Предисловие

Предлагаемая книга основана на курсе лекций, читаемых автором на факультете вычислительной математики и кибернетики (ВМК) МГУ им. М. В. Ломоносова. Ее цель — обсуждение основных понятий теории сложности и некоторых методов анализа сложности алгоритмов. Это обсуждение сопровождается подробным рассмотрением со сложностной точки зрения ряда алгоритмов арифметики, сортировки и поиска, вычислительной геометрии, теории графов и др. Материал группируется по главам и параграфам в соответствии с разделами самой теории сложности — сложность в худшем случае, сложность в среднем, нижние границы сложности и т. д. Для того, чтобы сосредоточиться именно на понятиях и подходах теории сложности, при этом не затрачивая слишком много времени на объяснение деталей трудных для понимания алгоритмов, в примерах исследуются либо алгоритмы достаточно известные (сложностные характеристики которых менее известны), либо алгоритмы, суть которых может быть изложена коротко. Сложностные вопросы рассматриваются в книге довольно подробно, но книга значительно уступает по широте охвата алгоритмического материала книгам Д. Кнута [25, 26, 27], А. Ахо, Дж. Хопкрофта и Дж. Ульмана [8], Т. Кормена, Ч. Лейзерсона, Р. Ривеста [28], С. Баасе и А. ван Гелдера [61], Ж. Брассара и П. Берли [64], отражающим обширный спектр вопросов построения алгоритмов, и книгам по специальным алгоритмическим разделам математики — вычислительной геометрии (Ф. Препарата, М. Шеймос [44]), алгоритмической теории чисел (Э. Бах, Дж. Шаллит [62]), комбинаторике (Э. Рейнгольд, Ю. Нивергельт, Н. Део [46], С. Джукна [75]) и др. Здесь надо сказать, что названная литература, как и некоторые другие издания, послужила источником ряда примеров и задач, предлагаемых в этой книге.

В последних трех параграфах, касающихся классов P и NP, понятия NP-полноты и т. д., ряд фактов дается без доказательства. Это объясняется тем, что в книге используются менее формальные модели вычислений, чем, скажем, машина Тьюринга, а доказательства упомянутых фактов опираются на полностью формализованные модели. Недостающие доказательства могут быть найдены, например, в [5], [15], [20], [31].

В приложениях даются дополнительные сведения по затрагиваемым в основном тексте вопросам. Исключение составляют приложения A, F: в первом из них содержатся необходимые сведения о ряде алгоритмов сортировки и поиска, во втором — о методе решения линейных рекуррентных уравнений с постоянными коэффициентами; эти два приложения носят характер напоминания.

Библиографические комментарии даются в сносках.

Каждая из глав снабжена задачами для самостоятельного решения, среди которых помимо легких имеются и такие, которые углубляют материал главы, поэтому полезно по крайней мере прочитывать условия всех задач. Задача содержит указание к решению в тех, например, случаях (довольно редких), когда в основном тексте имеется отсылка к этой задаче. По всем главам для задач используется сквозная нумерация. Многие из предлагаемых задач имеют вид утверждений, подразумевается, что каждое такое утверждение надо доказать.

Сжатое изложение курса этих лекций представлено в виде расширяемого эссе [3] на веб-странице

`http://www.ccas.ru/sabramov/essay/`

Автор благодарен своим коллегам по ВМК МГУ В. Б. Алексееву и В. П. Иванникову, а также Е. В. Зиме (университет Уилфрида Лорье, Ватерлоо, Канада) и М. Петковшеку (университет Любляны, Словения), беседы и дискуссии с которыми существенно помогли в отборе материала и выработке общей схемы курса лекций и этой книги, при этом надо особо отметить, что Е. В. Зима принял участие в написании главы 6. Большая благодарность и А. В. Бернштейну (ИСА РАН), А. А. Васину (ВМК МГУ), В. А. Серебрякову (ВЦ РАН), сделавшим полезные замечания по ряду разделов книги. Автор признателен рецензентам М. Н. Вялому (ВЦ РАН) и С. Б. Гашкову (мехмат МГУ) — их пометки на полях рукописи оказали значительную помощь на заключительном этапе работы над книгой. Особая благодарность за советы и многочисленные конструктивные замечания Е. А. Бордаченковой (ВМК МГУ) — научному редактору книги.

* * *

Первое издание книги вышло в 2009 г., второе — в 2012 г., третье — в 2020 г. Настоящее четвертое издание книги отличается от предыдущего исправлением замеченных погрешностей и опечаток. Добавлено некоторое количество задач. Автор благодарит М. Кауэрса (университет И. Кеплера в Линце, Австрия), А. А. Разборова (Чикагский университет) за полезные советы.

Введение

Исследование *сложности* алгоритма помогает понять степень его практической приемлемости. Сравнительный анализ сложности нескольких алгоритмов решения одной и той же задачи позволяет делать обоснованный выбор лучшего из них. Например, если для решения задачи предлагается новый алгоритм, то необходимы доводы, говорящие о его преимуществах в сравнении с известными ранее алгоритмами, и анализ сложности может предоставить такие доводы.

Слово «сложность» в этом контексте является математическим термином, а не общим обозначением препятствия к выполнению замысла. С понятием сложности связываются затраты времени или памяти, соответствующие худшему случаю, либо затраты в среднем; при этом, чтобы обсуждать худший или «средний» случай, нужно, прежде всего, договориться, как определяется *размер входа* (размер входных данных) алгоритма и в чем измеряются *затраты* при работе алгоритма над фиксированным входом.

Часто размер входа определяют как общее число символов в представлении входа, но возможны и другие пути. В задачах сортировки и поиска размер входа — это, как правило, количество элементов n входного массива, в задачах на графах — число вершин $|V|$ или число ребер $|E|$ входного графа $G = (V, E)$, но, с другой стороны, $|V|$ и $|E|$ могут рассматриваться и совместно как две *компоненты размера* входа. В арифметических задачах размером входа может быть, например, максимум абсолютных величин входных целых чисел, или количество цифр в двоичной записи этого максимума, или же, как уже говорилось, суммарное количество двоичных цифр всех входных чисел и т. д. — выбор делается в зависимости от характера задачи.

Для алгоритмов сортировки соответствующие затраты времени достаточно полно характеризуются количеством сравнений и перемещений элементов массива. Для алгоритмов на графах учитываемые затраты могут складываться, например, из операций над матрицей смежности или над массивом списков смежных вершин данного графа и из некоторых дополнительных вычислительных операций. Для арифметических алгоритмов в качестве объема затрат может быть взято количество всех выполняемых арифметических операций, или, как альтернатива, лишь наиболее дорогих операций (например, мультипликативных — умножений и делений); более тщательный анализ требует рассмотрения количества *битовых* операций. Если алгоритм является *рандомизированным* (содержит обращения к генератору случайных чисел с известным распределением), то затраты, вообще говоря, не определяются однозначно входом алгоритма, но зависят от полученных случайных чисел. Можно рассматривать усредненные затраты для каждого конкретного входа; такие затраты уже будут функцией входа.

При фиксированном значении размера входа сами входы алгоритма могут варьироваться, при этом меняются и затраты; алгоритм может быть охарактеризован наибольшими или средними (в соответствии с распределением вероятностей на входах данного размера) затратами. В обоих случаях сложность алгоритма — это функция размера входа или, соответственно, нескольких компонент размера входа. Для этого понятия иногда используется термин *вычислительная сложность*, чтобы избежать путаницы с описательной (в другой терминологии — изобразительной) сложностью, которая тем или иным способом определяется исходя из самой записи (текста) алгоритма; одним из видов описательной сложности является длина записи алгоритма, и именно так понятие сложности трактуется в некоторых теориях[1]. Мы будем рассматривать только вычислительную сложность, называя ее просто сложностью.

Понятие сложности является математическим уточнением довольно расплывчатого термина «трудоемкость», с помощью которого, наряду с не более ясными «быстродействием» и «эффективностью», иногда пытаются характеризовать алгоритмы. Принятие в качестве сложности именно затрат в среднем или затрат, соответствующих худшему (а не, скажем, лучшему) случаю, помогает оценить достаточность имеющихся вычислительных ресурсов для выполнения алгоритма.

Итак, сложность является функцией числового, чаще всего целого, аргумента, иногда — нескольких таких аргументов. Теория сложности изучает эти функции, сопоставленные как отдельным алгоритмам, так и классам алгоритмов решения некоторой задачи. При рассмотрении класса алгоритмов возникает семейство функций, для которого могут обсуждаться вопросы о нахождении *нижней границы*, о существовании минимальной функции этого семейства (если такая функция существует, то соответствующий ей алгоритм — *оптимальный* в рассматриваемом классе) и т. д. Важным является также

[1] См., например, статью «Алгоритма сложность» в [58].

вопрос о существовании в данном классе алгоритмов, нацеленных на решение конкретной задачи, алгоритма такого, сложность которого растет с увеличением размера входа не слишком быстро, например, остается ограниченной некоторым полиномом, вид которого не оговаривается (такой алгоритм принято называть *полиномиальным*).

При исследовании существования алгоритма решения задачи, имеющего «не очень высокую» сложность, важную роль играет *сводимость* какой-то задачи к другой — из возможности быстро решить некоторую задачу может следовать такая же возможность для ряда других, и наоборот, невозможность быстрого решения какой-то одной задачи может автоматически повлечь такую же невозможность для ряда других задач.

Сложности многих алгоритмов трудно или вообще нельзя представить в простом «замкнутом» виде; помимо этого, точное значение сложности алгоритма для каждого конкретного значения размера входа часто не представляет особого интереса, актуальным же является исследование роста сложности при возрастании размера входа (особый интерес к исходным данным большого размера оправдан тем, что на них «захлебываются» тривиальные алгоритмы). Поэтому в теории сложности широко используется *асимптотическое* оценивание. Однако сравнение сложностей различных алгоритмов на основе асимптотических оценок этих сложностей возможно не для всех типов таких оценок.

Этот круг вопросов, наряду с рассмотрением общих достаточно элементарных подходов и методов, которые могут оказаться полезными при сложностном анализе алгоритмов, составляет содержание предлагаемого курса теории сложности алгоритмов.