

# **Введение в теорию автоматов, языков и вычислений**

2-Е ИЗДАНИЕ

# Introduction to Automata Theory, Languages, and Computation

SECOND EDITION

JOHN E. HOPCROFT  
Cornell University

RAJEEV MOTWANI  
Stanford University

JEFFREY D. ULLMAN  
Stanford University



ADDISON-WESLEY PUBLISHING COMPANY

*Boston • San Francisco • New York  
London • Toronto • Sydney • Tokyo • Singapore • Madrid  
Mexico City • Munich • Paris • Cape Town • Hong Kong • Montreal*

# **Введение в теорию автоматов, языков и вычислений**

2-Е ИЗДАНИЕ

ДЖОН ХОПКРОФТ  
РАДЖИВ МОТВАНИ  
ДЖЕФФРИ УЛЬМАН



Москва • Санкт-Петербург • Киев  
2008

ББК 32.973.26-018.2.75

X78

УДК 681.3.07

Издательский дом “Вильямс”

Перевод с английского *О.И. Васылык, М. Саит-Аметова,*  
канд.физ.-мат.наук *А.Б. Ставровского*

Под редакцией канд.физ.-мат.наук *А.Б. Ставровского*

По общим вопросам обращайтесь в Издательский дом “Вильямс”  
по адресу: [info@williamspublishing.com](mailto:info@williamspublishing.com), <http://www.williamspublishing.com>

**Хопкрофт, Джон, Э., Мотвани, Раджив, Ульман, Джеффри, Д..**

X78 Введение в теорию автоматов, языков и вычислений, 2-е изд.: Пер. с англ. — М.: Издательский дом “Вильямс”, 2008. — 528 с.: ил. — Парал. тит. англ.

ISBN 978-5-8459-1347-0 (рус.)

Книга известных американских ученых посвящена теории автоматов и соответствующих формальных языков и грамматик - как регулярных, так и контекстно-свободных. Во второй части рассматриваются различные машины Тьюринга, при помощи которых формализуются понятия разрешимых и неразрешимых проблем, а также определяются функции временной и емкостной оценки сложности алгоритмов. Изложение ведется строго, но доступно, и сопровождается многочисленными примерами, а также задачами для самостоятельного решения.

Книга будет полезна читателям различных категорий - студентам, аспирантам, научным сотрудникам, преподавателям высших учебных заведений, а также всем, кто интересуется математическими основами современной вычислительной техники.

**ББК 32.973.26-018.2.75**

Все названия программных продуктов являются зарегистрированными торговыми марками соответствующих фирм.

Никакая часть настоящего издания ни в каких целях не может быть воспроизведена в какой бы то ни было форме и какими бы то ни было средствами, будь то электронные или механические, включая фотокопирование и запись на магнитный носитель, если на это нет письменного разрешения издательства Addison-Wesley Publishing Company, Inc.

Authorized translation from the English language edition published by Addison-Wesley Publishing Company, Inc, Copyright © 2001

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from the Publisher.

Russian language edition published by Williams Publishing House according to the Agreement with R&I Enterprises International, Copyright © 2008

ISBN 978-5-8459-1347-0 (рус.)

ISBN 0-201-44124-1 (англ.)

© Издательский дом “Вильямс”, 2008

© Addison-Wesley Publishing Company, Inc, 2001

# Оглавление

<b>Предисловие</b>	<b>14</b>
<b>ГЛАВА 1. Автоматы: методы и понятия</b>	<b>17</b>
<b>ГЛАВА 2. Конечные автоматы</b>	<b>53</b>
<b>ГЛАВА 3. Регулярные выражения и языки</b>	<b>101</b>
<b>ГЛАВА 4. Свойства регулярных языков</b>	<b>143</b>
<b>ГЛАВА 5. Контекстно-свободные грамматики и языки</b>	<b>185</b>
<b>ГЛАВА 6. Автоматы с магазинной памятью</b>	<b>233</b>
<b>ГЛАВА 7. Свойства контекстно-свободных языков</b>	<b>269</b>
<b>ГЛАВА 8. Введение в теорию машин Тьюринга</b>	<b>319</b>
<b>ГЛАВА 9. Неразрешимость</b>	<b>377</b>
<b>ГЛАВА 10. Труднорешаемые проблемы</b>	<b>423</b>
<b>ГЛАВА 11. Дополнительные классы проблем</b>	<b>481</b>
<b>Предметный указатель</b>	<b>523</b>

# Содержание

<b>Предисловие</b>	<b>14</b>
Как пользоваться книгой	15
Требования к уровню подготовки	15
Упражнения	16
Поддержка в World Wide Web	16
Благодарности	16
<b>ГЛАВА 1. Автоматы: методы и понятия</b>	<b>17</b>
1.1. Зачем изучается теория автоматов?	18
1.1.1. Введение в теорию конечных автоматов	18
1.1.2. Структурные представления	20
1.1.3. Автоматы и сложность	21
1.2. Введение в теорию формальных доказательств	21
1.2.1. Дедуктивные доказательства	22
1.2.2. Сведение к определениям	25
1.2.3. Другие формы теорем	27
1.2.4. Теоремы без гипотезы	30
1.3. Дополнительные схемы доказательств	30
1.3.1. Доказательства эквивалентностей, связанных с множествами	30
1.3.2. Контрапозиция	32
1.3.3. Доказательство методом “от противного”	34
1.3.4. Контрпримеры	34
1.4. Индуктивные доказательства	36
1.4.1. Индукция по целым числам	36
1.4.2. Более общие формы целочисленных индуктивных доказательств	39
1.4.3. Структурная индукция	40
1.4.4. Совместная индукция	43
1.5. Основные понятия теории автоматов	45
1.5.1. Алфавиты	46
1.5.2. Цепочки	46
1.5.3. Языки	47
1.5.4. Проблемы	48
Резюме	50
Литература	52
<b>ГЛАВА 2. Конечные автоматы</b>	<b>53</b>
2.1. Неформальное знакомство с конечными автоматами	54
2.1.1. Основные правила	54
2.1.2. Протокол	55
2.1.3. Возможность игнорирования автоматом некоторых действий	57

2.1.4. Система в целом как автомат	59
2.1.5. Проверка протокола с помощью автомата-произведения	61
2.2. Детерминированные конечные автоматы	61
2.2.1. Определение детерминированного конечного автомата	62
2.2.2. Как ДКА обрабатывает цепочки	62
2.2.3. Более простые представления ДКА	64
2.2.4. Расширение функции переходов на цепочки	65
2.2.5. Язык ДКА	68
2.2.6. Упражнения к разделу 2.2	69
2.3. Недетерминированные конечные автоматы	71
2.3.1. Неформальное описание недетерминированного конечного автомата	72
2.3.2. Определение недетерминированного конечного автомата	73
2.3.3. Расширенная функция переходов	74
2.3.4. Язык НКА	75
2.3.5. Эквивалентность детерминированных и недетерминированных конечных автоматов	77
2.3.6. Плохой случай для конструкции подмножеств	81
2.3.7. Упражнения к разделу 2.3	83
2.4. Приложение: поиск в тексте	85
2.4.1. Поиск цепочек в тексте	85
2.4.2. Недетерминированные конечные автоматы для поиска в тексте	86
2.4.3. ДКА, распознающий множество ключевых слов	87
2.4.4. Упражнения к разделу 2.4	89
2.5. Конечные автоматы с $\epsilon$ -переходами	89
2.5.1. Использование $\epsilon$ -переходов	89
2.5.2. Формальная запись $\epsilon$ -НКА	91
2.5.3. Что такое $\epsilon$ -замыкание	91
2.5.4. Расширенные переходы и языки $\epsilon$ -НКА	92
2.5.5. Устранение $\epsilon$ -переходов	94
2.5.6. Упражнения к разделу 2.5	97
Резюме	97
Литература	98

## **ГЛАВА 3. Регулярные выражения и языки** **101**

3.1. Регулярные выражения	101
3.1.1. Операторы регулярных выражений	102
3.1.2. Построение регулярных выражений	104
3.1.3. Приоритеты регулярных операторов	106
3.1.4. Упражнения к разделу 3.1	108
3.2. Конечные автоматы и регулярные выражения	108
3.2.1. От ДКА к регулярным выражениям	109
3.2.2. Преобразование ДКА в регулярное выражение методом исключения состояний	114
3.2.3. Преобразование регулярного выражения в автомат	120
3.2.4. Упражнения к разделу 3.2	124
3.3. Применение регулярных выражений	126

3.3.1. Регулярные выражения в UNIX	126
3.3.2. Лексический анализ	128
3.3.3. Поиск образцов в тексте	130
3.3.4. Упражнения к разделу 3.3	132
3.4. Алгебраические законы для регулярных выражений	132
3.4.1. Ассоциативность и коммутативность	133
3.4.2. Единичные и нулевые элементы	134
3.4.3. Дистрибутивные законы	134
3.4.4. Закон идемпотентности	135
3.4.5. Законы, связанные с оператором итерации	136
3.4.6. Установление законов для регулярных выражений	136
3.4.7. Проверка истинности алгебраических законов для регулярных выражений	139
3.4.8. Упражнения к разделу 3.4	140
Резюме	141
Литература	142
<b>ГЛАВА 4. Свойства регулярных языков</b>	<b>143</b>
4.1. Доказательство нерегулярности языков	143
4.1.1. Лемма о накачке для регулярных языков	144
4.1.2. Применение леммы о накачке	145
4.1.3. Упражнения к разделу 4.1	147
4.2. Свойства замкнутости регулярных языков	148
4.2.1. Замкнутость регулярных языков относительно булевых операций	149
4.2.2. Обращение	154
4.2.3. Гомоморфизмы	156
4.2.4. Обратный гомоморфизм	157
4.2.5. Упражнения к разделу 4.2	163
4.3. Свойства разрешимости регулярных языков	166
4.3.1. Преобразования различных представлений языков	167
4.3.2. Проверка пустоты регулярных языков	169
4.3.3. Проверка принадлежности регулярному языку	170
4.3.4. Упражнения к разделу 4.3	171
4.4. Эквивалентность и минимизация автоматов	171
4.4.1. Проверка эквивалентности состояний	172
4.4.2. Проверка эквивалентности регулярных языков	175
4.4.3. Минимизация ДКА	177
4.4.4. Почему минимизированный ДКА невозможно улучшить	180
4.4.5. Упражнения к разделу 4.4	182
Резюме	183
Литература	183
<b>ГЛАВА 5. Контекстно-свободные грамматики и языки</b>	<b>185</b>
5.1. Контекстно-свободные грамматики	185
5.1.1. Неформальный пример	185
5.1.2. Определение контекстно-свободных грамматик	187
5.1.3. Порождения с использованием грамматики	189



5.1.4. Левые и правые порождения	191
5.1.5. Язык, задаваемый грамматикой	193
5.1.6. Выводимые цепочки	194
5.1.7. Упражнения к разделу 5.1	195
5.2. Деревья разбора	197
5.2.1. Построение деревьев разбора	197
5.2.2. Крона дерева разбора	199
5.2.3. Вывод, порождение и деревья разбора	200
5.2.4. От выводов к деревьям разбора	201
5.2.5. От деревьев к порождениям	202
5.2.6. От порождений к рекурсивным выводам	205
5.2.7. Упражнения к разделу 5.2	207
5.3. Приложения контекстно-свободных грамматик	207
5.3.1. Синтаксические анализаторы	208
5.3.2. Генератор синтаксических анализаторов YACC	210
5.3.3. Языки описания документов	211
5.3.4. XML и определения типа документа	213
5.3.5. Упражнения к разделу 5.3	219
5.4. Неоднозначность в грамматиках и языках	220
5.4.1. Неоднозначные грамматики	220
5.4.2. Исключение неоднозначности из грамматик	222
5.4.3. Левые порождения как способ выражения неоднозначности	225
5.4.4. Существенная неоднозначность	226
5.4.5. Упражнения к разделу 5.4	228
Резюме	229
Литература	230

## **ГЛАВА 6. Автоматы с магазинной памятью 233**

6.1. Определение автоматов с магазинной памятью	233
6.1.1. Неформальное введение	233
6.1.2. Формальное определение автомата с магазинной памятью	235
6.1.3. Графическое представление МП-автоматов	237
6.1.4. Конфигурации МП-автомата	238
6.1.5. Упражнения к разделу 6.1	241
6.2. Языки МП-автоматов	242
6.2.1. Допустимость по заключительному состоянию	242
6.2.2. Допустимость по пустому магазину	244
6.2.3. От пустого магазина к заключительному состоянию	244
6.2.4. От заключительного состояния к пустому магазину	247
6.2.5. Упражнения к разделу 6.2	249
6.3. Эквивалентность МП-автоматов и КС-грамматик	251
6.3.1. От грамматик к МП-автоматам	251
6.3.2. От МП-автоматов к грамматикам	255
6.3.3. Упражнения к разделу 6.3	259
6.4. Детерминированные автоматы с магазинной памятью	260
6.4.1. Определение детерминированного МП-автомата	260
6.4.2. Регулярные языки и детерминированные МП-автоматы	261

6.4.3. Детерминированные МП-автоматы и КС-языки	262
6.4.4. Детерминированные МП-автоматы и неоднозначные грамматики	263
6.4.5. Упражнения к разделу 6.4	264
Резюме	265
Литература	266
<b>ГЛАВА 7. Свойства контекстно-свободных языков</b>	<b>269</b>
7.1. Нормальные формы контекстно-свободных грамматик	269
7.1.1. Удаление бесполезных символов	269
7.1.2. Вычисление порождающих и достижимых символов	271
7.1.3. Удаление $\epsilon$ -продукций	273
7.1.4. Удаление цепных продукций	276
7.1.5. Нормальная форма Хомского	280
7.1.6. Упражнения к разделу 7.1	284
7.2. Лемма о накачке для контекстно-свободных языков	287
7.2.1. Размер деревьев разбора	287
7.2.2. Утверждение леммы о накачке	288
7.2.3. Приложения леммы о накачке к КС-языкам	290
7.2.4. Упражнения к разделу 7.2	293
7.3. Свойства замкнутости контекстно-свободных языков	295
7.3.1. Подстановки	295
7.3.2. Приложения теоремы о подстановке	297
7.3.3. Обращение	298
7.3.4. Пересечение с регулярным языком	298
7.3.5. Обратный гомоморфизм	302
7.3.6. Упражнения к разделу 7.3	304
7.4. Свойства разрешимости КС-языков	306
7.4.1. Сложность взаимных преобразований КС-грамматик и МП-автоматов	306
7.4.2. Временная сложность преобразования к нормальной форме Хомского	308
7.4.3. Проверка пустоты КС-языков	309
7.4.4. Проверка принадлежности КС-языку	311
7.4.5. Обзор неразрешимых проблем КС-языков	314
7.4.6. Упражнения к разделу 7.4	315
Резюме	316
Литература	317
<b>ГЛАВА 8. Введение в теорию машин Тьюринга</b>	<b>319</b>
8.1. Задачи, не решаемые компьютерами	319
8.1.1. Программы печати "Hello, world"	320
8.1.2. Гипотетическая программа проверки приветствия мира	322
8.1.3. Сведение одной проблемы к другой	325
8.1.4. Упражнения к разделу 8.1	328
8.2. Машина Тьюринга	328
8.2.1. Поиски решения всех математических вопросов	329
8.2.2. Описание машин Тьюринга	330

8.2.3. Конфигурации машин Тьюринга	331
8.2.4. Диаграммы переходов для машин Тьюринга	334
8.2.5. Язык машины Тьюринга	337
8.2.6. Машины Тьюринга и останов	338
8.2.7. Упражнения к разделу 8.2	339
8.3. Техника программирования машин Тьюринга	340
8.3.1. Память в состоянии	340
8.3.2. Многодорожечные ленты	342
8.3.3. Подпрограммы	344
8.3.4. Упражнения к разделу 8.3	346
8.4. Расширения базовой машины Тьюринга	346
8.4.1. Многоленточные машины Тьюринга	347
8.4.2. Эквивалентность одноленточных и многоленточных машин Тьюринга	348
8.4.3. Время работы и конструкция “много лент к одной”	349
8.4.4. Недетерминированные машины Тьюринга	351
8.4.5. Упражнения к разделу 8.4	353
8.5. Машины Тьюринга с ограничениями	355
8.5.1. Машины Тьюринга с односторонними лентами	356
8.5.2. Мультистековые машины	358
8.5.3. Счетчиковые машины	361
8.5.4. Мощность счетчиковых машин	362
8.5.5. Упражнения к разделу 8.5	364
8.6. Машины Тьюринга и компьютеры	365
8.6.1. Имитация машины Тьюринга на компьютере	365
8.6.2. Имитация компьютера на машине Тьюринга	366
8.6.3. Сравнение времени работы компьютеров и машин Тьюринга	371
Резюме	374
Литература	376

## **ГЛАВА 9. Неразрешимость** **377**

9.1. Неперечислимый язык	378
9.1.1. Перечисление двоичных цепочек	378
9.1.2. Коды машин Тьюринга	379
9.1.3. Язык диагонализации	380
9.1.4. Доказательство неперечислимости $L_d$	381
9.1.5. Упражнения к разделу 9.1	382
9.2. Неразрешимая РП-проблема	382
9.2.1. Рекурсивные языки	383
9.2.2. Дополнения рекурсивных и РП-языков	385
9.2.3. Универсальный язык	387
9.2.4. Неразрешимость универсального языка	389
9.2.5. Упражнения к разделу 9.2	390
9.3. Неразрешимые проблемы, связанные с машинами Тьюринга	392
9.3.1. Сведения	392
9.3.2. Машины Тьюринга, допускающие пустой язык	394
9.3.3. Теорема Райса и свойства РП-языков	397

9.3.4. Проблемы, связанные с описаниями языков в виде машин Тьюринга	399
9.3.5. Упражнения к разделу 9.3	400
9.4. Проблема соответствий Поста	401
9.4.1. Определение проблемы соответствий Поста	402
9.4.2. “Модифицированная” ПСП	404
9.4.3. Завершение доказательства неразрешимости ПСП	407
9.4.4. Упражнения к разделу 9.4	412
9.5. Другие неразрешимые проблемы	413
9.5.1. Проблемы, связанные с программами	413
9.5.2. Неразрешимость проблемы неоднозначности КС-грамматик	413
9.5.3. Дополнение языка списка	416
9.5.4. Упражнения к разделу 9.5	418
9.6. Резюме	420
9.7. Литература	421

## **ГЛАВА 10. Труднорешаемые проблемы** **423**

10.1. Классы $\mathcal{P}$ и $\mathcal{NP}$	424
10.1.1. Проблемы, разрешимые за полиномиальное время	424
10.1.2. Пример: алгоритм Крускала	424
10.1.3. Недетерминированное полиномиальное время	429
10.1.4. Пример из $\mathcal{NP}$ : проблема коммивояжера	429
10.1.5. Полиномиальные сведения	431
10.1.6. NP-полные проблемы	432
10.1.7. Упражнения к разделу 10.1	434
10.2. Первая NP-полная проблема	436
10.2.1. Проблема выполнимости	436
10.2.2. Представление экземпляров ВВП	438
10.2.3. NP-полнота проблемы ВВП	439
10.2.4. Упражнения к разделу 10.2	445
10.3. Ограниченная проблема выполнимости	445
10.3.1. Нормальные формы булевых выражений	446
10.3.2. Преобразование формул в КНФ	447
10.3.3. NP-полнота проблемы ВКНФ	450
10.3.4. NP-полнота проблемы 3-выполнимости	455
10.3.5. Упражнения к разделу 10.3	456
10.4. Еще несколько NP-полных проблем	457
10.4.1. Описание NP-полных проблем	458
10.4.2. Проблема независимого множества	458
10.4.3. Проблема узельного покрытия	462
10.4.4. Проблема ориентированного гамильтонова цикла	464
10.4.5. Неориентированные гамильтоновы циклы и ПКМ	470
10.4.6. Вывод относительно NP-полных проблем	472
10.4.7. Упражнения к разделу 10.4	473
10.5. Резюме	477
10.6. Литература	478

<b>ГЛАВА 11. Дополнительные классы проблем</b>	<b>481</b>
11.1. Дополнения языков из $\mathcal{NP}$	482
11.1.1. Класс языков $\text{co-}\mathcal{NP}$	482
11.1.2. $\mathcal{NP}$ -полные проблемы и $\text{co-}\mathcal{NP}$	483
11.1.3. Упражнения к разделу 11.1	484
11.2. Проблемы, разрешимые в полиномиальном пространстве	485
11.2.1. Машины Тьюринга с полиномиальным пространством	485
11.2.2. Связь $\mathcal{PS}$ и $\mathcal{NP}$ с определенными ранее классами	486
11.2.3. Детерминированное и недетерминированное полиномиальное пространство	487
11.3. Проблема, полная для $\mathcal{PS}$	489
11.3.1. $\mathcal{PS}$ -полнота	490
11.3.2. Булевы формулы с кванторами	491
11.3.3. Вычисление булевых формул с кванторами	492
11.3.4. $\mathcal{PS}$ -полнота проблемы КБФ	494
11.3.5. Упражнения к разделу 11.3	498
11.4. Классы языков, основанные на рандомизации	499
11.4.1. Быстрая сортировка — пример рандомизированного алгоритма	499
11.4.2. Вариант машины Тьюринга с использованием рандомизации	500
11.4.3. Язык рандомизированной машины Тьюринга	502
11.4.4. Класс $\mathcal{RP}$	504
11.4.5. Распознавание языков из $\mathcal{RP}$	506
11.4.6. Класс $\mathcal{ZPP}$	506
11.4.7. Соотношение между $\mathcal{RP}$ и $\mathcal{ZPP}$	507
11.4.8. Соотношения с классами $\mathcal{P}$ и $\mathcal{NP}$	509
11.5. Сложность проверки простоты	509
11.5.1. Важность проверки пустоты	510
11.5.2. Введение в модулярную арифметику	512
11.5.3. Сложность вычислений в модулярной арифметике	514
11.5.4. Рандомизированная полиномиальная проверка простоты	515
11.5.5. Недетерминированные проверки простоты	516
11.5.6. Упражнения к разделу 11.5	519
Резюме	520
Литература	521
<b>Предметный указатель</b>	<b>523</b>

# Предисловие

В предисловии к своей книге 1979 года, предшествовавшей данному изданию, Дж. Хопкрофт и Дж. Ульман с удивлением отмечали, что за время, прошедшее после выхода их первой книги в 1969 году, произошел взрыв в развитии теории автоматов. Действительно, книга, вышедшая в 1979 году, содержала множество тем, не затронутых в предыдущей работе, и по объему была почти вдвое больше. Сравнив эту книгу с книгой 1979 года, вы увидите, что она, как автомобили 1970-х “больше снаружи, но меньше изнутри”. И хотя это может показаться шагом назад, мы считаем такие изменения целесообразными в силу ряда причин.

Во-первых, в 1979 году теория автоматов и языков еще активно развивалась, и одной из целей той книги было пробудить интерес математически одаренных студентов к исследованиям в этой области. На сегодня в теории автоматов имеется лишь узкое направление для исследований (чего не скажешь о ее приложениях). Поэтому, на наш взгляд, не имеет смысла сохранять здесь лаконичный, сугубо математический стиль книги 1979 года.

Во-вторых, за последние двадцать лет изменилась роль теории автоматов и языков. В 1979 году это был сложный предмет, требующий от читателя высокого уровня подготовки. Читатель нашей книги, в особенности последних ее глав, представлялся нам хорошо подготовленным студентом-старшекурсником. Сегодня же этот предмет входит в стандартную вузовскую программу для младших курсов. Поэтому содержание книги должно быть по возможности доступным, а следовательно, содержать больше подробных доказательств и обоснований по сравнению с предыдущей книгой.

В-третьих, за два последних десятилетия информатика (Computer Science) невообразимо разрослась. И если в 1979 году вузовскую программу приходилось искусственно заполнять предметами, которые, как нам казалось, могли послужить новой волне развития технологий, то сегодня таких дисциплин уже слишком много.

В-четвертых, за эти годы информатика стала практическим предметом, и многие изучающие ее студенты настроены прагматически. Мы по-прежнему убеждены, что теория автоматов является мощным инструментом для исследований во множестве новых областей. А упражнения теоретического, развивающего плана, которые содержит обычный курс теории автоматов, сохраняют свое значение вне зависимости от того, предпочитает ли студент изучать лишь практические, “денежные” технологии. Однако для того, чтобы обеспечить данному предмету достойное место среди прочих дисциплин, изучаемых студентами-информатиками, нам представляется необходимым, наряду с математической частью, подчеркнуть и практические приложения теории. С этой целью мы заменили часть довольно сложных для понимания тем из предыдущей книги актуальными примерами применения этих идей на практике. В то время как приложения теории автоматов

и языков для компиляторов сегодня настолько понятны и естественны, что входят во всякий стандартный курс по компиляторам, существует немало и совершенно новых приложений. Например, алгоритмы проверки моделей, используемые для верификации протоколов, и языки описания документов, построенные по образцу контекстно-свободных грамматик.

И, наконец, последнее объяснение одновременного увеличения объема книги и уменьшения ее фактического содержания состоит в том, что при ее верстке использовались все преимущества издательских систем  $\text{T}_\text{E}\text{X}$  и  $\text{L}^\text{A}\text{T}_\text{E}\text{X}$ , которые поощряют “открытый” стиль печати, благодаря чему книга увеличивается в объеме, но становится удобнее для чтения. Мы с благодарностью отмечаем труд создателей этих издательских систем Дона Кнута и Леса Лампорта.

## **Как пользоваться книгой**

Эта книга рассчитана на полусеместровый или семестровый курс лекций для студентов второго года обучения и выше. На ее основе в Стэнфордском университете Радживом и Джеффом читается полусеместровый курс (CS154) по теории автоматов и языков. Ввиду ограниченности по времени в этом курсе не охвачены глава 11 и часть материала главы 10, например, довольно сложные вопросы о полиномиально-временной сводимости. На Web-сайте книги (см. ниже) помещено несколько сокращенных вариантов этого курса с замечаниями.

Несколько лет назад мы столкнулись с тем, что многие студенты, поступившие в Стэнфорд после окончания колледжа, прошли курс теории автоматов, не содержащий теории сложности. Преподавательский состав Стэнфорда полагает, что для всякого, кто серьезно занимается информатикой, эти идеи чрезвычайно важны. Тут недостаточно знать лишь то, что “для решения NP-полной задачи требуется очень много времени”. Специально для таких студентов был разработан курс лекций (CS154N), который содержит материал только 8, 9 и 10 глав. Фактически, для того, чтобы освоить курс CS154N, они прослушивают лишь последнюю треть курса CS154. И по сегодняшний день на каждом курсе находится несколько студентов, желающих заниматься именно таким образом. Мы рекомендуем этот подход, поскольку он не требует чрезмерных усилий.

## **Требования к уровню подготовки**

Чтение этой книги не вызовет затруднений у студентов, освоивших основы дискретной математики, в том числе изучивших графы, деревья, логику и методы доказательств. Кроме того, мы предполагаем, что читатель в достаточной степени знаком с программированием и, в частности, имеет представление об общих структурах данных, рекурсии и роли таких главных системных компонентов, как компиляторы. Эта сумма знаний соответствует стандартной программе первых двух лет обучения для студентов, изучающих информатику.

## Упражнения

Книга содержит большое число упражнений, по несколько почти в каждом разделе. Упражнения или части упражнений повышенной сложности отмечены восклицательным знаком. Наиболее трудные из них отмечены двумя восклицательными знаками.

Некоторые упражнения отмечены звездочкой. Их решения мы намерены поместить на Web-странице книги. Они предназначены для самопроверки и доступны широкой аудитории. Отметим, что в некоторых случаях в одном упражнении *B* требуется определенным образом изменить или адаптировать ваше решение другого упражнения *A*. Если какая-то часть упражнения *A* имеет решение, то естественно ожидать, что соответствующая часть упражнения *B* также имеет решение.

## Поддержка в World Wide Web

Адрес книги в Internet:

<http://www-db.stanford.edu/~ullman/ialc.html>

Здесь вы найдете решения заданий, отмеченных звездочкой, список замеченных опечаток и некоторые вспомогательные материалы. По мере чтения лекций по курсу CS154 мы постараемся размещать тут наши замечания по поводу домашних заданий, решений упражнений и экзаменов.

## Благодарности

На часть материала главы 1 повлияла рукопись Крейга Сильверштейна (Craig Silverstein) о том, “как строить доказательства”. Мы благодарим также следующих лиц: Зое Абрамс (Zoe Abrams), Джордж Кандеа (George Candea), Хаовен Чен (Haowen Chen), Бьен-Ган Чан (Byong-Gun Chun), Джеффри Шаллит (Jeffrey Shallit), Брет Тейлор (Bret Taylor), Джейсон Таунсенд (Jason Townsend) и Эрик Узурео (Erik Uzureau), которые высказали свои замечания и указали на ряд опечаток в черновом варианте книги. Ошибки, оставшиеся незамеченными, авторы безусловно относят на свой счет.

Джон Хопкрофт  
Раджив Мотвани  
Джеффри Ульман  
Итака, Нью-Йорк и Стэнфорд, Калифорния  
Сентябрь, 2000.



# ГЛАВА 1

## Автоматы: методы и понятия

Теория автоматов занимается изучением абстрактных вычислительных устройств, или “машин”. В 1930-е годы, задолго до появления компьютеров, А. Тьюринг исследовал абстрактную машину, которая, по крайней мере в области вычислений, обладала всеми возможностями современных вычислительных машин. Целью Тьюринга было точно описать границу между тем, что вычислительная машина может делать, и тем, чего она не может. Полученные им результаты применимы не только к абстрактным *машинам Тьюринга*, но и к реальным современным компьютерам.

В 1940-х и 1950-х годах немало исследователей занимались изучением простейших машин, которые сегодня называются “конечными автоматами”. Такие автоматы вначале были предложены в качестве модели функционирования человеческого мозга. Однако вскоре они оказались весьма полезными для множества других целей, которые будут упомянуты в разделе 1.1. А в конце 1950-х лингвист Н. Хомский занялся изучением формальных “грамматик”. Не будучи машинами в точном смысле слова, грамматики, тем не менее, тесно связаны с абстрактными автоматами и служат основой некоторых важнейших составляющих программного обеспечения, в частности, компонентов компиляторов.

В 1969 году С. Кук развил результаты Тьюринга о вычислимости и невычислимости. Ему удалось разделить задачи на те, которые могут быть эффективно решены вычислительной машиной, и те, которые, в принципе, могут быть решены, но требуют для этого так много машинного времени, что компьютер оказывается бесполезным для решения практически всех экземпляров задачи, за исключением небольших. Задачи последнего класса называют “трудно разрешимыми” (“труднорешаемыми”) или “NP-трудными”. Даже при экспоненциальном росте быстродействия вычислительных машин (“закон Мура”) весьма маловероятно, что нам удастся достигнуть значительных успехов в решении задач этого класса.

Все эти теоретические построения непосредственно связаны с тем, чем занимаются ученые в области информатики сегодня. Некоторые из введенных понятий, такие, например, как конечные автоматы и некоторые типы формальных грамматик, используются при проектировании и создании важных компонентов программного обеспечения. Другие понятия, например, машина Тьюринга, помогают нам уяснить принципиальные возможности программного обеспечения. В частности, теория сложности вычислений

позволяет определить, можем ли мы решить ту или иную задачу “в лоб” и написать соответствующую программу для ее решения (если эта задача не принадлежит классу “трудно разрешимых”), или же нам следует искать решение данной трудно разрешимой задачи в обход, используя приближенный, эвристический, или какой-либо другой метод, с помощью которого удастся ограничить время, затрачиваемое программой на ее решение.

В этой вводной главе дается обзор содержания теории автоматов и ее приложений. Основная часть главы посвящена рассмотрению различных методов доказательств и способов их нахождения. Рассматриваются дедуктивные доказательства, утверждения, получаемые переформулировкой, доказательства от противного, доказательства по индукции и другие важные понятия. В последней части вводится ряд основополагающих понятий теории автоматов: алфавиты, цепочки и языки.

## **1.1. Зачем изучается теория автоматов?**

В силу ряда причин теория автоматов и теория сложности являются важнейшей частью ядра информатики. В этом разделе мы познакомим читателя с главными мотивами, побуждающими нас к их изучению, а также обрисует в общих чертах основные темы, затрагиваемые в книге.

### **1.1.1. Введение в теорию конечных автоматов**

Конечные автоматы являются моделью для многих компонентов аппаратного и программного обеспечения. Ниже (начиная со второй главы) мы рассмотрим примеры их использования, а сейчас просто перечислим наиболее важные из них.

1. Программное обеспечение, используемое для разработки и проверки цифровых схем.
2. “Лексический анализатор” стандартного компилятора, т.е. тот компонент компилятора, который отвечает за разбивку исходного текста на такие логические единицы, как идентификаторы, ключевые слова и знаки пунктуации.
3. Программное обеспечение для сканирования таких больших текстовых массивов, как наборы Web-страниц, с целью поиска заданных слов, фраз или других последовательностей символов (шаблонов).
4. Программное обеспечение для проверки различного рода систем (протоколы связи или протоколы для защищенного обмена информацией), которые могут находиться в конечном числе различных состояний.

С точными определениями автоматов различного типа мы встретимся вскоре. А начнем с краткого описания того, что представляет собой конечный автомат, и как он действует. Существует множество различных систем и их компонентов, например,

только что перечисленные, которые можно рассматривать, как находящиеся в каждый момент времени в одном из конечного числа “состояний”. Назначением каждого состояния является запоминание определенного момента истории системы. Поскольку число состояний конечно, то запомнить всю историю системы невозможно, а значит, необходимо строить систему тщательно, чтобы хранить только действительно важную информацию и забывать несущественную. Преимущество конечности числа состояний заключается в том, что систему можно реализовать, имея ограниченные ресурсы. Например, ее можно реализовать “в железе” как схему или же в виде простой программы, принимающей решения на основе ограниченного количества данных или текущей команды машинного кода.

**Пример 1.1.** Простейшим нетривиальным конечным автоматом является переключатель “включено-выключено”. Это устройство помнит свое текущее состояние, и от этого состояния зависит результат нажатия кнопки. Из состояния “выключено” нажатие кнопки переводит переключатель в состояние “включено”, и наоборот.

На рис. 1.1 представлена конечноавтоматная модель переключателя. Как и для всех конечных автоматов, состояния обозначены кружками. В данном случае они отмечены как “*вкл.*” и “*выкл.*”. Дуги между состояниями отмечены “входными символами”, задающими внешние воздействия на систему. В нашем случае это *Нажатие*, что означает нажатие на кнопку переключателя. Стрелки указывают, что всякий раз при *Нажатии* система переходит из одного состояния в другое.

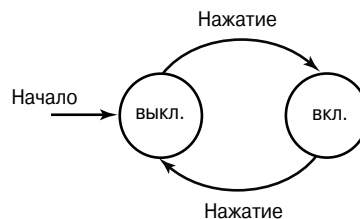


Рис. 1.1. Конечный автомат, моделирующий переключатель

Одно из состояний является так называемым “начальным состоянием”. Это состояние (в нашем случае — “*выкл.*”), в котором система находится изначально. На рисунке оно отмечено словом *Начало* и стрелкой, указывающей на это состояние.

Часто необходимо выделять одно или несколько “заключительных”, или “допускающих”, состояний. Попав в одно из них в результате реализации некоторой последовательности входных воздействий, мы можем считать такую входную последовательность в определенном смысле “хорошей”. Так, например, состояние “*вкл.*” на рис. 1.1 можно рассматривать как допускающее, поскольку если переключатель находится в этом состоянии, то устройство, управляемое им, находится в рабочем режиме. Допускающие состояния принято обозначать двойным кружком, хотя мы не использовали это обозначение на рис 1.1. □

**Пример 1.2.** Иногда состояние автомата запоминает гораздо более сложную информацию, чем выбор “вкл.–выкл.”. На рис. 1.2 представлен конечный автомат, который может служить частью лексического анализатора. Его функцией является распознавание ключевого слова `then`. Этот автомат должен иметь пять различных состояний, каждое из которых представляет позицию в слове `then`, достигнутую на данный момент. Эти позиции соответствуют префиксам слова, от пустой цепочки (никакая позиция в слове еще не достигнута) до целого слова.

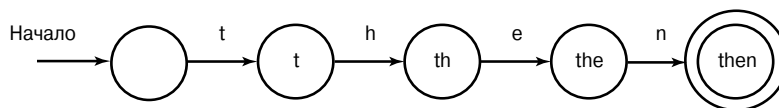


Рис. 1.2. Конечный автомат, моделирующий распознавание слова `then`

На рис. 1.2 каждое из пяти состояний обозначено частью слова, прочитанной на данном шаге. Входным сигналам соответствуют буквы. Мы можем считать, что данный лексический анализатор всякий раз просматривает по одному символу компилируемой программы. Каждый следующий символ рассматривается как входной сигнал для данного автомата. Начальное состояние автомата соответствует пустой цепочке, и каждое состояние имеет переход по очередной букве слова `then` в состояние, соответствующее следующему префиксу. Состояние, обозначенное словом “`then`”, достигается, когда по буквам введено все данное слово. Поскольку функция автомата заключается в распознавании слова `then`, то последнее состояние будем считать единственным допускающим. □

### 1.1.2. Структурные представления

Следующие системы записи не являются автоматными, но играют важную роль в теории автоматов и ее приложениях.

1. *Грамматика*. Они являются полезными моделями при проектировании программного обеспечения, обрабатывающего данные рекурсивной структуры. Наиболее известный пример — “синтаксический анализатор”. Этот компонент компилятора работает с такими рекурсивно вложенными конструкциями в типичных языках программирования, как выражения: арифметические, условные и т.п. Например, грамматическое правило типа  $E \Rightarrow E + E$  означает, что выражение может быть получено соединением любых двух выражений с помощью знака “плюс”. Это типичное правило построения выражений в существующих языках программирования. Ниже, в главе 5, будут определены так называемые контекстно-свободные грамматики.
2. *Регулярные выражения*. Они также задают структуру данных, в частности, текстовых цепочек. Как будет показано в главе 3, шаблоны описываемых ими цепочек представляют собой то же самое, что задают конечные автоматы. Стиль этих выражений суще-