ББК 32.973.26-018.2.75
С50
УДК 681.3.07

Издательский дом "Вильямс"

Зав. редакцией *С.Н. Тригуб*

Перевод с английского *Р.Г. Имамутдиновой*, канд. физ.-мат. наук *А.А. Минько*

Под редакцией канд. физ.-мат. наук *А.А. Минько*

По общим вопросам обращайтесь в Издательский дом "Вильямс" по адресу:
info@williamspublishing.com, http://www.williamspublishing.com
115419, Москва, а/я 783; 03150, Киев, а/я 152

Смит, Билл.

С50 Методы и алгоритмы вычислений на строках. : Пер. с англ. — М. : ООО "И.Д. Вильямс", 2006. — 496 с. : ил. — Парал. тит. англ.

ISBN 5–8459–1081–1 (рус.)

Книга представляет собой фундаментальное введение в алгоритмы и методы, эффективно вычисляющие паттерны в строковых последовательностях. Речь идет об общих алгоритмах и методах, которые находят применение во многих областях науки и информационных технологий: сжатие данных, криптография, распознавание речи и компьютерное зрение, вычислительная геометрия и молекулярная биология. Рассмотренные в книге алгоритмы предназначены для нахождения в строковых последовательностях определенных типов паттернов — частных, характеристических и внутренних. Каждому типу паттернов посвящена соответствующая часть книги. Книга отличается последовательным изложением материала, большим количеством иллюстративных примеров, свободным обсуждением текущих исследований в этой области, содержит более 500 упражнений, поясняющих и расширяющих материал, изложенный в тексте книги.

Книга предназначена для тех, кто имеет достаточную подготовку в математике и информатике и хочет познакомиться с этой интересной и важной областью. Материал книги доступный для студентов старших курсов и аспирантов соответствующих специальностей.

ББК 32.973.26–018.2.75

ISBN 5–8459–1081–1 (рус.)
ISBN 0–201–39839–7 (англ.)

ОГЛАВЛЕНИЕ

СОДЕРЖАНИЕ

Предисловие

В начале было Слово,
и Слово было у Бога,
и Слово было Бог.

— Иоанн 1.1

Вычисление паттернов в строковых последовательностях — это фундаментальная проблема, которая возникает во многих областях науки и информационных технологий. Манипулирование текстом в текстовых редакторах, лексический анализ компьютерных программ, работа конечных автоматов, извлечение информации из баз данных — это малая часть тех процессов, которые требуют нахождения или вычисления паттернов. Алгоритмы вычисления паттернов находят применение в таких областях, как сжатие данных, криптография, распознавание речи и компьютерное зрение, вычислительная геометрия и молекулярная биология. Тема вычисления паттернов в строковых последовательностях важна не только из-за своего практического применения. Она является частью комбинаторики, где, как известно, существует много просто формулируемых задач, для которых, однако, очень сложно найти решение, и интерпретация таких задач, как вычисление паттернов, часто позволяет найти элегантное и точное их решение.

В этой связи вызывает большое удивление, что академические факультеты математики или компьютерных наук в своем большинстве не включают в магистерские курсы или в курсы для аспирантов эту интересную, важную и сложную для исследований тему. Еще более удивительно, что существует всего несколько книг и обзоров, где собраны вместе и основополагающие теоретические результаты, и практические алгоритмы, которые появились в последнюю четверть прошедше-

го столетия. Я знаю всего пять книг [214, 67, 108, 206, 61] и три объемных обзора научных статей [21, 2, 191], содержимое которых значительно перекрывает материал данной книги. Обзорные статьи и книги [214, 67] написаны в большей мере как обобщение итогов исследования авторов, нежели как книги для студентов. Книги [108, 206] касаются, в основном, применения строковых алгоритмов в молекулярной биологии. Последняя монография [61], написанная легко и элегантно, сочетает в себе как монографию, так и руководство по строковым алгоритмам. К сожалению, в настоящее время она доступна только на французском языке.

Данная книга ставит целью заполнить этот пробел: дать общее введение в алгоритмы вычисления паттернов, которое было бы полезно в качестве отправного пункта для научных исследований и давало бы серьезный фактический материал, доступный для изучения студентам старших курсов и аспирантам. Задержимся на некоторое время на трех словах из предыдущего предложения: "доступный", "алгоритм" и "паттерн".

Основное "свойство" этой книги — сделать материал *доступным* для студентов старших курсов и аспирантов, имеющих специализацию по математике или компьютерным наукам и которые знакомы с дискретными структурами и алгоритмами, оперирующими этими структурами.

Первым условием доступности материала книги является достаточная математическая подготовка читателя, а не его знакомство со строковыми последовательностями. Предполагается, что студент прослушал стандартный курс дискретной математики, курсы по структурам данных и анализу алгоритмов. В этом случае он знает, что такое стеки, очереди, связанные списки и массивы, имеет понятие об анализе алгоритмов и о том, как записать "асимптотическую сложность" алгоритма, имеет некоторый опыт работы с математическими утверждениями и методами, используемыми для доказательства корректности алгоритмов, также знаком с алгоритмами, выполняемыми на графах и деревьях. В дополнение к перечисленному, предполагается, что читатель знаком с какими-либо языками программирования и способен читать и понимать алгоритмы, записанные на этих языках.

Второе условие не связано с компетентностью и подготовленностью читателя: моя цель — "соблазнить" студента и читателя интереснейшей и увлекательной областью новых знаний. Я не собирался писать энциклопедию алгоритмов, вычисляющих паттерны в строковых последовательностях. Вместе с тем я хотел представить результаты, которые (я надеюсь) важны и которые можно обобщить и расширить. Но это неизбежно ведет к тому, что некоторые интересные результаты были опущены (нельзя охватить необъятное!). И все-таки я надеюсь, что выбранный подход к изложению материала будет стимулировать читателя и далее к самостоятельному изучению им научной литературы.

Особым "субъектом" материала этой книги является математический объект, который в компьютерных науках называется "строка" (string) или "строковая последовательность" (в Европе, в среде математиков, более распространен термин

"слово"). Но основное внимание в книге уделяется *алгоритмам*, т.е. точным методам и процедурам, предназначенным для выполнения "чего-то". Исходя из этого данную книгу скорее можно отнести к книгам по компьютерным наукам, чем к математическим книгам. Поэтому она значительно отличается от классической монографии [164], посвященной этой теме, и ее "потомков" [162, 163], ставящих во главу угла математические аспекты данной темы. Нас в первую очередь будут интересовать алгоритмы, находящие в строковых последовательностях различного рода паттерны, и только во вторую очередь — математические свойства самих строк. Это, конечно, не означает, что математические результаты не будут представлены строго и последовательно. Это означает, что будут представлены только те математические результаты, которые необходимы для пояснения построения и поведения алгоритмов. И последнее замечание: я сознательно ограничился изложением последовательных алгоритмов для обработки одномерных строк, не делая ссылок на обширную литературу по алгоритмам с распараллеливанием процесса вычисления или на быстро растущую литературу по многомерным (особенно, двухмерным) строкам.

Другой основной "герой" нашей книги — это *паттерн*. Все рассмотренные в книге алгоритмы предназначены для нахождения в строковых последовательностях определенных типов паттернов. Я говорю "определенных типов", поскольку будем различать три основных типа паттернов — частные, характеристические и внутренние. Каждому типу паттернов посвящена соответствующая часть книги.

Частный паттерн (specific pattern) — это единственный вид паттернов, который можно задать в виде списка символов в нужном порядке. Например, в строке $\boldsymbol{x} = abaababaabaab$ мы можем найти (трижды) паттерн $\boldsymbol{u} = abaab$, но не найдем паттерн $\boldsymbol{u} = ababab$. (Иногда паттерн может содержать специальные "символы замещения", и в этом случае возможно только "приближенное" (в некотором точно определенном смысле) сравнение паттерна и строки.)

Характеристические паттерны (generic patterns) основаны на специальных представлениях структурной информации о строковых последовательностях. Например, мы можем говорить о "повторениях" в строке $\boldsymbol{x}$ — в этом случае в строке $\boldsymbol{x}$ есть несколько смежных одинаковых подстрок. (Например, в приведенной выше строке $\boldsymbol{x}$ присутствуют повторяющиеся подстроки $(aba)(aba)$, $(abaab)(abaab)$, aa (три отдельные серии) и несколько других, если вы сможете их найти.)

Последний тип паттернов, которые будут рассмотрены в книге, я назвал *внутренними* (intrinsic). Эти паттерны отображают внутреннюю структуру строковых последовательностей. Мы рассмотрим различные паттерны, которые показывают наличие периодических структур в строках, например нормальную форму, дерево суффиксов, линдонскую декомпозицию, s-факторизацию. Эти паттерны вездесущие: они используются почти во всех алгоритмах вычисления частных и характеристических паттернов. Другими словами, они формируют основу для эффективных процедур обработки строковых последовательностей. Раз-

нообразие внутренних паттернов поразительно: для строки $\boldsymbol{x}$ нашего примера нормальная форма имеет вид $(abaababa)(abaab)$, тогда как линдонскую декомпозицию можно записать как $(ab)(aabab)(aab)(aab)$, а s-факторизацию — как $(a)(b)(a)(aba)(baaba)(ab)$, и все эти паттерны полезны и эффективны с вычислительной точки зрения.

Эта книга имеет следующую организацию. В части I приведены основные сведения о строковых последовательностях и алгоритмах обработки строк. Здесь даны терминология, формы записи и основные свойства строковых последовательностей. Глава 2 является ключом к остальной части книги: здесь четко поставлены задачи, алгоритмы для решения которых описаны в последующих частях книги. На основе материала этой главы читатель может выбрать для себя направление дальнейшего чтения в виде тех глав, которые представляют для него наибольший интерес. В этой части также обсуждаются качества "хороших" алгоритмов и вопросы их реализации на практике. В частях II–IV описаны алгоритмы для вычисления внутренних, частных и характеристических паттернов соответственно.

Всего в книге 13 глав, разбитых на четыре части. Каждая глава, как видно из оглавления, разделена на несколько разделов, каждый из которых заканчивается набором упражнений. Там, где это необходимо, главы заканчиваются разделами, обобщающими изложенный материал и рассматривающими смежные вопросы, дополнительные темы и нерешенные проблемы.

Отметим, что в книге приведено примерно 500 упражнений, которые составляют неотъемлемую часть книги и могут использоваться для следующих целей.

- Для проверки степени усвоения читателем прочитанного материала.
- Сделать более ясными или показать в другом контексте понятия или результаты, приведенные в тексте.
- Показать расширения или модификации алгоритмов и математических результатов, которые приведены в тексте без подробного обсуждения.
- Показать детали алгоритмов и доказательств, которые не включены в основной текст из-за их громоздкости или их "ухода" от основой темы.

С помощью упражнений я также пытался вовлечь читателя в процесс разработки и анализа представленных алгоритмов. Этим я хотел показать, что в основе большинства алгоритмов и их модификаций лежат простые идеи, проникнуть в суть которых не составляет труда, но которые, может быть, "затемнены" или отброшены предыдущими исследователями. Если идея понята и принята, остается только ее техническая реализация. Это общее замечание относится и к строковым алгоритмам.

Признаюсь, что непосредственно в процессе написания книги я мог допускать ошибки. Поэтому при ее вычитке я старался исправить все замеченные ошибки и оплошности и сгладить шероховатости изложения материала. Но, конечно, я не

могу гарантировать, что внимательный читатель не найдет "дефектов" в моей книге. Я поддерживаю Web-узел

`http://www.cs.curtin.edu.au/~smyth/patterns.shtml`,

где вы можете оставить свои сообщения о замеченных ошибках и предложения по улучшению книги. Я также буду благодарен, если читатели по этой же причине свяжутся со мной по электронной почте

`smyth@computing.edu.au` или `smyth@mcmaster.ca`

Материал книги можно использовать, как минимум, для чтения двухсеместрового курса (каждый семестр по 12–14 недель) для студентов старших курсов и аспирантов. Материал первых глав в разное время уже читался аспирантам факультета компьютерных наук и систем и факультета вычислительной техники и программного обеспечения университета Мак-Мастера, г. Гамильтон, Онтарио, Канада, и аспирантам факультета компьютерных наук университета Дебрецена, Венгрия. Аспиранты, прослушавшие эти курсы, внесли свой вклад в создание этой книги.

Я хотел бы выразить глубочайшую благодарность школе вычислительной техники университета Кортина, Перт, Западная Австралия, и ее бывшим и настоящему руководителям Деннису Муру, Терри Силли, Свете Венкатеш и Джеффу Уесту (Dennis Moore, Terry Cealli, Svetha Venkatesh, Geoff West) за щедрую поддержку и за содействие, как интеллектуальное, так и практическое, на протяжении нескольких лет. Бóльшая часть книги написана во время моих продолжительных визитов в Кортин. Я также благодарен профессору Петью Аттиле (Pethő Attila), декану факультета компьютерных наук университета Дебрецена, за его интерес к моей работе и поддержку, особенно на последнем этапе создания электронного варианта книги. Хотел бы также выразить свою глубокую благодарность моим друзьям и коллегам Лейле Багдади, Джерри Чепплу, Франью Франеку, Костасу Илиопулосу, Терри Лекроку, Ян Ли, Деннису Муру, Пату Риану, Джеми Симпсону и Ксиангдонгу Ксиао (Leila Baghdadi, Jerry Chapple, Franya Franěk, Costas Iliopoulos, Thierry Lecroq, Yin Li, Dennis Moore, Pat Ryan, Jaime Simpson, Xiangdong Xiao) за их дружеское и полезное содействие. Большая благодарность двум анонимным рецензентам, которые сделали конструктивные замечания и предложения по книге. Наконец, возношу хвалу своей дочери Жаклин за ее восхитительный выбор и попытки попробовать на вкус слова "строка" и "слово".

W.F.S.

Моим родителям

ЧАСТЬ I

Строковые последовательности и алгоритмы

Слово спасет мир (если не будет поздно)

— Аноним

ГЛАВА 1

Свойства строковых последовательностей

Истинные слова не могут быть красивыми;
Красивые слова не могут быть истинными.

— Лао-цзы (604–531 гг. до н.э.). Путь Лао-цзы

1.1 Последовательность жемчужин

Предположим, что у нас есть кучка жемчужин. Для наглядности расположим жемчужины на столе слева направо в одну линию рядом друг с другом. Пусть всего есть n жемчужин, на каждую жемчужину приклеим маленькую бирку с меткой. Предположим, что метками являются целые числа от 1 до n и метки присваиваются жемчужинам по следующим правилам.

- Метка самой левой жемчужины равна 1.
- Для жемчужины с меткой i ($i = 1, 2, \ldots, n-1$) соседняя справа жемчужина имеет метку $i + 1$.

Эти правила удовлетворяют нашему интуитивному представлению о том, как сделать из кучки жемчужин "строковую последовательность": жемчужины расположены в одномерную цепочку (ряд), по которой можно перемещаться из одного конца в другой через соседние жемчужины. Но опираясь на приведенные выше правила, можно сформулировать более общее понятие последовательности. Во-первых, можно отказаться от "жемчужин", если ввести понятие ***элемент*** последовательности. Во-вторых, метки не обязаны принимать значения только целых

положительных чисел. Метки могут быть, например, цветными или буквами какого-нибудь алфавита. Что действительно важно, так это следующие правила, определяющие строковую последовательность.

0. Каждый элемент последовательности имеет уникальную метку.

1. Каждый элемент с некоторой меткой x (за исключением не более одного элемента, который называется ***самый левый элемент***) имеет единственный ***предшествующий элемент*** с меткой $p(x)$.

2. Каждый элемент с некоторой меткой x (за исключением не более одного элемента, который называется ***самый правый элемент***) имеет единственный ***последующий элемент*** с меткой $s(x)$.

3. Для любого элемента с меткой x, который не является самым левым, выполняется равенство $x = s(p(x))$.

4. Для любого элемента с меткой x, который не является самым правым, выполняется равенство $x = p(s(x))$.

5. Для двух различных элементов с метками x и y существует такое целое положительное число k, что $x = s^k(y)$ или $x = p^k(y)$.

Эти правила охватывают сущность понятия ***сочленения*** и операции ***конкатенации***, выполняющей это сочленение: каждый элемент последовательности, за исключением самого левого и самого крайнего элементов, имеет единственный предшествующий и единственный последующий элементы. Самый левый и самый крайний элементы имеют или единственного последующего, или единственного предшествующего элемента. Более того, начиная с любого элемента с меткой x мы, перебирая конечную последовательность предшествующих и последующих элементов, можем достичь любого другого элемента с меткой y. Отсюда вытекает следующее утверждение.

Определение 1.1.1. Строковая последовательность — это набор элементов, которые удовлетворяют правилам 0–5. ■

Критическим (но не очевидным) фактором этого определения являются условия в правилах 1 и 2, что может быть *не более одного* самого левого и *не более одного* самого правого элементов. Для примера рассмотрим, что случится, если из цепочки жемчужин сформировать петлю, т.е. расположить их по кругу. Теперь в этой последовательности жемчужин нет "самого левого" и "самого правого" элементов. Однако отметим, что это не порождает проблем, поскольку все правила 0–5 выполняются.

Теперь предположим, что количество жемчужин бесконечно — есть левый конец цепочки жемчужин, а правый конец "скрыт за горизонтом". Для этой бесконечной последовательности также выполняется определение 1.1.1, при этом существует самый левый элемент, но не существует самого правого. Нетрудно

проверить, что правила 0–5 будут выполняться, если цепочку жемчужин продолжить до бесконечности и в левом направлении; в этом случае нет ни самого левого, ни самого правого элементов.

В этой книге в свое время будут рассмотрены все возможные виды строковых последовательностей. Примем следующее соглашение по терминологии. Строковую последовательность (string) для краткости будем называть ***строкой***.[1] Строку с конечным числом элементов, содержащую как самый левый, так и самый правый элементы, назовем ***линейной строковой последовательностью*** или ***линейной строкой***. Строковую последовательность с конечным (не нулевым) числом элементов, не имеющую ни самого левого, ни самого правого элементов, назовем ***строковой петлей*** (в литературе такой вид последовательности также имеет название ***циклическая*** (или ***круговая***) ***строковая последовательность***). Строковую последовательность с бесконечным количеством элементов, содержащую самый левый элемент, назовем ***бесконечной строковой последовательностью*** или ***бесконечной строкой***; бесконечную строковую последовательность, не имеющую ни самого левого, ни самого правого элементов, назовем ***бесконечной петлей***. Далее термин "строковая последовательность" (или просто "строка") будет означать любой объект, который удовлетворяет определению 1.1.1. Тип строки обычно легко определить из контекста. Кроме того, на практике нетрудно отличить линейную строку от петли, поскольку, как правило, петля определяется через соответствующую линейную строку $\boldsymbol{x}$ и записывается как $C(\boldsymbol{x})$, т.е. петля $C(\boldsymbol{x})$ формируется из линейной строки $\boldsymbol{x}$ путем назначения самого левого элемента этой последовательности последующим элементом для самого правого элемента.

Упражнения 1.1

1. Объясните, почему для сочленения строковой последовательности необходимо правило 3? Приведите пример математического объекта, для которого выполняются правила 0–2, но не выполняется правило 3.
2. Можно ли правило 4 вывести из правил 0–3? Объясните свой ответ.
3. Объясните, почему для сочленения строковой последовательности необходимо правило 5? Приведите пример математического объекта, для которого выполняются правила 0–4, но не выполняется правило 5.
4. Является ли бесконечное множество $\{a,\ a^2,\ \ldots\}$ бесконечной строковой последовательностью?
5. Может ли в соответствии с определением 1.1.1 строковая последовательность состоять из одного элемента a? Может ли такая последовательность быть петлей?

[1] В русской математической литературе строковые последовательности часто называют "словами". — *Примеч. ред.*

6. Может ли в соответствии с определением 1.1.1 строковая последовательность не иметь элементов (такая последовательность или строка называется ***пустой***)? Может ли пустая строка быть линейной строковой последовательностью? А петлей?

7. С учетом предыдущих упражнений определите, сколько различных типов строк нужно дополнительно включить в классификацию строковых последовательностей, данную в этом разделе?

8. Классификация строковых последовательностей, приведенная в этом разделе, не включает следующие типы строк.

а) строковая последовательность с бесконечным числом элементов, имеющая самый правый элемент, но не имеющая самого левого;

б) строковая последовательность с конечным числом элементов, имеющая или самый правый элемент, или самый левый, но не оба сразу.

Объясните эти исключения.

9. Существует ли какой-нибудь способ *доказать*, что определение 1.1.1 определяет строковую последовательность?

1.2 Линейные строковые последовательности

Определение строковых последовательностей, данное в предыдущем разделе, очень общее и охватывает многие типы последовательностей, встречающиеся на практике. Так, строками будут

- слова в английском языке, здесь элементами последовательностей будут прописные и строчные буквы английского алфавита вместе с апострофом и дефисом;
- текстовые файлы, здесь элементами будут символы в кодировке ASCII;
- книга, написанная на китайском языке, где элементами последовательности являются китайские иероглифы;
- компьютерная программа, где элементами будут определенные (пробел, двоеточие, точка с запятой и т.п.) вместе со "словами", со "словами", заключенными между этими разделителями;
- последовательность молекул ДНК (иногда длиной до трех миллиардов элементов-молекул), состоящая только из букв C, G, A и T, которые соответствуют нуклеотидам цитозину, гуанину, аденину и тимину;
- поток частиц, бомбардирующих космический аппарат;
- список длин сторон выпуклого многоугольника.

Это примеры того, что в предыдущем разделе мы назвали "линейной строковой последовательностью". В книге, в основном, рассматриваются именно такие строки, поэтому в данном разделе даны некоторые определения и соответствующая терминология, необходимые для работы с линейными строковыми последовательностями. Большинство (но не все) из представленного ниже применимо также к петлям, бесконечным и пустым строкам.

В приведенных выше примерах видно, что важным свойством любой строки является природа ее элементов, будь-то действительные числа, слова какого-либо языка, иероглифы и т.п. На практике удобно представлять элементы строк как члены некоторого множества. Такое множество называется ***алфавитом***, поэтому члены этого множества естественно назвать ***буквами*** (далее мы увидим, что термин "буква" можно интерпретировать значительно шире, чем просто буква какого-нибудь естественного языка).[2] Будем говорить, что строковая последовательность ***определена на этом алфавите***. Конечно, если строка $\boldsymbol{x}$ определена на алфавите A, то она определена и на любом множестве, подмножеством которого является алфавит A. Другими словами, алфавит для данной строки $\boldsymbol{x}$ определяется неоднозначно. Можно определить минимальный алфавит для строки $\boldsymbol{x}$ как множество различных элементов, каждый из которых обязательно входит в $\boldsymbol{x}$. Иногда принимается соглашение, что алфавитом может быть только минимальный алфавит (например, "двоичный" алфавит), иногда это условие снимается (пример — алфавит "действительных чисел").

В книге алфавит будет обозначаться как A, а $\alpha = |A|$ будет обозначать его мощность. В случае α, равного 2, 3 и 4, алфавит будет называться ***бинарным***, ***тернарным*** и ***кватернарным*** соответственно. Далее мы увидим, сколько интересных строковых последовательностей можно построить на бинарном алфавите. Последовательности на основе кватернарного алфавита находят применение при анализе последовательностей ДНК.

В общем случае, кроме требования "различности" элементов множества A, на само это множество особых условий не накладывается. Алфавит может быть конечным (даже нулевым!) множеством, счетным множеством (например, множество целых чисел) и даже несчетным (как континуальное множество вещественных чисел). Кроме того, элементы алфавита могут быть упорядоченными (тогда для любой пары различных элементов определен результат их сравнения "больше" или "меньше"), неупорядоченными и частично упорядоченными. Для многих алгоритмов, описанных в книге, используется неупорядоченный алфавит. В главе 4 показано, что для многих алгоритмов свойство упорядоченности алфавита значительно влияет на их вычислительную эффективность и на выбор тестовых данных для проверки алгоритмов.

[2]Из этих определений вытекает обоснованность терминов "строковая последовательность" и "строка". — *Примеч. ред.*