.

2. Пусть $\gamma_t' = a_i a \delta_t'$, где $i = 1, 2, \ldots, n$. Тогда $\delta_t' = b\delta_t''$ и $\gamma_t = a_i \delta_t$, где δ_t получается из δ_t'' удалением всех вхождений букв a и b.

Пусть $\delta_t = a_j \delta_{t1}$, $j = 1, 2, \ldots, n+1$. Тогда $a_i a_j \delta_{t1} \xrightarrow{T} \delta_{t1} \alpha_i$, $\gamma = \delta_{t1} \alpha_i$. Учитывая замечание 5, имеем $\delta_t'' = a_j \delta_{t1}'$; слово δ_{t1} получается из δ_{t1}' удалением всех вхождений букв a и b. Значит, $a_i a b a_j \delta_{t1}' a \xrightarrow{T'} b a_j \delta_{t1}' a b \alpha_i a \xrightarrow{T'} \delta_{t1}' a b \alpha_i a$.

Если $\delta_{t1} = \varepsilon$, то и $\delta_{t1}' = \varepsilon$. Тогда $\delta_{t1}' a b \alpha_i a \to \alpha_i a$ и $\gamma' = \alpha_i$. Если $\delta_{t1} \neq \varepsilon$, то либо первая буква слова δ_{t1}' отлична от a и b (в этом случае $\gamma' = \delta_{t1}' a b \alpha_i$), либо $\delta_{t1}' = ab\delta_{t1}''$, где первая буква слова δ_{t1}'' отлична от a и b, а δ_{t1} получается из δ_{t1}'' удалением всех вхождений букв a и b. В последнем случае $ab\delta_{t1}'' a b \alpha_i a \xrightarrow{T'} \delta_{t1}'' a b \alpha_i a$ и требуемое слово γ' есть $\delta_{t1}'' a b \alpha_i$.

Лемма 4 доказана.

Теперь перейдем собственно к построению машины из $\mathcal{UTM}(5, 5)$. Она имеет состояния $q_1, \ldots, q_5$ и символы 0, 1, b (пустой символ), c и d; программа приведена в табл. 6.

Таблица 6

Программа УМТ с 5 состояниями и 5 символами

$q_1 01 R q_1$	$q_2 00 R q_2$	$q_3 0 c L q_4$	$q_4 01 L q_4$	$q_5 0$ —
$q_1 10 L q_1$	$q_2 10 R q_2$	$q_3 10 R q_3$	$q_4 10 R q_2$	$q_5 11 R q_5$
$q_1 bd R q_1$	$q_2 b0 L q_4$	$q_3 bb R q_5$	$q_4 bd L q_3$	$q_5 b$ —
$q_1 c0 R q_2$	$q_2 cc R q_2$	$q_3 cc R q_3$	$q_4 cc L q_4$	$q_5 c1 R q_1$
$q_1 db L q_1$	$q_2 dd R q_2$	$q_3 dd R q_3$	$q_4 dd L q_4$	$q_5 db R q_5$

Пусть $N_1 = 3$, $N_{k+1} = N_k + m_k + 4$ при $k = 1, \ldots, n$, $N_a = N_{n+1} + 2$ и $N_b = 1$. Пусть N — произвольное число, бóльшее N_a. Пусть продукция $\alpha_i = a_{i1} a_{i2} \ldots a_{im_i}$ таг-системы, $i = 1, \ldots, n$, кодируется словом P_i, где $P_i = bb1^{N_{im_i}} 1 b 1^{N_{im_i - 1}} 1 b 1 1^{N_{im_i - 1}} b \ldots 1 b 1 1^{N_{i1}} 1 b 1 1^{N_{i1}} 1 b$, т. е. $A_j = 1^{N_j}$, $j = 1, \ldots, n+1$, $A = 1^{N_a}$, $B = 1^{N_b}$, символ b — метка. Пусть также $P_0 = bbb$, $P_{n+1} = 1b1b$.

Пусть начальное слово $\beta = a_r a_s a_t \ldots a_w$, преобразуемое таг-системой, имеет код S, где $S = 1^{N_r} c 1^{N_s} c 1^{N_t} \ldots c 1^{N_w} c 1^{N_v}$, т. е. $\widetilde{A}_j = 1^{N_j}$, $j = 1, \ldots, n+1$, $\widetilde{A} = 1^{N_v}$, $\widetilde{B} = 1^{N_b}$, символ c — также метка.

На первом этапе моделирования таг-системы имеем $\frac{dLb}{q_1 dbL q_1}$, $\frac{1L0}{q_1 10 L q_1}$, $\frac{Lb(d)R}{q_1 bdR q_1}$, $\frac{0R1}{q_1 01 R q_1}$, $\frac{bRd}{q_1 bdR q_1}$ и $\frac{R1(0)L}{q_1 10 L q_1}$.

Когда головка УМТ, двигаясь вправо, встречает метку c, первый этап моделирования заканчивается. УМТ заменяет метку c на символ 0: $\frac{cR0}{q_1 c0 R q_2}$, и начинается второй этап моделирования. На втором этапе происходит следующее: $\frac{dLb}{q_4 ddL q_4}$, $\frac{cLc}{q_4 ccL q_4}$ и $\frac{0L1}{q_4 01 L q_4}$.

Если текущее состояние — q_4, а текущий символ — 1, выполняется $\frac{L1(0)R}{q_4 10 R q_2}$ и УМТ пишет символ 0 в Q_R. В этом случае имеем $\frac{dRd}{q_2 ddR q_2}$, $\frac{cRc}{q_2 ccR q_2}$, $\frac{1R0}{q_2 10 R q_2}$ $\left(\text{или } \frac{0R0}{q_2 00 R q_2}, \text{ если ранее УМТ написала символ } 0\right)$, затем $\frac{Rb(0)L}{q_2 b0 L q_4}$.

После первого этапа моделирования код P'_k, $k = 0, \ldots, r-1$, отличается от P_k только тем, что метки b заменены на метки d, поэтому УМТ запишет в Q_R столько же символов 0, сколько символов 1 было между кодами P_r и S. В результате получим новый код A символа a.

Если текущее состояние — q_4 и влево от головки имеем сочетание $1b$, выполняется $\frac{L1b(0d)R}{q_4 bdL q_3 \; q_3 10 R q_3 q_3 ddR q_3}$, и УМТ пишет метку c в Q_R. В этом случае имеем $\frac{dRb}{q_3 ddR q_3}$, $\frac{cRc}{q_3 ccR q_3}$, $\frac{1R0}{q_3 10 R q_3}$ и $\frac{R0(c)L}{q_3 0cL q_4}$.

Если головка УМТ, двигаясь влево, встречает группу $1b1b = P_{n+1}$, то машина попытается записать в Q_R две метки c подряд и в результате остановится ($q_3 bbR q_5$, $q_5 b$ —).

Если же головка УМТ, двигаясь влево, встречает пару bb, второй этап моделирования завершается: $\frac{Lbb(bb)R}{q_4 bdL q_3 \; q_3 bbR q_5 \; q_5 dbR q_5}$.

Если к началу нового цикла моделирования имеем $A_r = A$, то на первом этапе головка обязательно выйдет в зону Q_L и на втором этапе в Q_R будет записано столько символов 0, сколько символов 1 содержалось в P. В результате получится новый код A символа a. Затем (на втором этапе) УМТ встретит в Q_L пару bb, что и вызовет переход к третьему этапу моделирования.

Если к началу нового цикла моделирования окажется $A_r = B$, то на первом этапе в P_0 встретится пара bb, что немедленно вызовет переход к третьему этапу. На третьем этапе моделирования имеем $\frac{dRb}{q_5 dbR q_5}$ и $\frac{1R1}{q_5 11 R q_5}$.

Когда головка УМТ, двигаясь вправо, встречает метку c, третий этап (и вместе с ним — весь цикл моделирования) заканчивается. Метка c удаляется, и начинается новый цикл моделирования: $\frac{cR1}{q_5 c1 R q_1}$.

6.5. УМТ с 4 состояниями и 6 символами (см. [6, 45]). Предъявим машину из класса $\mathcal{UTM}(4, 6)$, которая моделирует следующий класс таг-систем:

$$\begin{cases} a_i \to \alpha_i, & i \in \{1, \ldots, n\}, \\ a_n \to a_n a_n, \\ a_{n+1} \to \text{STOP}; \end{cases} \tag{6}$$

здесь $\alpha_i = a_n a_n \beta_i$ и $\beta_i \neq \varepsilon$.

Универсальность таг-систем типа (6) утверждает следующая

Лемма 5. *Для каждой таг-системы T типа* (4) *существует моделирующая ее таг-система T' типа* (6).

Доказательство этой леммы аналогично доказательству леммы 4.

Перейдем к описанию машины из класса $\mathcal{UTM}(4, 6)$. Она имеет состояния $q_1, \ldots, q_4$ и символы 0 (пустой символ), 1, b, $\overleftarrow{b}$, $\vec{b}$ и c. Ее программа приведена в табл. 7.

Таблица 7

Программа УМТ с 4 состояниями и 6 символами

$q_1 0 \overleftarrow{b} L q_1$	$q_2 0 1 L q_2$	$q_3 0 c R q_1$	$q_4 0 c L q_2$
$q_1 1 \overleftarrow{b} L q_1$	$q_2 1 0 R q_2$	$q_3 1 1 R q_3$	$q_4 1 0 R q_4$
$q_1 b \vec{b} R q_1$	$q_2 b \vec{b} L q_3$	$q_3 b \overleftarrow{b} R q_4$	$q_4 b c L q_2$
$q_1 \vec{b} b L q_1$	$q_2 \vec{b} \overleftarrow{b} R q_2$	$q_3 \vec{b} b R q_3$	$q_4 \vec{b} \overleftarrow{b} R q_4$
$q_1 \overleftarrow{b} 0 R q_1$	$q_2 \overleftarrow{b} \vec{b} L q_2$	$q_3 \overleftarrow{b}$ —	$q_4 \overleftarrow{b}$ —
$q_1 c 0 R q_4$	$q_2 c b R q_2$	$q_3 c 1 R q_1$	$q_4 c b R q_4$

Пусть $N_1 = 1$, $N_{k+1} = N_k + 2m_k$ при $k = 1, \ldots, n$, Заметим, что $m_n = 2$. Пусть продукция $\alpha_i = a_n a_n a_{i3} \ldots a_{im_i}$ таг-системы, $i = 1, \ldots, n$, кодируется словом P_i, где $P_i = b1b1^{N_{im_i}} bb1^{N_{im_i - 1}} \ldots \ldots bb1^{N_{i3}} bb1^{N_n} bb1^{N_n - N_i}$, т. е. $A_j = 1^{N_j}$, $j = 1, \ldots, n+1$, а символ b — метка. В частности, $P_n = b1b1^{N_n} bb$. Пусть также $P_0 = b$, $P_{n+1} = \overleftarrow{b}b$. Пусть начальное слово $\beta = a_r a_s a_t \ldots a_w$, преобразуемое таг-системой, имеет код S, где $S = 1^{N_r} c 1^{N_s} c 1^{N_t} \ldots c 1^{N_w}$, т. е. символ c — также метка. На первом этапе моделирования таг-системы имеем $\frac{1L\overleftarrow{b}}{q_1 1 \overleftarrow{b} L q_1}$, $\frac{0L\overleftarrow{b}}{q_1 0 \overleftarrow{b} L q_1}$, $\frac{\vec{b}Lb}{q_1 \vec{b} b L q_1}$, $\frac{Lb(\vec{b})R}{q_1 b \vec{b} R q_1}$, $\frac{\overleftarrow{b}R0}{q_1 \overleftarrow{b} 0 R q_1}$, $\frac{bR\vec{b}}{q_1 b \vec{b} R q_1}$ и $\frac{R1(\overleftarrow{b})L}{q_1 1 \overleftarrow{b} L q_1}$.

Первый этап моделирования закончен, когда головка УМТ, двигаясь вправо, встречает метку c. В этот момент лента имеет вид

$$Q_L P_{n+1} P_n \ldots P_{r+1} P_r P'_{r-1} \ldots P'_1 P'_0 R' A'_r A_s A_t \ldots A_w Q_R,$$

головка находится над символом c, расположенным между кодами A'_r и A_s, где $A'_j = 0^{N_j}$, $j = 1, \ldots, n+1$. Коды P'_k, $k = 0, \ldots, r-1$, отличаются от P_k тем, что метки b изменены на $\vec{b}$, а A_j — на A'_j, $j = 1, \ldots, n+1$, т. е.

$$P'_k = \vec{b}0\vec{b}0^{N_{km_k}} \vec{b}\,\vec{b}0^{N_{km_k - 1}} \ldots \vec{b}\,\vec{b}0^{N_{k3}} \vec{b}\,\vec{b}0^{N_n} \vec{b}\,\vec{b}0^{N_n - N_k}.$$

После этого УМТ удаляет метку c (команда $q_1 c 0 R q_4$) и переходит ко второму этапу моделирования, записывая метку c в зону Q_R. На втором этапе происходит следующее: $\frac{0L1}{q_2 01 L q_2}$, $\frac{\overleftarrow{b}L\vec{b}}{q_2 \overleftarrow{b} \vec{b} L q_2}$, $\frac{0bL1c}{\begin{matrix} q_2 b \vec{b} L q_3 & q_3 0 c R q_1 & q_1 \vec{b} b L q_1 \\ q_1 c 0 R q_4 & q_4 b c L q_2 & q_2 01 L q_2 \end{matrix}}$. Затем $\frac{L\vec{b}(\overleftarrow{b})R}{q_2 \vec{b} \overleftarrow{b} R q_2}$, и УМТ записывает символ 1 в Q_R.

После первого этапа моделирования между кодами P_r и S имеется в точности N_r меток $\vec{b}$, поэтому УМТ запишет N_r символов 1 в Q_r. После этого (когда головка будет находиться внутри кода P_r), в Q_r будет записано еще $N_n - N_r$ символов 1. В результате получится код A_n.

Далее, $\frac{L1(0)R}{q_2 10 R q_2}$, и УМТ пишет символ 1 в Q_R. В этом случае имеем $\frac{1R0}{q_2 10 R q_2}$, $\frac{\vec{b}R\overleftarrow{b}}{q_2 \vec{b} \overleftarrow{b} R q_2}$, $\frac{cRb}{q_2 cbR q_2}$ и $\frac{R0(1)L}{q_2 01 L q_2}$. Наконец, $\frac{Lbb(\vec{b}\overleftarrow{b})R}{q_2 b \vec{b} L q_3 \;\; q_3 b \overleftarrow{b} R q_4 \;\; q_4 \vec{b} \overleftarrow{b} R q_4}$ и УМТ пишет метку c в Q_R. В этом случае имеем $\frac{1R0}{q_4 10 R q_4}$, $\frac{\vec{b}R\overleftarrow{b}}{q_4 \vec{b} \overleftarrow{b} R q_4}$, $\frac{cRb}{q_4 cbR q_4}$ и $\frac{R0(c)L}{q_4 0 c L q_2}$. УМТ останавливается, когда головка, двигаясь влево, встречает пару $\overleftarrow{b}b$ (команды $q_2 b \vec{b} L q_3$ и $q_3 \overleftarrow{b}$ —).

Второй этап моделирования закончен, когда головка УМТ, двигаясь влево, встречает пару $1b$. В этот момент лента имеет вид

$$Q_L P_{n+1} P_n \ldots P_{r+1} P''_r P''_{r-1} \ldots P''_1 P''_0 R'' A_t \ldots A_w A_n A_n A_{r3} A_{r4} \ldots A_{rm_r} Q_R$$

и головка находится над вторым (слева) символом кода P''_r (код P''_k, $k = 0, \ldots, r$, отличается от P_k тем, что метки b заменены на $\vec{b}$). Затем $\frac{L1b(1b)R}{q_2 b \vec{b} L q_3 \;\; q_3 11 R q_3 \;\; q_3 \vec{b} b R q_3}$, и УМТ переходит к третьему этапу моделирования.

На третьем этапе моделирования УМТ восстанавливает ленту в части P: $\frac{\vec{b}Rb}{q_3\vec{b}bRq_3}$, $\frac{1R1}{q_3\vec{1}1Rq_3}$. Третий этап (и весь цикл моделирования) завершены, когда головка, двигаясь вправо, встречает метку c. УМТ удаляет эту метку и начинается новый цикл моделирования (команда q_3c1Rq_1).

6.6. УМТ с 3 состояниями и 10 символами (см. [44, 45]). Предъявим машину из класса $\mathcal{UTM}(3, 10)$, имеющую состояния q_1, q_2 и q_3 и символы 0 (пустой символ), 1, $\vec{1}$, $\overleftarrow{1}$, b, $\overleftarrow{b}$, $\vec{b}$, c, $\overleftarrow{c}$ и $\vec{c}$. Ее программа приведена в табл. 8. Пусть $N_1 = 1$, $N_{k+1} = N_k + 2m_k + 2$, $k = 1, \ldots, n$. Пусть продукция $\alpha_i = a_{i1}a_{i2}\ldots a_{im_i}$ таг-системы, $i = 1, \ldots, n$, кодируется словом P_i, где $P_i = b1bbb1^{N_{im_i}}bb1^{N_{im_i-1}}\ldots bb1^{N_{i2}}bb1^{N_{i1}}$, т. е. $A_j = 1^{N_j}$, $j = 1, \ldots, n+1$, а символ b — метка. Пусть, кроме того, $P_0 = b$ и $P_{n+1} = \vec{c}b$. Пусть начальное слово $\beta = a_r a_s a_t \ldots a_w$, преобразуемое таг-системой, имеет код S, где $S = 1^{N_r}c1^{N_s}c1^{N_t}\ldots c1^{N_w}c$, т. е. символ c — также метка.

Таблица 8

Программа УМТ с 3 состояниями и 10 символами

$q_1 0\, cL q_3$	$q_2 0\, \vec{1} L q_2$	$q_3 0$ —
$q_1 1\, \vec{1} L q_1$	$q_2 1\, \vec{1} R q_2$	$q_3 1\, 1 R q_3$
$q_1 \vec{1}\, \vec{1} L q_1$	$q_2 \vec{1}\, \vec{1} L q_2$	$q_3 \vec{1}\, 1 L q_3$
$q_1 \vec{1}\, \overleftarrow{1} R q_1$	$q_2 \vec{1}\, \overleftarrow{1} R q_2$	$q_3 \vec{1}\, 1 R q_3$
$q_1 b\, \overleftarrow{b} R q_1$	$q_2 b\, \vec{b} L q_3$	$q_3 b\, \overleftarrow{b} R q_1$
$q_1 \overleftarrow{b}\, bL q_1$	$q_2 \overleftarrow{b}\, \vec{b} L q_2$	$q_3 \overleftarrow{b}\, \vec{b} L q_2$
$q_1 \vec{b}\, \overleftarrow{b} R q_1$	$q_2 \vec{b}\, \overleftarrow{b} R q_2$	$q_3 \vec{b}\, bR q_3$
$q_1 c\, \vec{1} L q_2$	$q_2 c\, \overleftarrow{c} R q_2$	$q_3 c\, 1 R q_1$
$q_1 \overleftarrow{c}$ —	$q_2 \overleftarrow{c}\, cL q_2$	$q_3 \overleftarrow{c}\, cL q_3$
$q_1 \vec{c}\, cR q_1$	$q_2 \vec{c}\, cR q_2$	$q_3 \vec{c}$ —

Таг-система моделируется следующим образом. На первом этапе имеем $\frac{1L\vec{1}}{q_1 1\vec{1}Lq_1}$, $\frac{\vec{1}L\vec{1}}{q_1\vec{1}\,\vec{1}Lq_1}$, $\frac{\overleftarrow{b}Lb}{q_1\overleftarrow{b}bLq_1}$, $\frac{Lb(\overleftarrow{b})R}{q_1 b\overleftarrow{b}Rq_1}$, $\frac{\vec{1}R\overleftarrow{1}}{q_1\vec{1}\,\overleftarrow{1}Rq_1}$, $\frac{bR\overleftarrow{b}}{q_1 b\overleftarrow{b}Rq_1}$ и $\frac{R1(\vec{1})L}{q_1 1\vec{1}Lq_1}$. Когда головка УМТ, двигаясь вправо, встречает метку c, первый этап моделирования заканчивается. УМТ удаляет эту метку и начинается второй этап (команда $q_1c\vec{1}Lq_2$).

На втором этапе моделирования УМТ пишет метки c и символы $\vec{1}$ в Q_R. Более того, метка c записывается только после символа $\vec{1}$.

Если УМТ пишет символ $\vec{1}$ в Q_R после записи метки c или после первого этапа моделирования, происходит следующее: $\frac{\vec{b}R\overleftarrow{b}}{q_2\vec{b}\,\overleftarrow{b}Rq_2}$, $\frac{\vec{1}R\overleftarrow{1}}{q_2\vec{1}\,\overleftarrow{1}Rq_2}$, $\frac{1R\overleftarrow{1}}{q_2 1\overleftarrow{1}Rq_2}$, $\frac{cR\overleftarrow{c}}{q_2\vec{c}\,\overleftarrow{c}Rq_2}$ и $\frac{R0(\vec{1})L}{q_2 0\vec{1}Lq_2}$. Если УМТ пишет символ $\vec{1}$ в Q_R после записи такого же символа $\vec{1}$, имеем $\frac{\vec{b}R\overleftarrow{b}}{q_2\vec{b}\,\overleftarrow{b}Rq_2}$, $\frac{\vec{1}R\overleftarrow{1}}{q_2\vec{1}\,\overleftarrow{1}Rq_2}$, $\frac{\vec{c}R\overleftarrow{c}}{q_2\vec{c}\,\overleftarrow{c}Rq_2}$ и $\frac{R0(\vec{1})L}{q_2 0\vec{1}Lq_2}$. Головка УМТ после записи символа $\vec{1}$ в Q_R движется влево: $\frac{\overleftarrow{1}L\overleftarrow{1}}{q_2\overleftarrow{1}\,\overleftarrow{1}Lq_2}$, $\frac{\overleftarrow{c}L\overleftarrow{c}}{q_2\overleftarrow{c}\,\overleftarrow{c}Lq_2}$, $\frac{\overleftarrow{b}L\overleftarrow{b}}{q_2\overleftarrow{b}\,\overleftarrow{b}Lq_2}$.

Затем, встретив символ 1 в P_r, головка изменит направление движения, заменит символ 1 на $\overleftarrow{1}$ и запишет в Q_R символ $\vec{1}$: $\frac{L1(\overleftarrow{1})R}{q_2 1\overleftarrow{1}Rq_2}$. Если в P_r встретятся метки bb, УМТ запишет в Q_R метку c: $\frac{Lbb(\vec{b}\,\overleftarrow{b})R}{q_2 b\vec{b}Lq_3\ q_3 b\overleftarrow{b}Rq_1\ q_1\vec{b}\,\overleftarrow{b}Rq_1}$, затем $\frac{\vec{b}R\overleftarrow{b}}{q_1\vec{b}\,\overleftarrow{b}Rq_1}$, $\frac{\vec{1}R\overleftarrow{1}}{q_1\vec{1}\,\overleftarrow{1}Rq_1}$, $\frac{\vec{c}R\overleftarrow{c}}{q_1\vec{c}\,\overleftarrow{c}Rq_1}$ и $\frac{R0(c)L}{q_1 0cLq_3}$.

Когда головка УМТ движется влево после записи символа c в Q_R, имеем $\frac{\overleftarrow{1}L1}{q_3\overleftarrow{1}1Lq_3}$, $\frac{\overleftarrow{c}Lc}{q_3\overleftarrow{c}cLq_3}$, и УМТ восстанавливает часть S на ленте. Затем УМТ встречает метку $\overleftarrow{b}$ в P: $\frac{\overleftarrow{b}L\vec{b}}{q_3\overleftarrow{b}\,\vec{b}Lq_2\ q_2\overleftarrow{b}\,\vec{b}Lq_2}$, $\frac{\overleftarrow{1}L\vec{1}}{q_2\overleftarrow{1}\,\vec{1}Lq_2}$.

УМТ останавливается, встретив пару $\vec{c}b$ (команды $q_2b\vec{b}Lq_3$ и $q_3\vec{c}$ —).

Второй этап моделирования закончен, когда УМТ встречает пару $1b$. Затем $\frac{L1b(1b)R}{q_2 b\vec{b}Lq_3\ q_3 11Rq_3\ q_3\vec{b}bRq_3}$, и начинается третий этап моделирования.

На третьем этапе УМТ восстанавливает участок P ленты (участок S восстановлен после записи метки c в Q_R): $\frac{\vec{1}R1}{q_3\vec{1}1Rq_3}$, $\frac{1R1}{q_311Rq_3}$ и $\frac{\vec{b}Rb}{q_3\vec{b}bRq_3}$.

Когда головка УМТ, двигаясь вправо, встречает метку c, третий этап (и весь цикл моделирования) закончены. УМТ удаляет метку c, и начинается новый цикл моделирования (команда q_3c1Rq_1).

6.7. УМТ с 2 состояниями и 18 символами (см. [45]). Предъявим машину из класса $\mathcal{UTM}(2, 18)$, имеющую состояния q_1 и q_2 и символы $\vec{1}$ (пустой символ), 1, $\vec{1}$, $\vec{1}_1$, $\vec{1}_1$, b, $\vec{b}$, $\vec{b}$, $\vec{b}_1$, $\vec{b}_1$, b_2, b_3, c, $\vec{c}$, $\vec{c}$, $\vec{c}_1$, $\vec{c}_1$ и c_2. Ее программа приведена в табл. 9. Пусть $N_1 = 1$, $N_{k+1} = N_k + 2m_k + 1$, $k = 1, \ldots, n$. Пусть продукция $\alpha_i = a_{i1}a_{i2}\ldots a_{im_i}$ таг-системы, $i = 1, \ldots, n$, кодируется словом P_i, где $P_i = bb1^{N_{im_i}}1b1^{N_{im_i-1}}\ldots 1b1^{N_{i2}}1b1^{N_{i1}}$, т. е. $A_j = 1^{N_j}$, $j = 1, \ldots, n+1$, а символ b — метка. Пусть, кроме того, $P_0 = bb$ и $P_{n+1} = \vec{c}_1\vec{c}_1$. Пусть начальное слово $\beta = a_r a_s a_t \ldots a_w$, преобразуемое таг-системой, имеет код S, где $S = 1^{N_r}c1^{N_s}c1^{N_t}\ldots c1^{N_w}c$, т. е. символ c — также метка.

Таблица 9

Программа УМТ с 2 состояниями и 18 символами

$q_1 1$	$c_2 L q_1$	$q_2 1$	$\vec{1}\ Rq_2$
$q_1\vec{1}$	$\vec{1}_1 Rq_1$	$q_2\vec{1}$	$\vec{1}\ Rq_2$
$q_1\vec{1}$	$c_2 Lq_1$	$q_2\vec{1}$	$\vec{1}\ Lq_2$
$q_1\vec{1}_1$	$1\ Rq_1$	$q_2\vec{1}_1$	$\vec{1}_1 Rq_2$
$q_1\vec{1}_1$	$\vec{1}_1 Lq_1$	$q_2\vec{1}_1$	$1\ Lq_2$
$q_1 b$	$\vec{b}\ Rq_1$	$q_2 b$	$b_2 Rq_1$
$q_1\vec{b}$	$\vec{b}_1 Rq_1$	$q_2\vec{b}$	$\vec{b}\ Rq_2$
$q_1\vec{b}$	$b\ Lq_1$	$q_2\vec{b}$	$\vec{b}\ Lq_2$
$q_1\vec{b}_1$	$b\ Rq_1$	$q_2\vec{b}_1$	$\vec{b}_1 Rq_2$
$q_1\vec{b}_1$	$\vec{b}_1 Lq_1$	$q_2\vec{b}_1$	$\vec{b}\ Lq_2$
$q_1 b_2$	$b_3 Lq_2$	$q_2 b_2$	$b\ Rq_1$
$q_1 b_3$	$\vec{b}_1 Lq_2$	$q_2 b_3$	$\vec{b}_1 Rq_2$
$q_1\vec{c}$	$\vec{1}\ Lq_2$	$q_2\vec{c}$	$\vec{c}\ Rq_2$
$q_1\vec{c}$	$\vec{c}\ Rq_1$	$q_2\vec{c}$	$\vec{c}\ Rq_2$
$q_1\vec{c}$	$\vec{c}_1 Lq_1$	$q_2\vec{c}$	$\vec{c}\ Lq_2$
$q_1\vec{c}_1$	$c_1 Rq_2$	$q_2\vec{c}_1$	$c_2 Rq_2$
$q_1 c_1$	—	$q_2 c_1$	$c_2 Lq_1$
$q_1 c_2$	$\vec{1}\ Rq_1$	$q_2 c_2$	$c\ Lq_2$

Таг-система моделируется следующим образом. На первом этапе имеем $\frac{1Lc_2}{q_11c_2Lq_1}$, $\frac{\vec{1}Lc_2}{q_1\vec{1}c_2Lq_1}$, $\frac{\vec{b}Lb}{q_1\vec{b}bLq_1}$, $\frac{Lb(\vec{b})R}{q_1b\vec{b}Rq_1}$, $\frac{c_2R\vec{1}}{q_1c_2\vec{1}Rq_1}$, $\frac{bR\vec{b}}{q_1b\vec{b}Rq_1}$ и $\frac{R1(c_2)L}{q_11c_2Lq_1}$.

Первый этап завершен, когда головка УМТ, двигаясь вправо, встречает метку c. Эта метка удаляется и начинается второй этап моделирования (команда $q_1c\vec{1}Lq_2$). Лента УМТ в этот момент имеет вид

$$Q_L P_{n+1}P_n\ldots P_{r+1}P_rP'_{r-1}\ldots P'_1P'_0R'A_t\ldots A_wQ_R,$$

где в P'_i, $i = 0, \ldots, r-1$, символы 1 заменены на $\vec{1}$, а метки b — на $\vec{b}$. Участок R' содержит $\vec{1}$ и $\vec{1}$, головка УМТ находится в R', текущее состояние — q_2. На втором этапе моделирования УМТ пишет метки c_2 и символы $\vec{1}$ в Q_R, причем метки c_2 записываются только после символа $\vec{1}$.

Если головка УМТ, двигаясь влево в коде P_r, встречает символ 1, то направление движения меняется на противоположное, символ 1 заменяется на $\vec{1}$, а в Q_R записывается символ $\vec{1}$: $\frac{L1(\vec{1})R}{q_21\vec{1}Rq_2}$.

Если УМТ пишет символ $\vec{1}$ в Q_R после записи метки c_2 или после первого этапа моделирования, имеем $\frac{\vec{b}R\vec{b}}{q_2\vec{b}\vec{b}Rq_2}$, $\frac{\vec{1}R\vec{1}}{q_2\vec{1}\vec{1}Rq_2}$, $\frac{1R\vec{1}}{q_21\vec{1}Rq_2}$, $\frac{cR\vec{c}}{q_2c\vec{c}Rq_2}$ и $\frac{R\vec{1}(\vec{1})L}{q_2\vec{1}\vec{1}Lq_2}$. Если же УМТ пишет в Q_R символ $\vec{1}$ после записи такого же символа $\vec{1}$, выполняется следующее: $\frac{\vec{b}R\vec{b}}{q_2\vec{b}\vec{b}Rq_2}$, $\frac{\vec{1}R\vec{1}}{q_2\vec{1}\vec{1}Rq_2}$, $\frac{\vec{c}R\vec{c}}{q_2\vec{c}\vec{c}Rq_2}$ и $\frac{R\vec{1}(\vec{1})L}{q_2\vec{1}\vec{1}Lq_2}$.

Когда головка УМТ движется влево после записи символа $\vec{1}$ в Q_R, имеем $\frac{\vec{1}L\vec{1}}{q_2\vec{1}\vec{1}Lq_2}$, $\frac{\vec{c}L\vec{c}}{q_2\vec{c}\vec{c}Lq_2}$, $\frac{\vec{b}L\vec{b}}{q_2\vec{b}\vec{b}Lq_2}$.

Если головка УМТ, двигаясь влево в коде P_r после записи символа $\vec{1}$ в Q_R, встречает метку b, то в Q_R записывается метка c_2: $\frac{Lb(b_2)R}{q_2bb_2Rq_1}$, $\frac{\vec{b}R\vec{b}_1}{q_1\vec{b}\vec{b}_1Rq_1}$, $\frac{\vec{1}R\vec{1}_1}{q_1\vec{1}\vec{1}_1Rq_1}$, $\frac{\vec{c}R\vec{c}}{q_1\vec{c}\vec{c}Rq_1}$ и $\frac{R\vec{1}(c_2)L}{q_1\vec{1}c_2Lq_1}$.

Когда головка УМТ движется влево после записи символа c_2 в Q_R, имеем $\frac{\overleftarrow{1}_1 L \overrightarrow{1}_1}{q_1 \overleftarrow{1}_1 \overrightarrow{1}_1 L q_1}$, $\frac{\overleftarrow{c} L \overrightarrow{c_1}}{q_1 \overleftarrow{c c_1} L q_1}$, и $\frac{\overleftarrow{b}_1 L \overrightarrow{b}_1}{q_1 \overleftarrow{b}_1 \overrightarrow{b}_1 L q_1}$.

УМТ останавливается на втором этапе, достигнув символа $\overleftarrow{c}_1$ в состоянии q_1 (команды $q_2 \overleftarrow{c}_1 c_2 L q_1$ и $q_1 \overleftarrow{c}_1$ —).

Теперь возможны два варианта: головка УМТ, двигаясь влево, встречает пару $1b_2$ или bb_2, соответственно.

В первом случае имеем $\frac{L 1 b_2 (\overleftarrow{1}\, \overleftarrow{b}_1) R}{q_1 b_2 b_3 L q_2 \;\; q_2 1 \overleftarrow{1} R q_2 \;\; q_2 b_3 \overleftarrow{b}_1 R q_2}$; УМТ восстанавливает на ленте участок S, продолжая второй этап моделирования: $\frac{\overrightarrow{1}_1 R \overleftarrow{1}_1}{q_2 \overrightarrow{1}_1 \overleftarrow{1}_1 R q_2}$, $\frac{\overrightarrow{b}_1 R \overleftarrow{b}_1}{q_2 \overrightarrow{b}_1 \overleftarrow{b}_1 R q_2}$, $\frac{\overrightarrow{c}_1 R c_2}{q_2 \overrightarrow{c}_1 c_2 R q_2}$, $\frac{R c_2 (c) L}{q_2 c_2 c L q_2}$, $\frac{\overleftarrow{1}_1 L 1}{q_2 \overleftarrow{1}_1 1 L q_2}$, $\frac{c_2 L c}{q_2 c_2 c L q_2}$, $\frac{\overleftarrow{b}_1 L \overrightarrow{b}}{q_2 \overleftarrow{b}_1 \overrightarrow{b} L q_2}$. Теперь УМТ может записывать символ $\overrightarrow{1}$ в Q_R.

Во втором случае имеем $\frac{L b b_2 (b b) R}{q_1 b_2 b_3 L q_2 \;\; q_2 b b_2 R q_1 \;\; q_1 b_3 \overrightarrow{b}_1 L q_2 \;\; q_2 b_2 b R q_1 \;\; q_1 \overrightarrow{b}_1 b R q_1}$, и начинается третий этап моделирования. На третьем этапе восстанавливается участок P ленты: $\frac{\overrightarrow{1}_1 R 1}{q_1 \overrightarrow{1}_1 1 R q_1}$, $\frac{\overrightarrow{b}_1 R b}{q_1 \overrightarrow{b}_1 b R q_1}$.

Когда головка УМТ, двигаясь вправо, встречает метку $\overrightarrow{c}_1$, эта метка меняется на $\overleftarrow{c}_1$ (команда $q_1 \overrightarrow{c}_1 \overleftarrow{c}_1 R q_2$); затем УМТ восстанавливает участок S ленты: $\frac{\overrightarrow{1}_1 R \overleftarrow{1}_1}{q_2 \overrightarrow{1}_1 \overleftarrow{1}_1 R q_2}$, $\frac{\overrightarrow{c}_1 R c_2}{q_2 \overrightarrow{c}_1 c_2 R q_2}$, $\frac{R c_2 (c) L}{q_2 c_2 c L q_2}$, $\frac{\overleftarrow{1}_1 L 1}{q_2 \overleftarrow{1}_1 1 L q_2}$ и $\frac{c_2 L c}{q_2 c_2 c L q_2}$.

Наконец, когда головка УМТ, двигаясь влево, встречает метку $\overleftarrow{c}_1$, третий этап (и весь цикл моделирования) завершаются. УМТ удаляет метку $\overleftarrow{c}_1$ и начинается новый цикл моделирования (команда $q_2 \overleftarrow{c}_1 c_2 L q_1$).

§ 7. Пример Н-системы

Грамматика $G = (N, T, R, S)$ называется *праволинейной*, если все ее правила (элементы множества R) имеют вид $x \to ay$, $x \to a$ или $x \to \varepsilon$, где $x, y \in N$ и $a \in T$. Таким образом, любое вычисление имеет вид $S \Rightarrow a_1 x_1 \Rightarrow$ $\Rightarrow a_1 a_2 x_2 \Rightarrow \ldots \Rightarrow a_1 a_2 \ldots a_n x_n \Rightarrow a_1 a_2 \ldots a_{n+1}$, где $a_i \in T$, $x_i \in N$.

Язык L называется *регулярным*, если существует порождающая его праволинейная грамматика G, т. е. если $L = L(G)$. Очевидно, что регулярные языки порождаются вычислениями специального вида: подстановки выполняются не внутри слова, а только для правого его конца. Но такая «подстановка для правого конца» легко выражается операцией рекомбинации: $wx \Rightarrow_G way$ есть результат разрезания слова wx и специального слова Zay перед символами x и a, соответственно, и рекомбинирования их в way и Zx. Таким образом, праволинейное правило $x \to ay$ становится правилом рекомбинации $\frac{\varepsilon | x}{Z | ay}$.

Следующая праволинейная грамматика G_0 порождает все слова в алфавите $\{a, b\}$, имеющие четное число вхождений буквы a и кратное 3 число вхождений буквы b. Пусть $G_0 = (N_0, T_0, R_0, S)$, где $N_0 := \{S, x_{0,0}, x_{0,1}, x_{0,2}, x_{1,0}, x_{1,1}, x_{1,2}\}$, $T_0 := \{a, b\}$, $R_0 := \{S \to \varepsilon, x_{0,0} \to a x_{1,0}, x_{0,2} \to a x_{1,2}, x_{1,1} \to a x_{0,1}, S \to a x_{1,0}, x_{0,0} \to b x_{0,1}, x_{0,2} \to b x_{0,0}, x_{1,1} \to b x_{1,2}, S \to b x_{0,1}, x_{0,1} \to a x_{1,1}, x_{1,0} \to a x_{0,0}, x_{1,2} \to a x_{0,2}, x_{0,0} \to \varepsilon, x_{0,1} \to b x_{0,2}, x_{1,0} \to b x_{1,1}, x_{1,2} \to b x_{1,0}\}$.

Пример вычислений в G_0: $S \Rightarrow a x_{1,0} \Rightarrow ab x_{1,1} \Rightarrow abb x_{1,2} \Rightarrow abba x_{0,2} \Rightarrow$ $\Rightarrow abbab x_{0,0} \Rightarrow abbab$.

Очевидно, эти действия нетрудно промоделировать посредством операций рекомбинации. Рассмотрим Н-систему $H = (\Sigma, M, E)$, где

$\Sigma := N \cup T \cup \{Z\}$ (Z — новая буква, не принадлежащая $N \cup T$); $M := \{S, ZZ\} \cup \{Zax_{i,j}\} \cup \{Zbx_{i,j}\}$, $i = 0, 1$, $j = 0, 1, 2$; E есть следующий список правил рекомбинации:

1) $\dfrac{\varepsilon \mid S}{ZZ \mid \varepsilon}$, 5) $\dfrac{\varepsilon \mid x_{0,0}}{Z \mid ax_{1,0}}$, 9) $\dfrac{\varepsilon \mid x_{0,2}}{Z \mid ax_{1,2}}$, 13) $\dfrac{\varepsilon \mid x_{1,1}}{Z \mid ax_{0,1}}$,

2) $\dfrac{\varepsilon \mid S}{Z \mid ax_{1,0}}$, 6) $\dfrac{\varepsilon \mid x_{0,0}}{Z \mid bx_{0,1}}$, 10) $\dfrac{\varepsilon \mid x_{0,2}}{Z \mid bx_{0,0}}$, 14) $\dfrac{\varepsilon \mid x_{1,1}}{Z \mid bx_{1,2}}$,

3) $\dfrac{\varepsilon \mid S}{Z \mid bx_{0,1}}$, 7) $\dfrac{\varepsilon \mid x_{0,1}}{Z \mid ax_{1,1}}$, 11) $\dfrac{\varepsilon \mid x_{1,0}}{Z \mid ax_{0,0}}$, 15) $\dfrac{\varepsilon \mid x_{1,2}}{Z \mid ax_{0,2}}$,

4) $\dfrac{\varepsilon \mid x_{0,0}}{ZZ \mid \varepsilon}$, 8) $\dfrac{\varepsilon \mid x_{0,1}}{Z \mid bx_{0,2}}$, 12) $\dfrac{\varepsilon \mid x_{1,0}}{Z \mid bx_{1,1}}$, 16) $\dfrac{\varepsilon \mid x_{1,2}}{Z \mid bx_{1,0}}$.

Теперь легко моделируются любые вычисления в G_0. Рассмотрим приведенный выше пример. Как моделировать $S \Rightarrow ax_{1,0}$? В H мы имеем молекулы S и $Zax_{1,0}$ и можем применить правило 2), чтобы получить $\{S, Zax_{1,0}\} \vdash_2 \vdash_2 \{ZS, ax_{1,0}\}$, где $ax_{1,0}$ есть желаемая молекула плюс «мусор» ZS. Мы можем продолжить и применить правило 12) к молекулам $ax_{1,0}$ и $Zbx_{1,1}$, получая $\{ax_{1,0}, Zbx_{1,1}\} \vdash_{12} \{Zx_{1,0}, abx_{1,1}\}$, т. е. $abx_{1,1}$ плюс «мусор» $Zx_{1,0}$. Легко проверить, что $S \vdash_2 ax_{1,0} \vdash_{12} abx_{1,1} \vdash_{14} abbx_{1,2} \vdash_{15} abbax_{0,2} \vdash_{10} abbabx_{0,0} \vdash_4 abbab$ есть возможная цепь реакций в H. Роль символа Z состоит в работе с «мусором», т. е. с некоторыми нежелаемыми результатами.

Предположим, что мы работаем без символа Z. Тогда правило 10) должно иметь форму

10′) $\dfrac{\varepsilon \mid x_{0,2}}{bx_{0,0}}$.

Мы могли бы применить правило 10′) к $abbabx_{0,0}$ и к «мусору» $x_{0,2}$ (вместо $Zx_{0,2}$). В результате получим «обратное» вычисление $\{abbabx_{0,0}, x_{0,2}\} \vdash_{10'} \{abbax_{0,2}, bx_{0,0}\}$. Эти «обратные» вычисления безвредны для G_0, так как G_0 имеет так называемое свойство реверсивности («backward deterministic»). Тем не менее, обычно обратные вычисления «ведут к бедствиям» и поэтому мы от них избавляемся с помощью специального символа Z. (Отметим, что К. Беннетт [11] показал, как имитировать детерминированную машину Тьюринга другой машиной, которая имеет свойство реверсивности.)

Используя примененную в этом примере технику, легко показать, что каждый регулярный язык есть результат расширенной 1-tt-системы; больше сведений об этом можно найти в [37].

§ 8. 3-tt-системы имитируют любую грамматику

В отличие от праволинейных грамматик, в общем случае правило $u \to v$ формальной грамматики определяет подстановку внутри слова: $w_1 u w_2 \Rightarrow w_1 v w_2$. В то время как правила рекомбинации тривиально моделируют правила праволинейной грамматики, они не могут непосредственно моделировать общую подстановку. В [12, 17, 50] введен *метод ротации слов*, превращающий $w_1 u w_2$ в $w_2 w_1 u$; подстановки применяются исключительно к концу слова, как и в случае праволинейных грамматик. Дополнительная буква B отмечает правильное начало слова. Так, $w_2 B w_1 u$ есть ротационная версия слова $B w_1 u w_2$. Для «технических» целей вводятся еще две дополнительные буквы, X и Y, отмечающие начало и конец ротационных слов. *Представителем* слова w в алфавите $N \cup T$ является любое слово вида $X w_2 B w_1 Y$, где $X, B, Y \notin N \cup T$ и $w = w_1 w_2$.

Для любой грамматики G, $G=(N, T, R, S)$, мы построим такую 3-tt-систему, чтобы для любого слова w из $(N\cup T)^*$, где $S\Rightarrow_G^* w$, в первой пробирке нашлись все его представители Xw_2Bw_1Y. Здесь мы следуем идеям работ [12, 17]. Правило $u\to v$ из R, применяемое к w_1uw_2, моделируется правилом рекомбинации $\frac{\varepsilon|uY}{Z|vY}$, применяемым к представителю Xw_2Bw_1uY слова w_1uw_2. Мы хотим быть уверены в том, что все представители слова w, порожденного из S в G, будут находиться в первой пробирке. Для этого надо использовать ротацию слов между X и Y, которая выполняется с помощью дополнительных пробирок 2 и 3 и кодирования букв алфавита $N\cup T\cup\{B\}$. Пусть $N\cup T\cup\{B\}=\{l_1, \ldots, l_n\}$. Мы кодируем l_i словом $\beta\alpha^i\beta$, где $\alpha^i=\underbrace{\alpha\ldots\alpha}_{i}$, α и β — новые буквы. Правило рекомбинации $\frac{\varepsilon|l_iY}{Z|\beta\alpha^i\beta Y}$ кодирует последнюю букву l_i перед Y в слове $\beta\alpha^i\beta$. Если эта последняя буква есть β (или α), мы удаляем ее и меняем Y на Y_β (или Y_α). Никаких дальнейших реакций с буквами Y_α и Y_β в первой пробирке не происходит.

Слова с буквами Y_β (или Y_α) могут проходить только фильтр пробирки 2 (или 3), где новые буквы β (или α) добавляются сразу после X. Когда полное преобразование кода $\beta\alpha^i\beta$ выполнено, дальнейшее правило рекомбинации декодирует $\beta\alpha^i\beta$ обратно в l_i: $\frac{X\beta\alpha^i\beta|\varepsilon}{Xl_i|Z}$.

Итак, с каждой формальной грамматикой G, $G=(N, T, R, S)$, мы ассоциируем 3-tt-систему H_G. Обозначим элементы множества $N\cup T\cup\{B\}$ через $l_1, \ldots, l_n$. Тогда $H_G=(\Sigma, T_1, T_2, T_3)$, где $\Sigma=N\cup T\cup\{B, X, Y, \alpha, \beta, X', Y_\alpha, Y_\beta\}$, $T_i=(M_i, E_i, F_i)$, $i=1, 2, 3$; далее,

$F_1=N\cup T\cup\{B, X, Y, \alpha, \beta\}$; $F_2=N\cup T\cup\{B, \alpha, \beta, X', Y_\beta\}$; $F_3=N\cup T\cup\{B, \alpha, \beta, X', Y_\alpha\}$;

$M_1=\{XSBY, XBSY, ZY_\alpha, ZY_\beta, X'Z\}\cup\{Z\beta\alpha^i\beta Y: i=1, \ldots, n\}\cup\{ZvY: (\exists u)((u\to v)\in R)\}\cup\{Xl_iZ: i=1, \ldots, n\}$, $M_2=\{ZY, X\beta Z\}$, $M_3=\{ZY, X\alpha Z\}$;

множества E_1, E_2 и E_3 состоят, соответственно, из указанных ниже правил рекомбинации 1)–6), 7) и 8), 9) и 10):

1) $\frac{\varepsilon|uY}{Z|vY}$, $(u\to v)\in R$;

2) $\frac{\varepsilon|l_iY}{Z|\beta\alpha^i\beta Y}$, $i=1, \ldots, n$;

3) $\frac{\varepsilon|\beta Y}{Z|Y_\beta}$;

4) $\frac{\varepsilon|\alpha Y}{Z|Y_\alpha}$;

5) $\frac{X|\varepsilon}{X'|Z}$;

6) $\frac{X\beta\alpha^i\beta|\varepsilon}{Xl_i|Z}$, $i=1, \ldots, n$;

7) $\frac{\varepsilon|Y_\beta}{Z|Y}$;

8) $\frac{X'|\varepsilon}{X\beta|Z}$;

9) $\frac{\varepsilon|Y_\alpha}{Z|Y}$;

10) $\frac{X'|\varepsilon}{X\alpha|Z}$.

Предположим, что $S\Rightarrow_G^* w_1uw_2\Rightarrow_G w_1vw_2$ верно при $(u\to v)\in R$. В пробирке 1 мы имеем $XBSY$ как молекулу в M_1. Пусть в пробирке 1 уже сгенерированы все представители слова w_1uw_2. Тогда в пробирке 1 также содержится Xw_2Bw_1uY, и возможна следующая цепь реакций:

Пробирка 1: $\{Xw_2Bw_1uY, ZvY\}\vdash_1\{Xw_2Bw_1vY, ZuY\}$. Пусть $v=v'l_i$ для некоторых v' и l_i; тогда $\{Xw_2Bw_1v'l_iY, Z\beta\alpha^i\beta Y\}\vdash_2\vdash_2\{Xw_2Bw_1v'\beta\alpha^i\beta Y, Zl_iY\}$, $\{Xw_2Bw_1v'\beta\alpha^i\beta Y, ZY_\beta\}\vdash_3\{Xw_2Bw_1v'\beta\alpha^iY_\beta, Z\beta Y\}$, $\{Xw_2Bw_1v'\beta\alpha^iY_\beta, X'Z\}\vdash_5\{X'w_2Bw_1v'\beta\alpha^iY_\beta, XZ\}$.

Пробирка 2: $\{X'w_2Bw_1v'\beta\alpha^iY_\beta,\ ZY\}\vdash_7\{X'w_2Bw_1v'\beta\alpha^iY,\ ZY_\beta\}$, $\{X'w_2Bw_1v'\beta\alpha^iY,\ X\beta Z\}\vdash_8\{X\beta w_2Bw_1v'\beta\alpha^iY,\ X'Z\}$.

Пробирка 1: $\{X\beta w_2Bw_1v'\beta\alpha^{i-1}\alpha Y,\ ZY_\alpha\}\vdash_4\{X\beta w_2Bw_1v'\beta\alpha^{i-1}Y_\alpha,\ Z\alpha Y\}$, $\{X\beta w_2Bw_1v'\beta\alpha^{i-1}Y_\alpha,\ X'Z\}\vdash_5\{X'\beta w_2Bw_1v'\beta\alpha^{i-1}Y_\alpha,\ XZ\}$.

Пробирка 3: $\{X'\beta w_2Bw_1v'\beta\alpha^{i-1}Y_\alpha,\ ZY\}\vdash_9\{X'\beta w_2Bw_1v'\beta\alpha^{i-1}Y,\ ZY_\alpha\}$, $\{X'\beta w_2Bw_1v'\beta\alpha^{i-1}Y,\ X\alpha Z\}\vdash_{10}\{X\alpha\beta w_2Bw_1v'\beta\alpha^{i-1}Y,\ X'Z\}$.

Продолжая такие реакции, мы в конце концов получим в пробирке 1 слово $X\beta\alpha^i\beta w_2Bw_1v'Y$. Далее, $\{X\beta\alpha^i\beta w_2Bw_1v'Y,\ Xl_iZ\}\vdash_6\{Xl_iw_2Bw_1v'Y,\ X\beta\alpha^i\beta Z\}$; здесь последняя буква l_i перед Y в результате ротации стала первой буквой после X.

Используя эту технику, в пробирке 1 легко получить любой представитель слова w_1vw_2.

Результаты работы всех трех пробирок можно также описать следующим образом. Определим отображение c: $\{l_1,\ldots,l_n\}^*\to 2^{\{l_1,\ldots,l_n,\alpha,\beta\}^*}$ (где 2^M обозначает множество всех подмножеств множества M), положив $c(l_i):=\{l_i,\beta\alpha^i\beta\}$ и $c(w_1w_2):=c(w_1)c(w_2)$. В частности, $l_2\beta\alpha\alpha\alpha\beta l_1\beta\alpha\beta\in c(l_2l_3l_1l_1)$. Далее, отображение ρ: $\{l_1,\ldots,l_n,\alpha,\beta\}^*\to 2^{\{l_1,\ldots,l_n,\alpha,\beta\}^*}$ определим так, чтобы соотношение $w_2w_1\in\rho(w)$ было выполнено в том и только том случае, когда $w=w_1w_2$ для некоторых w_1 и w_2. Положим $\tilde{c}:=\rho\circ c$. В частности, $\alpha\beta l_1\beta\alpha\beta l_2\beta\alpha\alpha\in\tilde{c}(l_2l_3l_1l_1)$. Определим

$$C_1:=\{\mathscr{A}u\Omega\colon(\mathscr{A}\in\{X,X'\})\&(\Omega=Y)\&(\exists w)\ (u\in\tilde{c}(Bw)\&S\Rightarrow^*w)\vee$$
$$\vee(\mathscr{A}=X)\&(\Omega=Y_\alpha)\&(\exists w)\ (u\alpha\in\tilde{c}(Bw)\&S\Rightarrow^*w)\vee$$
$$\vee(\mathscr{A}=X)\&(\Omega=Y_\beta)\&(\exists w)\ (u\beta\in\tilde{c}(Bw)\&S\Rightarrow^*w)\},$$
$$C_2:=\{\mathscr{A}u\Omega\colon[(\mathscr{A}\in\{X',X\beta\})\&(\Omega=Y_\beta)\vee(A=X')\&(\Omega=Y)]\&$$
$$\&(\exists w)\ (u\beta\in\tilde{c}(Bw)\&S\Rightarrow^*w)\},$$
$$C_3:=\{\mathscr{A}u\Omega\colon[(\mathscr{A}\in\{X',X\alpha\})\&(\Omega=Y_\alpha)\vee(A=X')\&(\Omega=Y)]\&$$
$$\&(\exists w)\ (u\alpha\in\tilde{c}(Bw)\&S\Rightarrow^*w)\};$$
$$G_1:=\{ZuY\colon(\exists v)\ ((u\to v)\in R\vee(v\to u)\in R)\}\cup$$
$$\cup\{Z\beta\alpha^i\beta Y,\ X\beta\alpha^i\beta Z,\ Zl_iY,\ Xl_iZ,\ i=1,\ldots,n\}\cup$$
$$\cup\{Z\beta Y,\ ZY_\beta,\ Z\alpha Y,\ ZY_\alpha XZ,\ X'Z\},$$
$$G_2:=\{ZY,\ ZY_\beta,\ X\beta Z,\ X'Z\},\qquad G_3:=\{ZY,\ ZY_\alpha,\ X\alpha Z,\ X'Z\}.$$

Теперь легко доказать, что результат ρ_i в i-й пробирке, $i=1,2,3$, есть в точности $C_i\cup G_i$. Здесь G_i обозначает «мусор», а C_i — желаемое содержимое. Для доказательства соотношения $\rho_i\supseteq C_i\cup G_i$ применима описанная выше процедура: для любого слова из $C_i\cup G_i$ индуктивно находим цепь реакций, производящих его. Для получения обратного соотношения ($\subseteq$) просто отметим, что применение правила рекомбинации пробирки i к двум словам из $C_i\cup G_i$ имеет результатом вновь два слова из $C_i\cup G_i$. Итак, если рассматривать только «некодированные» слова (т. е. $c(l_i)=l_i$), то, как уже показано, имеет место следующая

Теорема 4 [41]. *3-tt-система H_G может произвести в пробирке 1 слово Xw_2Bw_1Y, где w_1 и w_2 — два слова в алфавите $N\cup T$, в том и только том случае, когда имеет место соотношение $S\Rightarrow_G^*w_1w_2$.*

Будем говорить, что класс n-tt-систем $\mathscr{C}$ является *n-tt-предсказуемым*, если существует алгоритм $\mathscr{A}$, позволяющий по любой системе H из класса $\mathscr{C}$ и любому слову w в алфавите системы H определить, будет ли оно порождено в первой пробирке этой системы.

Предположим теперь, что класс всех 3-tt-систем предсказуем. Тогда мы можем решить проблему вхождения для любой грамматики G следующим

образом. Пусть $G=(N, T, R, S)$ и $w \in T^*$. Рассмотрим систему H_G и слово $XBwY$. С помощью алгоритма $\mathcal{A}$ проверим, производит ли система H_G в пробирке 1 слово $XBwY$. В этом случае соотношение $S \Rightarrow_G^* w$ истинно, в противном — ложно. Но проблема вхождения для грамматик общего вида неразрешима, а потому имеет место

Следствие 3. *Конечная 3-tt-система непредсказуема.*

§ 9. Расширенная 3-tt-система может порождать любой «чистый» формальный язык

Расширенная n-tt-система $\mathcal{H}=(H, \Gamma)$ порождает данный язык L, если выполнено соотношение $L=\rho(H) \cap \Gamma^*$. Пусть G — формальная грамматика, $L=L(G)$; тогда для 3-tt-системы H_G из § 8 имеем $\rho(H_G) \cap \Gamma^* \neq L$. Мы «всего лишь» доказали, что для слов w в алфавите T соотношение $XBwY \in \rho(H_G)$ выполнено в том и только том случае, когда $w \in L(G)$. Для порождения «чистых» слов w из $L(G)$ в пробирке 1 можно применять расширенную 3-tt-систему $\mathcal{H}_G$, используя T в качестве терминального алфавита (т. е. $\mathcal{H}_G=(H_G, T)$); мы будем избавляться от символов X, B и Y. Стандартная идея из теории формальных языков состоит в том, что мы порождаем слова $XBwY$, где $w \in T^*$ или $w \in \Sigma^*$, и просто предполагаем, что $w \in T^*$ может быть верно. Если теперь удалить XB и Y, результатом может быть чистое терминальное слово (но все терминальные слова будут получены таким образом). Исходя из вышесказанного, можно ввести два новых правила рекомбинации для пробирки 1:

$$1') \ \frac{XB \mid \varepsilon}{\varepsilon \mid ZZ}, \qquad 2') \ \frac{\varepsilon \mid Y}{ZZ \mid \varepsilon}.$$

Однако, как будет видно из дальнейшего, это приведет к «хаосу». Предположим, получено слово $XBwY$, где $w \in T^*$, так что $w \in L$. Применяя к нему правило 1'), получаем wY. Теперь можно использовать правила всех трех пробирок для того, чтобы удалить последнюю букву слова w перед Y, не воспроизводя ее (в ротационной форме) сразу за X. Таким образом, мы можем получить любое слово $w'Y$, где $w=w''w'$ для некоторого w'. Согласно правилу 2') можно получить и w', хотя этот суффикс слова w не принадлежит $L(G)$.

Тем не менее, эту ситуацию можно исправить. Для этого предлагается метод, где букву Y можно удалить только после того, как она «сообщает X, что удаляется, и рекомендует сделать то же самое».

Когда мы предполагаем удалить XB и Y, мы сначала преобразуем Y в слово $\gamma\beta\beta Y$, где γ — новая буква; теперь ротация сдвигает $\beta\beta$ к началу слова, как было сделано выше. Таким образом, получаем $XBwY \vdash XBw\gamma\beta\beta Y \vdash^* X'\beta\beta Bw\gamma Y$. Буква γ — новая, поэтому правил для γY в H_G нет. Мы просто удаляем γY. Тем не менее, слова́ $X'\beta\beta Bw$ теперь могут войти в пробирки 2 и 3, и новые буквы α и β могут быть произведены после X. К счастью, это приведет только к появлению «мусора» не из T^*, так как мы удаляем X и B только вместе, в форме $X\beta\beta B$, с помощью правила $\frac{X\beta\beta B \mid \varepsilon}{\varepsilon \mid ZZ}$. Заметим, что слово $\beta\beta$ не кодирует никаких букв l_i, только слова́ $\beta\alpha^i\beta$ при $i>0$ кодирует буквы. Таким образом, $X\beta\beta B$ есть уникальное сообщение для удаления X.

Пусть G, $G=(N, T, R, S)$, — произвольная формальная грамматика. Обозначим элементы множества $N \cup T \cup \{B\}$ через $l_1, \ldots, l_n$. Пусть $\mathcal{H}_G$ — расширенная 3-tt-система, определяемая следующим образом:

$\mathscr{H}_G = (\Sigma, T_1', T_2', T_3', \Gamma)$, где $\Gamma = T$, $\Sigma = N \cup T \cup \{B, X, X', Y, \alpha, \beta, \gamma, Y_\alpha, Y_\beta\}$, $T_i' = (M_i', E_i', F_i')$, $i = 1, 2, 3$; далее,

$F_1' = N \cup T \cup \{B, X, Y, \alpha, \beta, \gamma\}$, $F_2' = V \cup T \cup \{B, \alpha, \beta, \gamma, X', Y_\beta\}$, $F_3' = V \cup T \cup \{B, \alpha, \beta, \gamma, X', Y_\beta\}$;

$M_1' = \{XSBY, XBSY, ZY_\alpha, ZY_\beta, X'Z, ZZ, Z\beta\alpha^1\beta Y, \ldots, Z\beta\alpha^n\beta Y, Xl_1Z, \ldots, Xl_nZ\} \cup \{ZvY: (\exists u)((u \to v) \in R)\}$, $M_2' = \{ZY, X\beta Z\}$, $M_3' = \{ZY, X\alpha Z\}$;

множества E_1', E_2' и E_3' состоят, соответственно, из указанных ниже правил рекомбинации 1)–9), 10) и 11), 12) и 13):

1) $\dfrac{\varepsilon \mid uY}{Z \mid vY}$, $(u \to v) \in R$;

2) $\dfrac{\varepsilon \mid l_i Y}{Z \mid \beta\alpha^i\beta Y}$, $i = 1, \ldots, n$;

3) $\dfrac{\varepsilon \mid \beta Y}{Z \mid Y_\beta}$;

4) $\dfrac{\varepsilon \mid \alpha Y}{Z \mid Y_\alpha}$;

5) $\dfrac{X \mid \varepsilon}{X' \mid Z}$;

6) $\dfrac{X\beta\alpha^i\beta \mid \varepsilon}{Xl_i \mid Z}$, $i = 1, \ldots, n$;

7) $\dfrac{\varepsilon \mid Y}{Z \mid \gamma\beta\beta Y}$;

8) $\dfrac{\varepsilon \mid \gamma Y}{ZZ \mid \varepsilon}$;

9) $\dfrac{X\beta\beta B \mid \varepsilon}{\varepsilon \mid ZZ}$;

10) $\dfrac{\varepsilon \mid Y_\beta}{Z \mid Y}$;

11) $\dfrac{X' \mid \varepsilon}{X\beta \mid Z}$;

12) $\dfrac{\varepsilon \mid Y_\alpha}{Z \mid Y}$;

13) $\dfrac{X' \mid \varepsilon}{X\alpha \mid Z}$.

Теперь, используя доказательство теоремы 4 со слегка модифицированными множествами C_i и G_i, $i = 1, 2, 3$, для слов w в алфавите T легко показать, что $w \in L(G)$ в том и только том случае, когда $XBwY \in \rho(H_G)$, а это равносильно соотношению $w \in \rho(\mathscr{H}_G)$.

Таким образом, $L(G) = \rho(\mathscr{H}_G)$, и доказана следующая

Теорема 5 [41]. *Любой рекурсивно перечислимый язык порождается расширенной 3-tt-системой.*

Расширенная n-tt-система $(\Sigma, T_1, \ldots, T_n, \Gamma)$ обладает терминальным алфавитом Γ. Его можно опустить в $\mathscr{H}_G$, если использовать дополнительную пробирку T_0 вида $(\varnothing, \varnothing, T)$, которая фильтрует все слова над терминальным алфавитом T. Отсюда получаем тривиальное

Следствие 4. *Любой рекурсивно перечислимый язык можно породить 4-tt-системой.*

Итак, мы представили достаточно простое доказательство того, что любой рекурсивно перечислимый язык порождается конечной 4-tt-системой или конечной расширенной 3-tt-системой. Более того, не существует алгоритма, предсказывающего результат работы конечной 3-tt-системы. Для конечных 2-tt-систем этот вопрос остается открытым.

§ 10. Универсальная расширенная 3-tt-система

В этом параграфе предлагается другой способ доказательства универсальности расширенных 3-tt-систем, обладающий по сравнению с предыдущим определенными преимуществами, хотя и более сложный. Мы покажем, как с помощью расширенных 3-tt-систем моделировать произвольную таг-систему с $m = 2$. Но такие таг-системы эквивалентны машинам Тьюринга, а потому существует универсальная расширенная 3-tt-система [30].

Лемма 6. *Для любой таг-системы T существует моделирующая ее расширенная 3-tt-система T'.*

Доказательство. Возьмем произвольную таг-систему T:

$$\begin{cases} a_i \to P_i, & i = 1, \ldots, n; \\ a_{n+1} \to \text{STOP}. \end{cases}$$

Опишем построение расширенной 3-tt-системы T'. Пусть $\Omega = \{a_1, \ldots, a_{n+1}\}$; символ a_{n+1} иногда будем обозначать через h. Пусть система T' состоит из пробирок A, B и C.

Пусть $\Sigma = \Omega \cup \{X, Y, Y', Y'', Z, Z', A, A', B, B', B'', B''', C, \alpha, \beta, \beta', \beta''\}$; пусть терминальный алфавит Г совпадает с множеством Ω.

Пусть система T стартует со слова w в алфавите Ω. Тогда к аксиомам первой пробирки 3-tt-системы T' добавляется аксиома XwY. Если таг-система T заканчивает вычисления на слове v из Ω^*, то это слово — и никакое другое слово в терминальном алфавите Ω — оказывается в пробирке A как результат. Таким образом, 3-tt-система T' моделирует работу таг-системы T.

Полностью пробирки системы T' описаны в табл. 10. (В аксиомах и правилах имеется в виду, что $i = 1, \ldots, n$, $j = 1, \ldots, n+1$.)

Таблица 10

	Пробирка A	Пробирка B	Пробирка C
Аксиомы	$X\beta\alpha^i\beta' Z$, ZP_iY'', ZY', $X\beta\alpha^{n+1}\beta'' Z$, CZ, BZ, $B'Z$, $B''Z$, $B'''Z$, ZZ, ZZ', XwY	$Z\beta Y$, XZ, ZZ, ZY, AZ, $A'Z$	$Z\alpha Y$, XZ, ZZ
Фильтр	$\Gamma \cup \{X, Y, A, A'\}$	$\Gamma \cup \{B, B', B'', B''', Y', Y''\}$	$\Gamma \cup \{C, Y'\}$
Правила	1) $\dfrac{Xa_ia_j \mid \varepsilon}{X\beta\alpha^i\beta' \mid Z}$ 2) $\dfrac{X\beta \mid \varepsilon}{B \mid Z}$ 3) $\dfrac{\varepsilon \mid Y}{Z \mid Z'}$ 4) $\dfrac{X\alpha \mid \varepsilon}{C \mid Z}$ 5) $\dfrac{X\beta' \mid \varepsilon}{B' \mid Z}$ 6) $\dfrac{X\beta'' \mid \varepsilon}{B''' \mid Z}$ 7) $\dfrac{\varepsilon \mid \beta\alpha^i\beta Y}{Z \mid P_iY''}$ 8) $\dfrac{A \mid \varepsilon}{B'' \mid Z}$ 9) $\dfrac{Xh \mid \varepsilon}{X\beta\alpha^{n+1}\beta'' \mid Z}$ 10) $\dfrac{\varepsilon \mid \beta\alpha^{n+1}\beta Y}{ZZ \mid \varepsilon}$ 11) $\dfrac{A' \mid \varepsilon}{\varepsilon \mid ZZ'}$	12) $\dfrac{a_i \mid Y'}{Z \mid \beta Y}$ 13) $\dfrac{\alpha \mid Y'}{Z \mid \beta Y}$ 14) $\dfrac{\beta \mid Y'}{Z \mid Z}$ 15) $\dfrac{B \mid \varepsilon}{X \mid Z}$ 16) $\dfrac{B' \mid \varepsilon}{A \mid Z}$ 17) $\dfrac{B'' \mid \varepsilon}{X \mid Z}$ 18) $\dfrac{\varepsilon \mid Y''}{Z \mid Y}$ 19) $\dfrac{B''' \mid \varepsilon}{A' \mid Z}$	20) $\dfrac{\alpha \mid Y'}{Z \mid \alpha Y}$ 21) $\dfrac{a_j\beta \mid Y'}{Z \mid \alpha Y}$ 22) $\dfrac{\alpha\beta \mid Y'}{Z \mid Z}$ 23) $\dfrac{C \mid \varepsilon}{X \mid Z}$

Опишем теперь работу 3-tt-системы T'. Начальное слово Xa_ia_jwY, $i = 1, \ldots, n$, $j = 1, \ldots, n+1$, соответствующее начальному слову a_ia_jw таг-системы T, перерабатывается 3-tt-системой T' следующим образом.

Пробирка A. Заметим, что любое слово вида WY, где $W \in \Sigma^*$, преобразуется в WY', т. е. $(W \mid Y, Z \mid Y') \vdash_3 (WY', ZY)$, или просто $WY \vdash_3 WY'$. Переработка слова ZY не приведет к появлению новых слов. (В дальнейшем легко проверяемые факты подобного рода мы будем опускать.)

Итак, $Xa_ia_jwY \vdash_3 Xa_ia_jwY'$. Затем $(Xa_ia_j \mid wY,\ X\beta\alpha^i\beta' \mid Z) \vdash_1 \vdash_1 (Xa_ia_jZ,\ X\beta\alpha^i\beta'wY')$. (С помощью слова $\beta\alpha^i\beta'$ кодируется первый слева символ a_i перерабатываемого слова). Потом $(X\beta \mid \alpha^i\beta'wY',\ B \mid Z) \vdash_2 (X\beta Z,\ B\alpha^i\beta'wY')$. Слово $B\alpha^i\beta'wY'$ в пробирке A далее не перерабатывается и попадает в пробирку B.

Пробирка B. Любое слово вида BW, где $W \in \Sigma^*$, преобразуется в XW: $BW \vdash_{15} XW$. В частности, $B\alpha^i\beta'wY' \vdash_{15} X\alpha^i\beta'wY'$. Далее $X\alpha^i\beta'wY' \vdash_{12} X\alpha^i\beta'w\beta Y$. Слово $X\alpha^i\beta'w\beta Y$ в пробирке B далее не перерабатывается и попадает в пробирку A.

Пробирка A. Здесь происходят реакции $X\alpha^i\beta'w\beta Y \vdash_3 \vdash_3 X\alpha^i\beta'w\beta Y' = X\alpha\alpha^{i-1}\beta'w\beta Y' \vdash_4 C\alpha^{i-1}\beta'w\beta Y'$, и слово $C\alpha^{i-1}\beta'w\beta Y'$ попадает в пробирку C.

Пробирка C. Слова вида CW, где $W \in \Sigma^*$, преобразуются в XW: $CW \vdash_{23} XW$. В частности, $C\alpha^{i-1}\beta'w\beta Y' \vdash_{23} X\alpha^{i-1}\beta'w\beta Y'$. Далее $X\alpha^{i-1}\beta'w\beta Y' \vdash_{21} X\alpha^{i-1}\beta'w\beta\alpha Y$, и $X\alpha^{i-1}\beta'w\beta\alpha Y$ попадает в пробирку A.

Пробирка A. Здесь происходят реакции $X\alpha^{i-1}\beta' w\beta\alpha Y \vdash_3 \vdash_3 X\alpha^{i-1}\beta' w\beta\alpha Y' = X\alpha\alpha^{i-2}\beta' w\beta\alpha Y' \vdash_4 C\alpha^{i-2}\beta' w\beta\alpha Y'$.

И так далее. В конце концов в пробирку A из пробирки C будет передано слово $X\beta' w\beta\alpha^i Y$. Далее $X\beta' w\beta\alpha^i Y \vdash_3 X\beta' w\beta\alpha^i Y' \vdash_5 B' w\beta\alpha^i Y'$, и слово $B' w\beta\alpha^i Y'$ попадает в пробирку B.

Пробирка B. Здесь происходят реакции $B' w\beta\alpha^i Y' \vdash_{13} \vdash_{13} B' w\beta\alpha^i \beta Y \vdash_{16} A w\beta\alpha^i \beta Y$, и слово $A w\beta\alpha^i \beta Y$ попадает в пробирку A.

Пробирка A. Рассмотрим два варианта переработки слова $A w\beta\alpha^i \beta Y$.

1. Преобразуем $A w\beta\alpha^i \beta Y \vdash_3 A w\beta\alpha^i \beta Y' \vdash_8 B'' w\beta\alpha^i \beta Y'$.

Пробирка B. Здесь происходят реакции $B'' w\beta\alpha^i \beta Y' \vdash_{17} \vdash_{17} X w\beta\alpha^i \beta Y' \vdash_{14} X w\beta\alpha^i \beta Z$, и слово $X w\beta\alpha^i \beta Z$ остается в пробирке B, не участвуя в дальнейших реакциях.

2. Преобразуем $(A w \mid \beta\alpha^i \beta Y,\ Z \mid P_i Y'') \vdash_7 (A w P_i Y'',\ Z\beta\alpha^i \beta Y)$. Слово $Z\beta\alpha^i \beta Y$ преобразуется в $Z\beta\alpha^i \beta Y'$ и остается в пробирке A, не участвуя в дальнейших реакциях. Затем $A w P_i Y'' \vdash_8 B'' w P_i Y''$, и слово $B'' w P_i Y''$ попадает в пробирку B.

Пробирка B. Здесь происходят реакции $B'' w P_i Y'' \vdash_{17} \vdash_{17} X w P_i Y'' \vdash_{18} X w P_i Y$, и слово $X w P_i Y$ передается в пробирку A. Тем самым смоделирован один шаг работы таг-системы T.

Рассмотрим теперь ситуацию, когда в процессе переработки таг-система T прекращает свою работу, т. е. перерабатываемое слово имеет вид hv, где $v \in \Omega^*$ (напомним, что $h = a_{n+1}$). Соответствующее слово 3-tt-системы T' имеет вид $XhvY$.

Пробирка A. Здесь $XhvY \vdash_3 XhvY' \vdash_9 X\beta\alpha^{n+1}\beta'' vY' \vdash_2 \vdash_2 B\alpha^{n+1}\beta'' vY'$, и слово $B\alpha^{n+1}\beta'' vY'$ попадает в пробирку B.

Далее происходит описанный выше процесс «перекачки информации»; в конце концов в пробирку A будет передано из пробирки C слово $X\beta'' v\beta\alpha^{n+1} Y$. Затем $X\beta'' v\beta\alpha^{n+1} Y \vdash_3 X\beta'' v\beta\alpha^{n+1} Y' \vdash_6 B''' v\beta\alpha^{n+1} Y'$, и слово $B''' v\beta\alpha^{n+1} Y'$ попадает в пробирку B.

Пробирка B. Здесь $B''' v\beta\alpha^{n+1} Y' \vdash_{13} B''' v\beta\alpha^{n+1}\beta Y \vdash_{19} A' v\beta\alpha^{n+1}\beta Y$, и слово $A' v\beta\alpha^{n+1}\beta Y$ попадает в пробирку A.

Пробирка A. Рассмотрим два варианта переработки слова $A' v\beta\alpha^{n+1}\beta Y$.

1. Преобразуем $A' v\beta\alpha^{n+1}\beta Y \vdash_3 A' v\beta\alpha^{n+1}\beta Y' \vdash_{11} v\beta\alpha^{n+1}\beta Y'$, и слово $v\beta\alpha^{n+1}\beta Y'$ попадает в пробирки B и C.

Пробирка B. Здесь $v\beta\alpha^{n+1}\beta Y' \vdash_{14} v\beta\alpha^{n+1}\beta Z$, и слово $v\beta\alpha^{n+1}\beta Z$ остается в пробирке B, не участвуя в дальнейших реакциях.

Пробирка C. Здесь $v\beta\alpha^{n+1}\beta Y' \vdash_{22} v\beta\alpha^{n+1}\beta Z$, и слово $v\beta\alpha^{n+1}\beta Z$ остается в пробирке C, не участвуя в дальнейших реакциях.

2. Преобразуем $A' v\beta\alpha^{n+1}\beta Y \vdash_{11} v\beta\alpha^{n+1}\beta Y$. Затем $(v \mid \beta\alpha^{n+1}\beta Y,\ ZZ \mid) \vdash_{10} (v,\ ZZ\beta\alpha^{n+1}\beta Y)$. Далее $ZZ\beta\alpha^{n+1}\beta Y \vdash_3 ZZ\beta\alpha^{n+1}\beta Y'$. Слово $ZZ\beta\alpha^{n+1}\beta Y'$ остается в пробирке A, не участвуя в дальнейших реакциях и не приводя к возникновению новых слов.

Полученное слово v (состоящее только из букв алфавита Ω) и есть искомый результат. Легко видеть, что в пробирке A нет других слов в алфавите Ω.

Лемма 6 доказана.

Непосредственно из этой леммы и существования универсальной 2-таг-системы следует

Теорема 6 [30]. *Существует универсальная расширенная 3-tt-система.*

§ 11. TVDH-системы степени 2 порождают любой рекурсивно перечислимый язык

Обозначим через RE множество всех рекурсивно перечислимых языков, через VDH_n — множество языков, порожденных TVDH-системами степени не более n, а через VDH_* — множество всех языков этого типа.

Теорема 7. *TVDH-системы степени 2 порождают любой рекурсивно перечислимый язык, т. е. имеют место равенства* $RE = VDH_2 = VDH_3 = \ldots = VDH_*$.

Основные шаги доказательства. 1. Для простоты вместо рекурсивно перечислимых языков мы будем рассматривать рекурсивно перечислимые множества натуральных чисел.

2. Известно, что каждое рекурсивно перечислимое множество L порождается некоторой машиной Тьюринга T_L: существует общерекурсивная функция $f_L(x)$, у которой $\mathrm{Val}(f_L) = L$, и машина T_L вычисляет ее, т. е. для каждого числа x имеет место соотношение $q_1 01^x \overset{T_L}{\Rightarrow} q_0 01^{f_L(x)}0\ldots0$.

Множество всех таких заключительных слов (без символа q_0) обозначим через L', т. е. $L' = \{01^{f_L(x)}0\ldots0\colon x = 0, 1, \ldots\}$.

3. Используя машину Тьюринга T_L, мы построим машину T'_L, работающую следующим образом: $q_1 01^x \overset{T'_L}{\Rightarrow} 01^{x+1} q_0 01^{f_L(x)}0\ldots0$.

Для этого мы используем простую вспомогательную машину Тьюринга C, преобразующую $q_1 01^x \overset{C}{\Rightarrow} 01^{x+1} q_0 01^x$, так что $T'_L = T_L \circ C$.

4. Мы строим TVDH-систему H_L степени 2, которая моделирует T'_L и такова, что $L(H_L) = L'$. Система H_L имеет аксиому Xq_1Y и работает так: $Xq_1Y \vdash X01q_{i_1}0Y \vdash X01q_0 01^{f_L(0)}0\ldots0Y \vdash \{01^{f_L(0)}0\ldots0,\ Xq_1 01Y\}$. Имея $01^{f_L(0)}0\ldots0$, продолжим: $Xq_1 01Y \vdash X011q_{i_1}01Y \vdash X011q_0 01^{f_L(1)}0\ldots0Y \vdash$ $\vdash \{01^{f_L(1)}0\ldots0,\ Xq_1011Y\}$. Результатом является $01^{f_L(1)}0\ldots0$. Продолжая, получаем все слова $01^{f_L(x)}0\ldots0$. Ясно, что $L(H_L) = L'$.

Детали доказательства. Машина T'_L имеет символы 0 (пустой символ) и 1 и состояния $q_0, q_1, \ldots, q_n$ (q_1 — начальное состояние, q_0 — финальное). Опишем TVDH-систему H_L, которая моделирует машину T'_L.

Терминальный алфавит T системы H_L есть $\{0, 1\}$. Алфавит V таков: $V = \{0, 1\} \cup \{X, Y, Y', q_1, \ldots, q_n, q'_1, \ldots, q'_n, q''_1, \ldots, q''_n, t_1, \ldots, t_5, t_R, t_L, Q, Z, \rightarrow, \leftarrow\}$. Множество аксиом есть $A = \{Xq_1Y,\ t_1q'_i0Y,\ t_2q_i0Y,\ Zq_k,\ t_Ra_lZ,\ t_Lq_ka_jZ,\ Xq''_k0a_jt_3,\ Xq_kt_4,\ Z\rightarrow,\ t_RQZ,\ t_3a_jZ,\ t_5\leftarrow,\ Za_j,\ t_4\leftarrow Z,\ Y',\ \leftarrow Y,\ Xq_1Z\}$; здесь $i = 1, \ldots, n$, $k = 0, 1, \ldots, n$ и $a_j, a_l = 0, 1$.

Замечание 6. В компонентах R_1 и R_2 системы H_L для каждой (за исключением Xq_1Y) аксиомы α из множества A имеются правила $\frac{\alpha\,|\,\varepsilon}{\alpha\,|\,\varepsilon}$. Таким образом, все (за исключением Xq_1Y) аксиомы доступны для применения на каждом шаге вычислений системы H_L.

Пусть $\alpha q_i a_j \beta$ — произвольная конфигурация машины T'_L. Систему H_L мы строим таким образом, чтобы слово $X\alpha q_i a_j \beta Y$ было доступно компоненте R_1 системы H_L, но не компоненте R_2.

1. Специальный случай. Система H_L моделирует правый край ленты машины Тьюринга T'_L: $X\alpha q_i Y \vdash X\alpha q_i 0Y$. Для этого применяем правила *)

1.1) $\dfrac{a_p\,|\,q_iY}{t_1\,|\,q'_i0Y}$, $a_p \in \{0, 1, X\}$, $i = 1, \ldots, n$,

2.1) $\dfrac{a_p\,|\,q'_i0Y}{t_2\,|\,q_i0Y}$, $a_p \in \{0, 1, X\}$, $i = 1, \ldots, n$.

*) Первый элемент номера правила обозначает, к какой компоненте системы H_L оно относится.

Применяя их, имеем $\{X\alpha \mid q_iY,\ t_1 \mid q_i'0Y\} \vdash_{1.1} \{X\alpha q_i'0Y,\ t_1q_iY\}$ и $\{X\alpha \mid q_i'0Y,\ t_2 \mid q_i0Y\} \vdash_{2.1} \{X\alpha q_i0Y,\ t_2q_i'0Y\}$. Молекулы t_1q_iY и $t_2q_i'0Y$ будут удалены. (Молекулы $t_1q_i'0Y$ и t_2q_i0Y — аксиомы системы H_L.)

2. Когда машина Тьюринга T_L' переходит от конфигурации $\alpha q_ia_ja_t\beta$ к $\alpha a_lq_ka_t\beta$ (применяется команда $q_ia_ja_lRq_k$), тогда система H_L, обрабатывая молекулу $X\alpha q_ia_ja_t\beta Y$, получает молекулу $X\alpha a_lq_ka_t\beta Y$ (эти молекулы появляются как обрабатываемые молекулы в компоненте R_1).

Для команды $q_ia_ja_lRq_k$ машины T_L', $i=1,\ldots,n$, $k=0,1,\ldots,n$, $a_j, a_l \in \{0,1\}$, в H_L имеются следующие правила рекомбинации:

1.2) $\dfrac{a_p \mid q_ia_j}{t_R \mid a_lq_k}$, $a_p \in \{0,1,X\}$,

2.2) $\dfrac{a_lq_k \mid \varepsilon}{t_Rq_ia_j \mid \varepsilon}$,

2.3) $\dfrac{Z \mid q_i}{t_Ra_j \mid Z}$, $i=0,1,\ldots,n$, $a_j \in \{0,1\}$.

Применяя их, имеем $\{X\alpha \mid q_ia_ja_t\beta Y,\ t_R \mid a_lq_k\} \vdash_{1.2} \{X\alpha a_lq_k,\ t_Rq_ia_ja_t\beta Y\}$ и $\{X\alpha a_lq_k \mid,\ t_Rq_ia_j \mid a_t\beta Y\} \vdash_{2.2} \{X\alpha a_lq_ka_t\beta Y,\ t_Rq_ia_j\}$. Молекула $t_Rq_ia_j$ будет удалена. Далее, $\{Z \mid q_i,\ t_Ra_j \mid Z\} \vdash_{2.3} \{ZZ,\ t_Ra_jq_i\}$. Это правило из аксиом Zq_k и t_Ra_lZ готовит молекулу $t_Ra_lq_k$, требуемую правилом 1.2). Молекула ZZ удаляется.

3. Аналогично, когда машина Тьюринга T_L' переходит от конфигурации $\alpha a_tq_ia_j\beta$ к $\alpha q_ka_ta_l\beta$ (применяется команда $q_ia_ja_lLq_k$), тогда система H_L, обрабатывая молекулу $X\alpha a_tq_ia_j\beta Y$, получает молекулу $X\alpha q_ka_ta_l\beta Y$ (эти молекулы появляются как обрабатываемые молекулы в компоненте R_1).

Для команды $q_ia_ja_lLq_k$ машины T_L', $i=1,\ldots,n$, $k=0,1,\ldots,n$, $a_j, a_l \in \{0,1\}$, в H_L имеются следующие правила рекомбинации:

1.3) $\dfrac{a_p \mid a_tq_ia_j}{t_L \mid q_ka_ta_l}$, $a_p \in \{0,1,X\}$, $a_t \in \{0,1\}$,

2.4) $\dfrac{a_pq_ka_ta_l \mid \varepsilon}{t_La_tq_ia_j \mid \varepsilon}$, $a_p \in \{0,1,X\}$, $a_t \in \{0,1\}$,

2.5) $\dfrac{Z \mid a_t}{t_Lq_ia_j \mid Z}$, $i=0,1,\ldots,n$, $a_j, a_t \in \{0,1\}$.

Применяя их, имеем $\{X\alpha \mid a_tq_ia_j\beta Y,\ t_L \mid q_ka_ta_l\} \vdash_{1.3} \{X\alpha q_ka_ta_l,\ t_La_tq_ia_j\beta Y\}$ и $\{X\alpha q_ka_ta_l \mid,\ t_La_tq_ia_j \mid \beta Y\} \vdash_{2.4} \{X\alpha q_ka_ta_l\beta Y,\ t_La_tq_ia_j\}$. Молекула $t_La_tq_ia_j$ будет удалена. Далее, $\{Z \mid a_t,\ t_Lq_ia_j \mid Z\} \vdash_{2.5} \{ZZ,\ t_Lq_ia_ja_t\}$. Это правило из аксиом Za_l и $t_Lq_ka_tZ$ готовит молекулу $t_Lq_ka_ta_l$, требуемую правилом 1.3). Молекула ZZ удаляется.

3'. Специальный случай — система H_L моделирует левый сдвиг и левый край ленты машины Тьюринга T_L': $Xq_ia_j\beta Y \vdash Xq_k0a_l\beta Y$. Для этого применяем правила

1.4) $\dfrac{Xq_ia_j \mid a_p}{Xq_k''0a_l \mid t_3}$, $a_p \in \{0,1,Y\}$,

2.6) $\dfrac{Xq_i'' \mid 0}{Xq_i \mid t_4}$, $i=0,1,\ldots,n$.

Применяя их, имеем $\{Xq_ia_j \mid \beta Y,\ Xq_k''0a_l \mid t_3\} \vdash_{1.4} \{Xq_ia_jt_3,\ Xq_k''0a_l\beta Y\}$ и $\{Xq_k'' \mid 0a_l\beta Y,\ Xq_k \mid t_4\} \vdash_{2.6} \{Xq_k''t_4,\ Xq_k0a_l\beta Y\}$. Молекулы $Xq_ia_jt_3$ и $Xq_k''t_4$ будут удалены. (Молекулы $Xq_k''0a_lt_3$ и Xq_it_4 — аксиомы системы H_L.)

4. Напомним, что система H_L имеет аксиому Xq_1Y и выполняет преобразование $Xq_1Y \vdash_{1.1,2.1} Xq_10Y$. Затем H_L моделирует машину T_L', и после

нескольких шагов мы получаем молекулу $X01q_001^{f_L(0)}0\ldots0Y$ для компоненты R_1 системы H_L.

Используя приводимые ниже правила, мы получаем молекулу $01^{f_L(0)}0\ldots0$ как результат (поскольку она содержит только буквы терминального алфавита системы H_L) и молекулу Xq_101Y для дальнейшей работы.

Итак, мы имеем молекулу $X01^{x+1}q_001^{f_L(x)}0\ldots0Y$ для компоненты R_1 системы H_L. Запишем эту молекулу как $X\alpha q_0\beta Y$. Тогда нашей целью является получение молекулы β как результата и молекулы $Xq_1\alpha Y$ для дальнейшей работы. Потребуются следующие правила рекомбинации (всюду подразумевается, что $a_j \in \{0, 1\}$):

1.5) $\dfrac{a_p \mid q_0 a_j}{t_R \mid Q \rightarrow}$, $a_p \in \{0, 1\}$;

1.6) $\dfrac{a_p \mid \rightarrow a_j}{t_3 \mid a_j \rightarrow}$, $a_p \in \{0, 1, Q\}$;

1.7) $\dfrac{\varepsilon \mid \rightarrow Y}{t_5 \mid \leftarrow}$;

1.8) $\dfrac{\leftarrow a_j \mid \varepsilon}{t_4 a_j \leftarrow \mid \varepsilon}$;

1.9) $\dfrac{Z \mid a_j}{t_4 \leftarrow \mid Z}$;

1.10) $\dfrac{\varepsilon \mid Q \leftarrow Y'}{\varepsilon \mid \leftarrow Y}$;

2.7) $\dfrac{Q \rightarrow \mid \varepsilon}{t_R q_0 \mid \varepsilon}$;

2.8) $\dfrac{Z \mid \rightarrow}{t_R Q \mid Z}$;

2.9) $\dfrac{a_j \rightarrow \mid \varepsilon}{t_3 \rightarrow a_j \mid \varepsilon}$;

2.10) $\dfrac{Z \mid \rightarrow}{t_3 a_j \mid Z}$;

2.11) $\dfrac{a_p \mid a_j \leftarrow}{t_4 \mid \leftarrow a_j}$, $a_p \in \{X, Q, 0, 1\}$;

2.12) $\dfrac{Q \leftarrow \mid a_j}{\varepsilon \mid Y'}$;

2.13) $\dfrac{X \leftarrow \mid a_j}{Xq_1 \mid Z}$.

Преобразование проходит следующим образом. Вначале выполняем $\{Z \mid \rightarrow, t_R Q \mid Z\} \vdash_{2.8} \{ZZ, t_R Q \rightarrow\}$ и $\{Z \mid \rightarrow, t_3 a_j \mid Z\} \vdash_{2.10} \{ZZ, t_3 a_j \rightarrow\}$, изготавливая из аксиом системы H_L молекулы $t_R Q \rightarrow$ и $t_3 a_j \rightarrow$ для правил 1.5) и 1.6). Молекула ZZ удаляется. Затем $\{X\alpha \mid q_0\beta Y, t_R \mid Q \rightarrow\} \vdash_{1.5} \{X\alpha Q \rightarrow, t_R q_0\beta Y\}$, $\{X\alpha Q \rightarrow \mid, t_R q_0 \mid \beta Y\} \vdash_{2.7} \{X\alpha Q \rightarrow \beta Y, t_R q_0\}$. Молекула $t_R q_0$ удаляется. Далее, $\{X\alpha Q\beta' \mid \rightarrow a_j\beta'' Y, t_3 \mid a_j \rightarrow\} \vdash_{1.6} \{X\alpha Q\beta' a_j \rightarrow, t_3 \rightarrow a_j\beta'' Y\}$, $\{X\alpha Q\beta' a_j \rightarrow \mid, t_3 \rightarrow a_j \mid \beta'' Y\} \vdash_{2.9} \{X\alpha Q\beta' a_j \rightarrow \beta'' Y, t_3 \rightarrow a_j\}$. Молекула $t_3 \rightarrow a_j$ будет удалена.

После нескольких шагов получаем молекулу $X\alpha Q\beta \rightarrow Y$ для компоненты R_1 системы H_L.

Далее, $\{X\alpha Q\beta \mid \rightarrow Y, t_5 \mid \leftarrow\} \vdash_{1.7} \{X\alpha Q\beta \leftarrow, t_5 \rightarrow Y\}$. Молекула $t_5 \rightarrow Y$ будет удалена. (Молекула $t_5 \leftarrow$ есть аксиома системы H_L.) Преобразование $\{Z \mid a_j, t_4 \leftarrow \mid Z\} \vdash_{1.9} \{ZZ, t_4 \leftarrow a_j\}$ из аксиом системы H_L готовит молекулу $t_4 \leftarrow a_j$ для правила 2.11). Молекула ZZ удаляется. Затем $\{X\alpha Q\beta' \mid a_j \leftarrow \beta'', t_4 \mid \leftarrow a_j\} \vdash_{2.11} \{X\alpha Q\beta' \leftarrow a_j, t_4 a_j \leftarrow \beta''\}$, $\{X\alpha Q\beta' \leftarrow a_j \mid, t_4 a_j \leftarrow \mid \beta''\} \vdash_{1.8} \{X\alpha Q\beta' \leftarrow a_j\beta'', t_4 a_j \leftarrow\}$. Молекула $t_4 a_j \leftarrow$ будет удалена.

После нескольких шагов мы получим молекулу $X\alpha Q \leftarrow \beta$ для компоненты R_2 системы H_L.

Далее, $\{X\alpha Q \leftarrow \mid \beta, \mid Y'\} \vdash_{2.12} \{X\alpha Q \leftarrow Y', \beta\}$. (Молекула Y' — аксиома системы H_L.) Таким образом, мы получили молекулу β как результат. Затем $\{X\alpha \mid Q \leftarrow Y', \mid \leftarrow Y\} \vdash_{1.10} \{X\alpha \leftarrow Y, Q \leftarrow Y'\}$. Молекула $Q \leftarrow Y'$ будет удалена. (Молекула $\leftarrow Y$ есть аксиома системы H_L.)

Итак, после нескольких шагов мы получим молекулу $X \leftarrow \alpha Y$ для компоненты R_2 системы H_L.

Наконец, $\{X \leftarrow \mid \alpha Y, Xq_1 \mid Z\} \vdash_{2.13} \{X \leftarrow Z, Xq_1\alpha Y\}$ и мы получим молекулу $Xq_1\alpha Y = Xq_101^{x+1}Y$ для компоненты R_1 системы H_L для дальнейшей работы ($Xq_1 Z$ — аксиома системы H_L). Молекула $X \leftarrow Z$ удаляется.

§ 12. Универсальная TVDH-система степени 2

Теорема 8 [31]. *Существует универсальная TVDH-система степени* 2.

Легко видеть, что эта теорема немедленно получается из следующей леммы.

Лемма 7. *Для каждой таг-системы T существует моделирующая ее TVDH-система T' степени* 2.

Доказательство. Рассмотрим таг-систему T:

$$\begin{cases} a_i \to P_i, & i = 1, \ldots, n; \\ a_{n+1} \to \text{STOP}. \end{cases}$$

Пусть $\Omega = \{a_1, \ldots, a_{n+1}\}$. Построим TVDH-систему T'. Пусть ее терминальный алфавит есть Ω, алфавит V есть $V = \Omega \cup \{X, X', X'', Y, Z, t_1, \ldots, t_5, \overset{1}{\to}, \ldots, \overset{n+1}{\to}, \leftarrow\}$. (Символы a_{n+1} и $\overset{n+1}{\to}$ мы иногда будем обозначать h и $\overset{h}{\to}$, соответственно.) Множество аксиом A таково: $A = \{X' \overset{1}{\to} Z, \ldots, X' \overset{n}{\to} Z, Z \overset{1}{\to}, \ldots, Z \overset{n}{\to}, t_1 a_1 Z, \ldots, t_1 a_{n+1} Z, \leftarrow Z, Z a_1 t_2, \ldots, Z a_{n+1} t_2, t_5 P_1 Z, \ldots, t_5 P_n Z, h \overset{h}{\to} Z, Z \overset{h}{\to} t_4, t_3 a_1 Z, \ldots, t_3 a_{n+1} Z, X'' Z, XZ, ZZ, Z \leftarrow Y\}$.

Множество правил рекомбинации таково (вновь первый элемент номера обозначает, к какой компоненте, R_1 или R_2, правило относится; всюду $i = 1, \ldots, n$ и $j, l = 1, \ldots, n+1$):

1.1) $\dfrac{X a_i a_j \mid a_l}{X' \overset{i}{\to} \mid Z}$, 1.5) $\dfrac{a_l \leftarrow \mid a_j}{\leftarrow a_l \mid t_2}$, 1.10) $\dfrac{Z \mid \leftarrow Y}{t_5 P_i \mid Z}$, 2.4) $\dfrac{Z \mid a_j t_2}{\leftarrow \mid Z}$,

1.1') $\dfrac{X a_i a_j \mid Y}{X' \overset{i}{\to} \mid Z}$, 1.6) $\dfrac{X' \leftarrow \mid a_j}{X'' \mid Z}$, 2.1) $\dfrac{X' \mid \overset{i}{\to} a_j}{t_1 \mid a_j \overset{i}{\to}}$, 2.5) $\dfrac{\varepsilon \mid a_j \leftarrow t_2}{\varepsilon \mid \leftarrow a_j}$,

1.2) $\dfrac{a_j \overset{i}{\to} \mid \varepsilon}{t_1 \overset{i}{\to} a_j \mid \varepsilon}$, 1.7) $\dfrac{Xh \mid a_j}{h \overset{h}{\to} \mid Z}$, 2.2) $\dfrac{a_l \mid \overset{i}{\to} a_j}{t_1 \mid a_j \overset{i}{\to}}$, 2.6) $\dfrac{X'' \mid a_j}{X \mid Z}$,

1.3) $\dfrac{Z \mid \overset{i}{\to}}{t_1 a_j \mid Z}$, 1.8) $\dfrac{a_j \overset{h}{\to} \mid t_4}{t_3 \overset{h}{\to} a_j \mid \varepsilon}$, 2.3) $\dfrac{a_j \mid \overset{i}{\to} Y}{t_5 \mid P_i \leftarrow Y}$, 2.7) $\dfrac{a_l \mid \overset{h}{\to} a_j}{t_3 \mid a_j \overset{h}{\to} t_4}$,

1.4) $\dfrac{a_j \leftarrow \mid Y}{\leftarrow a_j \mid t_2}$, 1.9) $\dfrac{Z \mid \overset{h}{\to} t_4}{t_3 a_j \mid Z}$, 2.3') $\dfrac{X' \mid \overset{i}{\to} Y}{t_5 \mid P_i \leftarrow Y}$, 2.8) $\dfrac{\varepsilon \mid \overset{h}{\to} Y}{ZZ \mid \varepsilon}$.

Замечание 7. Для каждой аксиомы α из множества A в R_1 и R_2 имеются правила $\dfrac{\alpha \mid \varepsilon}{\alpha \mid \varepsilon}$.

Система T' работает следующим образом.

1. Если таг-система T стартует со слова w в алфавите Ω, то TVDH-система T' стартует со слова XwY; оно добавляется к аксиомам системы T' без добавления новых правил рекомбинации. Если таг-система T останавливается на слове hv из Ω^*, то система T' производит то же самое слово hv и никаких других слов в терминальном алфавите Ω.

Пусть $XwY = X a_{i_0} a_{j_0} a_{l_0} a_{k_0} w' Y$, где $i_0 \in \{1, \ldots, n\}$, $j_0, l_0, k_0 \in \{1, \ldots, n+1\}$.

2. Все аксиомы α из множества A доступны для операций рекомбинации на каждом шаге работы TVDH-системы T' благодаря правилам $\dfrac{\alpha \mid \varepsilon}{\alpha \mid \varepsilon}$.

3. Когда управление передается компоненте R_1, в силу правила 2.4) доступны молекулы $\leftarrow a_j t_2$.

4. Когда управление передается компоненте R_2, доступны молекулы $t_1 a_j \xrightarrow{i}$, $t_3 a_j \xrightarrow{h} t_4$ и $t_5 P_i \leftarrow Y$ в силу правил 1.3), 1.9) и 1.10), соответственно.

5. Имеем $\{Xa_{i_0} a_{j_0} \mid a_{l_0} a_{k_0} w'Y,\ X' \xrightarrow{i_0} \mid Z\} \vdash_{1.1} \{Xa_{i_0} a_{j_0} Z,\ X' \xrightarrow{i_0} a_{l_0} a_{k_0} w'Y\}$. Таким образом, $L_1 = A \cup \{XwY\}$ и $L_2 = \{X' \xrightarrow{i_0} a_{l_0} a_{k_0} w'Y,\ Xa_{i_0} a_{j_0} Z,\ t_1 a_j \xrightarrow{i},\ t_3 a_j \xrightarrow{h} t_4,\ t_5 P_i \leftarrow Y\} \cup A$.

6. Молекулы $Xa_{i_0} a_{j_0} Z$, $t_1 a_j \xrightarrow{i}$ $(j \neq l_0,\ i \neq i_0)$, $t_3 a_j \xrightarrow{h} t_4$ и $t_5 P_i \leftarrow Y$ не могут участвовать в операциях рекомбинации в R_2 и поэтому удаляются. Далее, $\{X' \mid \xrightarrow{i_0} a_{l_0} a_{k_0} w'Y,\ t_1 \mid a_{l_0} \xrightarrow{i_0}\} \vdash_{2.1} \{X' a_{l_0} \xrightarrow{i_0},\ t_1 \xrightarrow{i_0} a_{l_0} a_{k_0} w'Y\}$. Итак, $L_3 = \{X' a_{l_0} \xrightarrow{i_0},\ t_1 \xrightarrow{i_0} a_{l_0} a_{k_0} w'Y,\ \leftarrow a_j t_2\} \cup A$.

7. Молекулы $\leftarrow a_j t_2$ не могут участвовать в операциях рекомбинации в R_1 и поэтому удаляются. Далее, $\{X' a_{l_0} \xrightarrow{i_0} \mid,\ t_1 \xrightarrow{i_0} a_{l_0} \mid a_{k_0} w'Y\} \vdash_{1.2} \vdash_{1.2} \{X' a_{l_0} \xrightarrow{i_0} a_{k_0} w'Y,\ t_1 \xrightarrow{i_0} a_{l_0}\}$. Итак, $L_4 = \{X' a_{l_0} \xrightarrow{i_0} a_{k_0} w'Y,\ t_1 \xrightarrow{i_0} a_{l_0},\ t_1 a_j \xrightarrow{i},\ t_3 a_j \xrightarrow{h} t_4,\ t_5 P_i \leftarrow Y\} \cup A$.

8. Молекулы $t_1 \xrightarrow{i_0} a_{l_0}$, $t_1 a_j \xrightarrow{i}$ $(j \neq k_0,\ i \neq i_0)$, $t_3 a_j \xrightarrow{h} t_4$ и $t_5 P_i \leftarrow Y$ не могут участвовать в операциях рекомбинации в R_2 и поэтому удаляются. Далее, $\{X' a_{l_0} \mid \xrightarrow{i_0} a_{k_0} w'Y,\ t_1 \mid a_{k_0} \xrightarrow{i_0}\} \vdash_{2.2} \{X' a_{l_0} a_{k_0} \xrightarrow{i_0},\ t_1 \xrightarrow{i_0} a_{k_0} w'Y\}$. Итак, $L_5 = \{X' a_{l_0} a_{k_0} \xrightarrow{i_0},\ t_1 \xrightarrow{i_0} a_{k_0} w'Y,\ \leftarrow a_j t_2\} \cup A$.

9. Молекулы $\leftarrow a_j t_2$ не могут участвовать в операциях рекомбинации в R_1 и поэтому удаляются. Далее, $\{X' a_{l_0} a_{k_0} \xrightarrow{i_0} \mid,\ t_1 \xrightarrow{i_0} a_{k_0} \mid w'Y\} \vdash_{1.2} \vdash_{1.2} \{X' a_{l_0} a_{k_0} \xrightarrow{i_0} w'Y,\ t_1 \xrightarrow{i_0} a_{k_0}\}$. Итак, $L_6 = \{X' a_{l_0} a_{k_0} \xrightarrow{i_0} w'Y,\ t_1 \xrightarrow{i_0} a_{k_0},\ t_1 a_j \xrightarrow{i},\ t_3 a_j \xrightarrow{h} t_4,\ t_5 P_i \leftarrow Y\} \cup A$.

Так мы моделируем движение информации от одного конца перерабатываемой молекулы к другому.

Через несколько шагов получаем $L_t = \{X' w'' a_{m_0} \xrightarrow{i_0} Y,\ t_1 \xrightarrow{i_0} a_{m_0}$ $(m_0 = 1, \ldots, n+1),\ t_1 a_j \xrightarrow{i},\ t_3 a_j \xrightarrow{h} t_4,\ t_5 P_i \leftarrow Y\} \cup A$.

10. Молекулы $t_1 \xrightarrow{i_0} a_{m_0}$, $t_1 a_j \xrightarrow{i}$, $t_3 a_j \xrightarrow{h} t_4$ и $t_5 P_i \leftarrow Y$ $(i \neq i_0)$ не могут участвовать в операциях рекомбинации в R_2 и поэтому удаляются. Далее, $\{X' w'' a_{m_0} \mid \xrightarrow{i_0} Y,\ t_5 \mid P_{i_0} \leftarrow Y\} \vdash_{2.3} \{X' w'' a_{m_0} P_{i_0} \leftarrow Y,\ t_5 \xrightarrow{i_0} Y\}$.

Мы приписываем к правому краю обрабатываемой молекулы необходимую продукцию P_{i_0}. Пусть $X' w'' a_{m_0} P_{i_0} \leftarrow Y = X' v a_{j_1} a_{k_1} \leftarrow Y$, где $j_1, k_1, l_1 \in \{1, \ldots, n+1\}$.

Итак, $L_{t+1} = \{X' v a_{j_1} a_{k_1} \leftarrow Y,\ t_5 \xrightarrow{i_0} Y,\ \leftarrow a_j t_2\} \cup A$.

11. Молекулы $t_5 \xrightarrow{i_0} Y$ и $\leftarrow a_j t_2$ $(j \neq k_1)$ не могут участвовать в операциях рекомбинации в R_1 и поэтому удаляются. Далее, $\{X' v a_{j_1} a_{k_1} \leftarrow \mid Y,\ \leftarrow a_{k_1} \mid t_2\} \vdash_{1.4} \{X' v a_{j_1} a_{k_1} \leftarrow t_2,\ \leftarrow a_{k_1} Y\}$. Итак, $L_{t+2} = \{X' v a_{j_1} a_{k_1} \leftarrow t_2,\ \leftarrow a_{k_1} Y,\ t_1 a_j \xrightarrow{i},\ t_3 a_j \xrightarrow{h} t_4,\ t_5 P_i \leftarrow Y\} \cup A$.

12. Молекулы $t_1 a_j \xrightarrow{i}$, $t_3 a_j \xrightarrow{h} t_4$ и $t_5 P_i \leftarrow Y$ не могут участвовать в операциях рекомбинации в R_2 и поэтому удаляются. Далее, $\{X' v a_{j_1} \mid a_{k_1} \leftarrow t_2,\ \mid \leftarrow a_{k_1} Y\} \vdash_{2.5} \{X' v a_{j_1} \leftarrow a_{k_1} Y,\ a_{k_1} \leftarrow t_2\}$. Итак, $L_{t+3} = \{X' v a_{j_1} \leftarrow a_{k_1} Y,\ a_{k_1} \leftarrow t_2,\ \leftarrow a_j t_2\} \cup A$.

13. Молекулы $a_{k_1} \leftarrow t_2$ и $\leftarrow a_j t_2$ $(j \neq j_1)$ не могут участвовать в операциях рекомбинации в R_1 и поэтому удаляются. Далее, $\{X' v a_{j_1} \leftarrow \mid a_{k_1} Y, \leftarrow a_{j_1} t_2\} \vdash_{1.5} \{X' v a_{j_1} \leftarrow t_2, \leftarrow a_{j_1} a_{k_1} Y\}$. Итак, $L_{t+4} = \{X' v a_{j_1} \leftarrow t_2, \leftarrow a_{j_1} a_{k_1} Y, t_1 a_j \xrightarrow{i}, t_3 a_j \xrightarrow{h} t_4, t_5 P_i \leftarrow Y\} \cup A$.

14. Молекулы $t_1 a_j \xrightarrow{i}$, $t_3 a_j \xrightarrow{h} t_4$ и $t_5 P_i \leftarrow Y$ не могут участвовать в операциях рекомбинации в R_2 и поэтому удаляются. Далее, $\{X' v \mid a_{j_1} \leftarrow t_2, \mid \leftarrow a_{j_1} a_{k_1} Y\} \vdash_{2.5} \{X' v \leftarrow a_{j_1} a_{k_1} Y, a_{j_1} \leftarrow t_2\}$. Итак, $L_{t+5} = \{X' v \leftarrow a_{j_1} a_{k_1} Y, a_{j_1} \leftarrow t_2, \leftarrow a_j t_2\} \cup A$.

Таким образом, через несколько шагов получаем $L_p = \{X' \leftarrow a_{l_1} v' Y, a_{l_1} \leftarrow t_2, \leftarrow a_j t_2\} \cup A$.

15. Молекулы $a_{l_1} \leftarrow t_2$ и $\leftarrow a_j t_2$ не могут участвовать в операциях рекомбинации в R_1 и поэтому удаляются. Далее, $\{X' \leftarrow \mid a_{l_1} v' Y, X'' \mid Z\} \vdash_{1.6} \{X'' a_{l_1} v' Y, X' \leftarrow Z\}$. Итак, $L_{p+1} = \{X'' a_{l_1} v' Y, X' \leftarrow Z, t_1 a_j \xrightarrow{i}, t_3 a_j \xrightarrow{h} t_4, t_5 P_i \leftarrow Y\} \cup A$.

16. Молекулы $X' \leftarrow Z$, $t_1 a_j \xrightarrow{i}$, $t_3 a_j \xrightarrow{h} t_4$ и $t_5 P_i \leftarrow Y$ не могут участвовать в операциях рекомбинации в R_2 и поэтому удаляются. Далее, $\{X'' \mid a_{l_1} v' Y, X \mid Z\} \vdash_{2.6} \{X a_{l_1} v' Y, X'' Z\}$. Итак, $L_{p+2} = \{X a_{l_1} v' Y, \leftarrow a_j t_2\} \cup A$.

Таким образом, один шаг работы таг-системы T смоделирован системой T'. После этого система T' переходит к п. 5 и начинает моделировать следующий шаг работы таг-системы T.

Если длина перерабатываемой молекулы равна 2, т. е. $|w| = |a_{i_0} a_{j_0}| = 2$, работают правила 1.1') и 2.3'), см. пп. 17 и 18.

17. Молекулы $\leftarrow a_j t_2$ не могут участвовать в операциях рекомбинации в R_1 и поэтому удаляются. Далее, $\{X a_{i_0} a_{j_0} \mid Y, X' \xrightarrow{i_0} \mid Z\} \vdash_{1.1'} \{X' \xrightarrow{i_0} Y, X a_{i_0} a_{j_0} Z\}$. Итак, $L_s = \{X' \xrightarrow{i_0} Y, X a_{i_0} a_{j_0} Z, t_1 a_j \xrightarrow{i}, t_3 a_j \xrightarrow{h} t_4, t_5 P_i \leftarrow Y\} \cup A$.

18. Молекулы $X a_{i_0} a_{j_0} Z$, $t_1 a_j \xrightarrow{i}$, $t_3 a_j \xrightarrow{h} t_4$ и $t_5 P_i \leftarrow Y$ $(i \neq i_0)$ не могут участвовать в операциях рекомбинации в R_2 и поэтому удаляются. Далее, $\{X' \mid \xrightarrow{i_0} Y, t_5 \mid P_{i_0} \leftarrow Y\} \vdash_{2.3'} \{X' P_{i_0} \leftarrow Y, t_5 \xrightarrow{i_0} Y\}$. Итак, $L_{s+1} = \{X' P_{i_0} \leftarrow Y, t_5 \xrightarrow{i_0} Y, \leftarrow a_j t_2\} \cup A$, и система T' переходит к п. 11.

19. Рассмотрим тот случай, когда вычисления таг-системы T прекращаются, т. е. обрабатываемая ею молекула w имеет вид $h a_{j_0} a_{k_0} w'$, а соответствующая молекула, обрабатываемая системой T', имеет вид $X h a_{j_0} a_{k_0} w' Y$.

Молекулы $\leftarrow a_j t_2$ не могут участвовать в операциях рекомбинации в R_1 и поэтому удаляются. Далее, $\{Xh \mid a_{j_0} a_{k_0} w' Y, h \xrightarrow{h} \mid Z\} \vdash_{1.7} \{h \xrightarrow{h} a_{j_0} a_{k_0} w' Y, XhZ\}$. Итак, $L_f = \{h \xrightarrow{h} a_{j_0} a_{k_0} w' Y, XhZ, t_1 a_j \xrightarrow{i}, t_3 a_j \xrightarrow{h} t_4, t_5 P_i \leftarrow Y\} \cup A$.

20. Молекулы XhZ, $t_1 a_j \xrightarrow{i}$, $t_3 a_j \xrightarrow{h} t_4$ $(j \neq j_0)$ и $t_5 P_i \leftarrow Y$ не могут участвовать в операциях рекомбинации в R_2 и поэтому удаляются. Далее, $\{h \mid \xrightarrow{h} a_{j_0} a_{k_0} w' Y, t_3 \mid a_{j_0} \xrightarrow{h} t_4\} \vdash_{2.7} \{h a_{j_0} \xrightarrow{h} t_4, t_3 \xrightarrow{h} a_{j_0} a_{k_0} w' Y\}$. Итак, $L_{f+1} = \{h a_{j_0} \xrightarrow{h} t_4, t_3 \xrightarrow{h} a_{j_0} a_{k_0} w' Y, \leftarrow a_j t_2\} \cup A$.

21. Молекулы $\leftarrow a_j t_2$ не могут участвовать в операциях рекомбинации в R_1 и поэтому удаляются. Далее, $\{h a_{j_0} \xrightarrow{h} \mid t_4, t_3 \xrightarrow{h} \mid a_{j_0} a_{k_0} w' Y\} \vdash_{1.8} \vdash_{1.8} \{h a_{j_0} \xrightarrow{h} a_{k_0} w' Y, t_3 \xrightarrow{h} a_{j_0} t_4\}$. Итак, $L_{f+2} = \{h a_{j_0} \xrightarrow{h} a_{k_0} w' Y, t_3 \xrightarrow{h} a_{j_0} t_4, t_1 a_j \xrightarrow{i}, t_3 a_j \xrightarrow{h} t_4, t_5 P_i \leftarrow Y\} \cup A$.

22. Молекулы $t_3 \xrightarrow{h} a_{j_0} t_4$, $t_1 a_j \xrightarrow{i}$, $t_3 a_j \xrightarrow{h} t_4$ $(j \neq k_0)$ и $t_5 P_i \leftarrow Y$ не могут участвовать в операциях рекомбинации в R_2 и поэтому удаляются. Далее, $\{ha_{j_0} \mid \xrightarrow{h} a_{k_0} w'Y,\ t_3 \mid a_{k_0} \xrightarrow{h} t_4\} \vdash_{2.7} \{ha_{j_0} a_{k_0} \xrightarrow{h} t_4,\ t_3 \xrightarrow{h} a_{k_0} w'Y\}$. Итак, $L_{f+3} = \{ha_{j_0} a_{k_0} \xrightarrow{h} t_4,\ t_3 \xrightarrow{h} a_{k_0} w'Y, \leftarrow a_j t_2\} \cup A$.

23. Молекулы $\leftarrow a_j t_2$ не могут участвовать в операциях рекомбинации в R_1 и поэтому удаляются. Далее, $\{ha_{j_0} a_{k_0} \xrightarrow{h} \mid t_4,\ t_3 \xrightarrow{h} a_{k_0} \mid w'Y\} \vdash_{1.8} \vdash_{1.8} \{ha_{j_0} a_{k_0} \xrightarrow{h} w'Y,\ t_3 \xrightarrow{h} a_{k_0} t_4\}$. Итак, $L_{f+4} = \{ha_{j_0} a_{k_0} \xrightarrow{h} w'Y,\ t_3 \xrightarrow{h} a_{k_0} t_4,\ t_1 a_j \xrightarrow{i},\ t_3 a_j \xrightarrow{h} t_4,\ t_5 P_i \leftarrow Y\} \cup A$. Таким образом, через несколько шагов получаем $L_z = \{w'' a_{m_0} \xrightarrow{h} Y,\ t_3 \xrightarrow{h} a_{m_0} t_4,\ t_1 a_j \xrightarrow{i},\ t_3 a_j \xrightarrow{h} t_4,\ t_5 P_i \leftarrow Y\} \cup A$.

24. Молекулы $t_3 \xrightarrow{h} a_{m_0} t_4$, $t_1 a_j \xrightarrow{i}$, $t_3 a_j \xrightarrow{h} t_4$ и $t_5 P_i \leftarrow Y$ не могут участвовать в операциях рекомбинации в R_2 и поэтому удаляются. Далее, $\{w'' a_{m_0} \mid \xrightarrow{h} Y,\ ZZ \mid\} \vdash_{2.8} \{w'' a_{m_0},\ ZZ \xrightarrow{h} Y\}$. Молекулы $ZZ \xrightarrow{h} Y$ будут удалены.

Молекула $w'' a_{m_0} = w$ и есть искомый результат!

СПИСОК ЛИТЕРАТУРЫ

1. Мальцев А. И. Алгоритмы и рекурсивные функции // М.: Наука, 1965, 1986 (2-е изд.).
2. Минский М. Вычисления и автоматы. — М.: Мир, 1971.
3. Павлоцкая Л. М. Разрешимость проблемы остановки для некоторых классов машин Тьюринга // Математич. заметки. — 1973. — Т. 13, № 6. — С. 899–909.
4. Павлоцкая Л. М. Достаточные условия разрешимости проблемы остановки для машин Тьюринга // Проблемы кибернетики. Вып. 33. — М.: Наука, 1978 — С. 91–118.
5. Рогожин Ю. В. Некоторые формализмы для машин Тьюринга // Ин-т кибернетики АН УССР / Препр. 74–44. – Киев, 1974.
6. Рогожин Ю. В. Семь универсальных машин Тьюринга // Системное и теоретическое программирование. Математич. исследования. Вып. 69. — Кишинев: АН МССР, 1982. — С. 76–90.
7. Рогожин Ю. В. Универсальная машина Тьюринга с 10 состояниями и 3 символами // Известия АН Республики Молдова. Математика. — 1992. — № 4(10). — С. 80–82.
8. Aanderaa S., Fischer P. The solvability of the halting problem for 2-state Post machine // J. Assoc. Comput. Mach. — 1967. — V. 14, № 4. — P. 677–682.
9. Adleman L. M. Molecular computation of solutions of combinatorial problems // Science. — 1994. — V. 226 (November). — P. 1021–1024.
10. Adleman L. M. On constracting a molecular computer // Computer Science Department, Univ. of Southern California / Manuscript. — 1995, Jan. 11. — ftp://us.edu/pub/csinfo/papers/adleman.
11. Bennett C. H. Logical reversibility of computation // IBM J. Res. Develop. — 1973. — V. 6. — P. 525–532.
12. Csuhaj-Varju E., Kari L., Păun G. Test tube distributed system based on splicing // Computer and AI. — 1996. — V. 15, № 2–3. — P. 211–232.
13. Davis M. A note on universal Turing machines // Automata studies / Ann of Math. Stud. V. 34 — Princeton: Princeton Univ. Press, 1956. — P. 167–175. [Имеется перевод: Дэвис М. Замечание об универсальных машинах Тьюринга // Автоматы / Сб. статей под ред. К. Шеннона и Дж. Маккарти. — М.: Изд-во иностр. лит., 1956. — С. 226–234.]
14. Davis M. The definition of the universal Turing machine // Proc. Amer. Math. Soc. — 1957. — V. 8, № 6. — P. 1125–1126.
15. Davis M. D., Weyuker E. J. Computability, complexity, and languages. — London: Academic Press, Inc., 1983.
16. Diekert V., Kudlek M. Small deterministic Turing machines // Papers on automata and languages / Dept. of Math., Karl Marx Univ. of Economics (Budapest). — 1989. — V. 188, № 4. — P. 77–87.
17. Ferretti C., Mauri G., Zandron C. Nine test tubes generate any RE language // Proc. 2*nd* International Colloquium "Universal Machines and Computations" (MCU/UMC'98). — Metz (France), 1998. — V. 2. — P. 30–41.
18. Fischer P. On formalisms for Turing machine // J. Assoc. Comput. Mach. — 1965. — V. 12. № 4. — P. 570–580.

19. Gruska J. Foundation of Computing. — Boston: International Thomson Computer Press, 1997.
20. Head T. Formal language theory and DNA: an analysis of the generative capacity of specific recombinant behaviors // Bulletin of Mathematical Biology. — 1987. — V. 49, № 6. — P. 737–759.
21. Herman G. T. A new hierarchy of elementary functions // Proc. Amer. Math. Soc. — 1969. — V. 20. — P. 557–562.
22. Hooper P. K. Some small universal Turing machines // Information Sciences. — 1969. — V. 1. — P. 205–215.
23. Kari L. DNA computers, tomorrow's reality // Bulletin of the European Association for Theoretical Computer Science. — 1996. — V. 59 (June). — P. 256–266.
24. Kari L. DNA computing: arrival of biological mathematics // The mathematical intelligencer. — 1997. — V. 19, № 2. — P. 9–22.
25. Korec I. Small universal register machines // Theoretical Computer Science. — 1996. — V. 168, № 2. — P. 267–301.
26. Kudlek M. Small deterministic Turing machines // Theoretical Computer Science. — 1996. — V. 168, № 2. — P. 241–255.
27. Margenstern M. Non erasing Turing machines: a frontier between a decidable halting problem and universality // Lecture Notes in Computer Science. — 1993. — V. 710. — P. 375–385.
28. Margenstern M. Decidability and undecidability of the halting problem on Turing machines, a survey // Lecture Notes in Computer Science. — 1997. — V. 1234. — P. 226–236.
29. Margenstern M. Frontier between decidability and undecidability: a survey // Proc. 2*nd* International Colloquium "Universal Machines and Computations" (MCU/UMC'98). — Metz (France), 1998. — V. 1. — P. 141–177.
30. Margenstern M., Priese L., Рогожин Ю. Молекулярные вычисления: трех *test tubes* достаточно для порождения любого рекурсивно перечислимого языка // Тр. VI Международного семинара «Распределенная обработка информации». — Новосибирск: Изд-во СО РАН, 1998. — С. 132-136.
31. Margenstern M., Rogozhin Yu. A universal time-varying distributed H-system of degree 2 // 4*th* International Meeting on DNA Based Computers (June 15–19, 1998) / Preliminary proceedings. — University of Pennsylvania, 1998. — P. 83–84.
32. Matiyasevich Yu. Universal polynomials // Proc. 2*nd* International Colloquium "Universal Machines and Computations" (MCU/UMC'98). — Metz (France), 1998. — V. 1. — P. 1–20.
33. Minsky M. L. Size and structure of universal Turing machines using tag systems // Recursive Function Theory. Proc. symp. in pure mathematics. V. 5. — AMS, 1962. — P. 229–238.
34. Nozaki A. On notion of universality of Turing machine // Kybernetika. — 1969. — V. 5, № 1. — P. 29–42.
35. Păun Gh. DNA computing: distributed splicing systems // Lecture Notes in Computer Science. — 1997. — V. 1261. — P. 353–370.
36. Păun G. DNA computing based on splicing: universality results // Proc. 2*nd* International Colloquium "Universal Machines and Computations" (MCU/UMC'98). — Metz (France), 1998. — V. 1. — P. 67–91.
37. Păun G., Rozenberg G., Salomaa A. Computing by splicing // Theoretical Computer Science. — 1996. — V. 168. — P. 321–336.
38. Păun G., Rozenberg G., Salomaa A. DNA computing. New computing paradigms. — Heidelberg: Springer-Verlag, 1998.
39. Pixton D. Regularity of splicing languages // Discrete Appl. Math. — 1996. — V. 69. — P. 101–124.
40. Priese L. Towards a precise characterization of the complexity of the universal and nonuniversal Turing machines // SIAM J. Comput. — 1979. — V. 8, № 4. — P. 508–523.
41. Priese L., Rogojine Y., Margenstern M. Finite H-systems with 3 test tubes are not predictable // Proc. 3rd Pacific Symp. on Biocomputing. (Kapalua, Maui, January 1998, Hawaii) / Eds. R. B. Altman, A. K. Dunker, L. Hunter, T. E. Klein. — Singapure: World Sci. Publ., 1998. — P. 545–556.
42. Robinson R. M. Minsky's small universal Turing machine // International Journal of Mathematics. — 1991. — V. 2, № 5. — P. 551–562.
43. Rogers H. Theory of recursive functions and effective computability. — New York: McGraw-Hill, 1967.
44. Rogozhin Yu. About Shannon's problem for Turing machines // Computer Science Journal of Moldova. — 1993. — V. 1, № 3 (3). — P. 108–111.
45. Rogozhin Yu. Small universal Turing machines // Theoretical Computer Science. — 1996. — V. 168. № 2. — P. 215–240.

46. Rogozhin Yu. A universal Turing machine with 22 states and 2 symbols // Romanian J. of Information Science and Technology. — 1998. — V. 1, № 3. — P. 259–265.
47. Shannon C. E. A universal Turing machine with two internal states // Automata studies. Ann. of Math. Stud. V. 34. — Princeton: Princeton Univ. Press, 1956. — P. 157–165. [Имеется перевод: Шеннон К. Универсальная машина Тьюринга с двумя внутренними состояниями // Автоматы. — М.: Изд-во иностр. лит., 1956. — С. 213–225.]
48. Turing A. M. On computable numbers, with an application to the Entscheidungsproblem // Proc. London Math. Soc. Ser. 2. — 1936. — V. 42. — P. 230–265, исправления: V. 43. — P. 544–546.
49. Vatanabe S. 5-symbol 8-state and 5-symbol 6-state universal Turing machines // J. ACM. — 1964. — V. 8, № 4. — P. 476–483. [Имеется перевод: Ватанабе С. Универсальные машины Тьюринга с 5 символами и 8 состояниями и 5 символами и 6 состояниями // Кибернетич. сб. Новая серия. Вып. 6. — М.: Мир, 1969. — С. 80–90.]
50. Zandron C., Ferretti C., Mauri G. A reduced distributed splicing system for RE languages // Lecture Notes in Computer Science. — 1997. — V. 1218. — P. 319–329.

Поступило в редакцию 26 XI 1998