

Алгоритм

Материал из Википедии — свободной энциклопедии

Алго́ри́зм (лат. *algorithmi* — от имени среднеазиатского математика Аль-Хорезми^[1]) — конечная совокупность точно заданных правил решения некоторого класса задач или набор *инструкций*, описывающих порядок действий исполнителя для решения определённой задачи. В старой трактовке вместо слова «порядок» использовалось слово «последовательность», но по мере развития параллельности в работе компьютеров слово «последовательность» стали заменять более общим словом «порядок». Независимые инструкции могут выполняться в произвольном порядке, параллельно, если это позволяют используемые исполнители.

Ранее в русском языке писали «алгори**ф**м», сейчас такое написание используется редко, но тем не менее имеет место исключение (*нормальный алгори**ф**м Маркова*).

Часто в качестве исполнителя выступает компьютер, но понятие алгоритма необязательно относится к *компьютерным программам*, так, например, чётко описанный рецепт приготовления блюда также является алгоритмом, в таком случае исполнителем является человек (а может быть и некоторый механизм, ткацкий станок, и пр.).

Можно выделить алгоритмы *вычислительные* (далее речь в основном идёт о них), и *управляющие*. Вычислительные, по сути, преобразуют некоторые начальные данные в выходные, реализуя вычисление некоторой функции. Семантика управляющих алгоритмов существенным образом может отличаться и сводиться к выдаче необходимых управляющих воздействий либо в заданные моменты времени, либо в качестве реакции на внешние события (в этом случае, в отличие от вычислительного алгоритма, управляющий может оставаться корректным при бесконечном выполнении).

Понятие алгоритма относится к первоначальным, основным, базисным понятиям математики. Вычислительные процессы алгоритмического характера (арифметические действия над целыми числами, нахождение наибольшего общего делителя двух чисел и т. д.) известны человечеству с глубокой древности. Однако в явном виде понятие алгоритма сформировалось лишь в начале XX века.

Частичная формализация понятия алгоритма началась с попыток решения *проблемы разрешения* (нем. *Entscheidungsproblem*), которую сформулировал Давид Гильберт в 1928 году. Следующие этапы формализации были необходимы для определения эффективных вычислений^[2] или «эффективного метода»^[3]; среди таких формализаций — рекурсивные функции Геделя — Эрбрана — Клини 1930, 1934 и 1935 гг., *λ-исчисление* Алонзо Чёрча 1936 г., «*Формулировка 1*» Эмиля Поста 1936 года и *машина Тьюринга*.

Содержание

История термина

Определения алгоритма

Свойства алгоритмов

Формальное определение

Машина Тьюринга

Рекурсивные функции

Нормальный алгоритм (алгори́фм) Маркова

Стохастические алгоритмы

Другие формализации

Виды алгоритмов

Нумерация алгоритмов

Алгоритмически неразрешимые задачи

Анализ алгоритмов

Время работы

Наличие исходных данных и некоторого результата

Представление алгоритмов

Эффективность алгоритмов

Пример

См. также

Примечания

Литература

Ссылки

История термина

Современное формальное определение вычислительного алгоритма было дано в 30—50-е годы XX века в работах Тьюринга, Поста, Чёрча (тезис Чёрча — Тьюринга), Н. Винера, А. А. Маркова.

Само слово «алгоритм» происходит от имени хорезмского учёного аль-Хорезми. Около 825 года он написал сочинение *Китаб аль-джебр валь-мукабала* («Книга о сложении и вычитании»), из оригинального названия которого происходит слово «алгебра» (аль-джебр — восполнение). В этой книге впервые дал описание придуманной в Индии позиционной десятичной системы счисления. Персидский оригинал книги не сохранился. Аль-Хорезми сформулировал правила вычислений в новой системе и, вероятно, впервые использовал цифру 0 для обозначения пропущенной позиции в записи числа (её индийское название арабы перевели как *as-sifr* или просто *sifr*, отсюда такие слова, как «цифра» и «шифр»). Приблизительно в это же время индийские цифры начали применять и другие арабские учёные.

В первой половине XII века книга аль-Хорезми в латинском переводе проникла в Европу. Переводчик, имя которого до нас не дошло, дал ей название *Algoritmi de numero Indorum* («Алгоритми о счёте индийском») — таким образом, латинизированное имя среднеазиатского учёного было вынесено в заглавие книги. Сегодня считается, что слово



Страница из «Алгебры» аль-Хорезми — хорезмского математика, от имени которого происходит слово *алгоритм*.

Сегодня считается, что слово

«алгоритм» попало в европейские языки именно благодаря этому переводу. В течение нескольких следующих столетий появилось множество других трудов, посвящённых всё тому же вопросу — обучению искусству счёта с помощью цифр, и все они имели в названии слово *algoritmi* или *algorismi*.

Про аль-Хорезми позднейшие авторы ничего не знали, но поскольку первый перевод книги начинается словами: «Dixit algorizmi: ...» («Аль-Хорезми говорил: ...»), всё ещё связывали это слово с именем конкретного человека. Очень распространённой была версия о греческом происхождении книги. В англо-норманнской рукописи XIII века, написанной в стихах, читаем:

Алгоризм был придуман в Греции.

Это часть арифметики. Придуман он был мастером по имени Алгоризм, который дал ему своё имя. И поскольку его звали Алгоризм,

Он назвал свою книгу «Алгоризм».



Аль-Хорезми на советской марке

Около 1250 года английский астроном и математик Иоанн Сакробоско написал труд по арифметике *Algorismus vulgaris*, на столетия ставший основным учебником по вычислениям в десятичной позиционной системе счисления во многих европейских университетах. Во введении Сакробоско назвал автором науки о счёте мудреца по имени Алгус (*Algus*). А в популярной средневековой поэме «Роман о Розе» (1275—1280) Жана де Мена «греческий философ Алгус» ставится в один ряд с Платоном, Аристотелем, Евклидом и Птолемеем! Встречался также вариант написания имени Аргус (*Argus*). И хотя, согласно древнегреческой мифологии, корабль «Арго» был построен Ясоном, именно этому Арго приписывалось строительство корабля.

«Мастер Алгус» (или Аргус) стал в средневековой литературе олицетворением счётного искусства. И в уже упоминавшейся «Романе о розе», и в известной итальянской поэме «Цветок», написанной Дуранте, имеются фрагменты, в которых говорится, что даже «mestre Argus» не сумеет подсчитать, сколько раз ссорятся и мирятся влюблённые. Английский поэт Джеффри Чосер в поэме «Книга герцогини» (1369 г.) пишет, что даже «славный счётчик Аргус» (noble countour Argu) не сможет счесть чудовищ, явившихся в кошмарных видениях герою.

Однако со временем такие объяснения всё менее занимали математиков, и слово *algorism* (или *algorismus*), неизменно присутствовавшее в названиях математических сочинений, обрело значение способа выполнения арифметических действий посредством арабских цифр, то есть на бумаге, без использования абака. Именно в таком значении оно вошло во многие европейские языки. Например, с пометкой «устар.» оно присутствует в представительном словаре английского языка *Webster's New World Dictionary*, изданном в 1957 г. Энциклопедический словарь Брокгауза и Ефрона предлагает такую трактовку: алгори́зм (кстати, до революции использовалось написание алгори́зм, через фиту) производится «от арабского слова Аль-Горетм, то есть корень».

Алгоритм — это искусство счёта с помощью цифр, но поначалу слово «цифра» относилось только к нулю. Знаменитый французский трувер Готье де Куанси (Gautier de Coincy, 1177—1236) в одном из стихотворений использовал слова *algorismus-cipher* (которые означали цифру 0) как метафору для характеристики абсолютно никчёмного человека. Очевидно, понимание такого образа требовало

соответствующей подготовки слушателей, а это означает, что новая система счисления уже была им достаточно хорошо известна.

Многие века абак был фактически единственным средством для практических вычислений, им пользовались и купцы, и менялы, и учёные. Достоинства вычислений на счётной доске разяснял в своих сочинениях такой выдающийся мыслитель, как Герберт Аврилакский (938—1003), ставший в 999 г. папой римским под именем Сильвестра II. Новое с огромным трудом пробивало себе дорогу, и в историю математики вошло упорное противостояние лагерей алгорисмиков и абацистов (иногда называемых гербекистами), которые пропагандировали использование для вычислений абака вместо арабских цифр. Интересно, что известный французский математик Николя Шюке (Nicolas Chuquet, 1445—1488) в реестр налогоплательщиков города Лиона был вписан как алгорисмик (algoriste). Но прошло не одно столетие, прежде чем новый способ счёта окончательно утвердился, столько времени потребовалось, чтобы выработать общепризнанные обозначения, усовершенствовать и приспособить к записи на бумаге методы вычислений. В Западной Европе учителей арифметики вплоть до XVII века продолжали называть «магистрами абака», как, например, математика Никколо Тарталью (1500—1557).



Баронесса Ада Лавлейс, которую считают первым программистом.

Итак, сочинения по искусству счёта назывались *Алгоритмами*. Из многих сотен можно выделить и такие необычные, как написанный в стихах трактат *Carmen de Algorismo* (латинское *carmen* и означает стихи) Александра де Вилла Деи (Alexander de Villa Dei, ум. 1240) или учебник венского астронома и математика Георга Пурбаха (Georg Peurbach, 1423—1461) *Opus algorismi jocundissimi* («Веселейшее сочинение по алгоритму»).

Постепенно значение слова расширялось. Учёные начинали применять его не только к сугубо вычислительным, но и к другим математическим процедурам. Например, около 1360 г. французский философ Николай Орем (Nicolaus Oresme, 1323/25-1382) написал математический трактат *Algorismus proportionum* («Вычисление пропорций»), в котором впервые использовал степени с дробными показателями и фактически вплотную подошёл к идее логарифмов. Когда же на смену абаку пришёл так называемый счёт на линиях, многочисленные руководства по нему стали называть *Algorismus linealis*, то есть правила счёта на линиях.

Можно обратить внимание на то, что первоначальная форма *algorismi* спустя какое-то время потеряла последнюю букву, и слово приобрело более удобное для европейского произношения вид *algorism*. Позднее и оно, в свою очередь, подверглось искажению, скорее всего, связанному со словом *arithmetic*.

В 1684 году Готфрид Лейбниц в сочинении *Nova Methodus pro maximis et minimis, itemque tangentibus...* впервые использовал слово «алгоритм» (*Algorithmo*) в ещё более широком смысле: как систематический способ решения проблем дифференциального исчисления.

В XVIII веке в одном из германских математических словарей, *Vollstandiges mathematisches Lexicon* (изданном в Лейпциге в 1747 г.), термин *algorithmus* всё ещё объясняется как понятие о четырёх арифметических операциях. Но такое значение не было единственным, ведь терминология математической науки в те времена ещё только формировалась. В частности, выражение *algorithmus infinitesimalis* применялось к способам выполнения действий с бесконечно малыми величинами. Пользовался словом алгоритм и Леонард Эйлер, одна из работ которого так и называется —

«Использование нового алгоритма для решения проблемы Пелля» (*De usu novi algorithmi in problemate Pelliano solvendo*). Мы видим, что понимание Эйлером алгоритма как синонима способа решения задачи уже очень близко к современному.

Однако потребовалось ещё почти два столетия, чтобы все старинные значения слова вышли из употребления. Этот процесс можно проследить на примере проникновения слова «алгоритм» в русский язык.

Историки датируют 1691 годом один из списков древнерусского учебника арифметики, известного как «Счётная мудрость». Это сочинение известно во многих вариантах (самые ранние из них почти на сто лет старше) и восходит к ещё более древним рукописям XVI в. По ним можно проследить, как знание арабских цифр и правил действий с ними постепенно распространялось на Руси. Полное название этого учебника — «Сия книга, глаголемая по-еллински и по-гречески арифметика, а по-немецки алгоризма, а по-русски цифирная счётная мудрость».

Таким образом, слово «алгоритм» понималось первыми русскими математиками так же, как и в Западной Европе. Однако его не было ни в знаменитом словаре В. И. Даля, ни спустя сто лет в «Толковом словаре русского языка» под редакцией Д. Н. Ушакова (1935 г.). Зато слово «алгорифм» можно найти и в популярном дореволюционном Энциклопедическом словаре братьев Гранат, и в первом издании Большой советской энциклопедии (БСЭ), изданном в 1926 г. И там, и там оно трактуется одинаково: как правило, по которому выполняется то или иное из четырёх арифметических действий в десятичной системе счисления. Однако к началу XX в. для математиков слово «алгоритм» уже означало любой арифметический или алгебраический процесс, выполняемый по строго определённым правилам, и это объяснение также даётся в следующих изданиях БСЭ.

Алгоритмы становились предметом всё более пристального внимания учёных, и постепенно это понятие заняло одно из центральных мест в современной математике. Что же касается людей, от математики далёких, то к началу сороковых годов это слово они могли услышать разве что во время учёбы в школе, в сочетании «алгоритм Евклида». Несмотря на это, алгоритм всё ещё воспринимался как термин сугубо специальный, что подтверждается отсутствием соответствующих статей в менее объёмных изданиях. В частности, его нет даже в десятичной Малой советской энциклопедии (1957 г.), не говоря уже об одностомных энциклопедических словарях. Но зато спустя десять лет, в третьем издании Большой советской энциклопедии (1969 г.) алгоритм уже характеризуется как одна из основных категорий математики, «не обладающих формальным определением в терминах более простых понятий, и абстрагируемых непосредственно из опыта». Как мы видим, отличие даже от трактовки первым изданием БСЭ разительное! За сорок лет алгоритм превратился в одно из ключевых понятий математики, и признанием этого стало включение слова уже не в энциклопедии, а в словари. Например, оно присутствует в академическом «Словаре русского языка» (1981 г.) именно как термин из области математики.

Одновременно с развитием понятия алгоритма постепенно происходила и его экспансия из чистой математики в другие сферы. И начало ей положило появление компьютеров, благодаря которому слово «алгоритм» вошло в 1985 г. во все школьные учебники информатики и обрело новую жизнь. Вообще можно сказать, что его сегодняшняя известность напрямую связана со степенью распространения компьютеров. Например, в третьем томе «Детской энциклопедии» (1959 г.) о вычислительных машинах говорится немало, но они ещё не стали чем-то привычным и воспринимаются скорее как некий атрибут светлого, но достаточно далёкого будущего. Соответственно и алгоритмы ни разу не упоминаются на её страницах. Но уже в начале 70-х гг. прошлого столетия, когда компьютеры перестали быть экзотической диковинкой, слово «алгоритм» стремительно входит в обиход. Это чутко фиксируют энциклопедические издания. В «Энциклопедии кибернетики» (1974 г.) в статье «Алгоритм» он уже связывается с реализацией на вычислительных машинах, а в «Советской военной энциклопедии» (1976 г.) даже появляется отдельная статья «Алгоритм решения задачи на ЭВМ». За последние полтора-два десятилетия

компьютер стал неотъемлемым атрибутом нашей жизни, компьютерная лексика становится всё более привычной. Слово «алгоритм» в наши дни известно, вероятно, каждому. Оно уверенно шагнуло даже в разговорную речь, и сегодня мы нередко встречаем в газетах и слышим в выступлениях политиков выражения вроде «алгоритм поведения», «алгоритм успеха» или даже «алгоритм предательства». Академик Н. Н. Моисеев назвал свою книгу «Алгоритмы развития», а известный врач Н. М. Амосов — «Алгоритм здоровья» и «Алгоритмы разума». А это означает, что слово живёт, обогащаясь всё новыми значениями и смысловыми оттенками.

Определения алгоритма

Свойства алгоритмов

Различные определения алгоритма в явной или неявной форме содержат следующий ряд общих требований:

- Дискретность — алгоритм должен представлять процесс решения задачи как упорядоченное выполнение некоторых простых шагов. При этом для выполнения каждого шага алгоритма требуется конечный отрезок времени, то есть преобразование исходных данных в результат осуществляется во времени дискретно.
- Детерминированность (определённость). В каждый момент времени следующий шаг работы однозначно определяется состоянием системы. Таким образом, алгоритм выдаёт один и тот же результат (ответ) для одних и тех же исходных данных. В современной трактовке у разных реализаций одного и того же алгоритма должен быть изоморфный граф. С другой стороны, существуют вероятностные алгоритмы, в которых следующий шаг работы зависит от текущего состояния системы и генерируемого случайного числа. Однако при включении метода генерации случайных чисел в список «исходных данных» вероятностный алгоритм становится подвидом обычного.
- Понятность — алгоритм должен включать только те команды, которые доступны исполнителю и входят в его систему команд.
- Завершаемость (конечность) — в более узком понимании алгоритма как математической функции, при правильно заданных начальных данных алгоритм должен завершать работу и выдавать результат за определённое число шагов. Дональд Кнут процедуру, которая удовлетворяет всем свойствам алгоритма, кроме, возможно, конечности, называет *методом вычисления* (англ. *computational method*)^[4]. Однако довольно часто определение алгоритма не включает завершаемость за конечное время^[5]. В этом случае алгоритм (метод вычисления) определяет частичную функцию. Для вероятностных алгоритмов завершаемость как правило означает, что алгоритм выдаёт результат с вероятностью 1 для любых правильно заданных начальных данных (то есть может в некоторых случаях не завершиться, но вероятность этого должна быть равна 0).
- Массовость (универсальность). Алгоритм должен быть применим к разным наборам начальных данных.
- Результативность — завершение алгоритма определёнными результатами.

Формальное определение

Разнообразные теоретические проблемы математики и ускорение развития физики и техники поставили на повестку дня точное определение понятия алгоритма.

Первые попытки уточнения понятия алгоритма и его исследования осуществляли в первой половине XX века Алан Тьюринг, Эмиль Пост, Жак Эрбран, Курт Гедель, А. А. Марков, Алонзо Чёрч. Было разработано несколько определений понятия алгоритма, но впоследствии было выяснено, что все они определяют одно и то же понятие (см. Тезис Чёрча — Тьюринга)^[6]

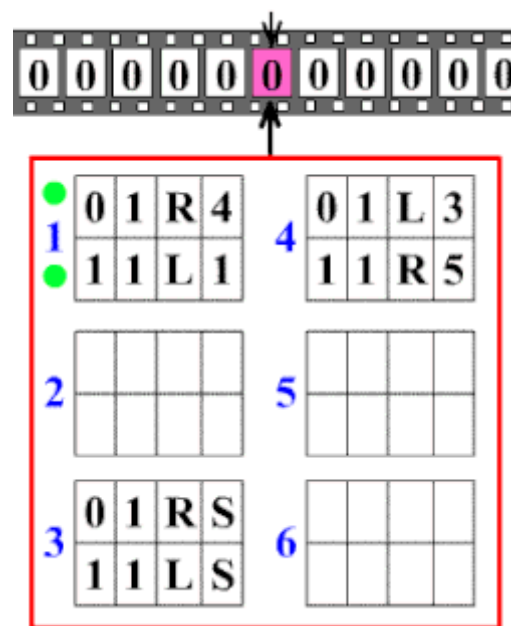
Российский математик, основоположник структурной лингвистики в Советском Союзе В. А. Успенский считал, что понятие алгоритма впервые появилось у Эмиля Бореля в 1912 году, в статье об определённом интеграле. Там он написал о «вычислениях, которые можно реально осуществить», подчеркивая при этом: «Я намеренно оставляю в стороне большую или меньшую практическую деятельность; суть здесь та, что каждая из этих операций осуществима в конечное время при помощи достоверного и недвусмысленного метода»^[7].

Машина Тьюринга

Основная идея, лежащая в основе машины Тьюринга, очень проста. Машина Тьюринга — это абстрактная машина (автомат), работающая с лентой отдельных ячеек, в которых записаны символы. Машина также имеет головку для записи и чтения символов из ячеек, которая может двигаться вдоль ленты. На каждом шаге машина считывает символ из ячейки, на которую указывает головка, и, на основе считанного символа и внутреннего состояния, делает следующий шаг. При этом машина может изменить своё состояние, записать другой символ в ячейку или передвинуть головку на одну ячейку вправо или влево.^[8]

На основе исследования этих машин был выдвинут тезис Тьюринга (основная гипотеза алгоритмов):

«*Некоторый алгоритм для нахождения значений функции, заданной в некотором алфавите, существует тогда и только тогда, когда функция исчисляется по Тьюрингу, то есть когда её можно вычислить на машине Тьюринга.*



Схематическая иллюстрация работы машины Тьюринга.

Этот тезис является аксиомой, постулатом, и не может быть доказан математическими методами, поскольку алгоритм не является точным математическим понятием.

Рекурсивные функции

С каждым алгоритмом можно сопоставить функцию, которую он вычисляет. Однако возникает вопрос, можно ли произвольной функции сопоставить машину Тьюринга, а если нет, то для каких функций существует алгоритм? Исследования этих вопросов привели к созданию в 1930-х годах теории рекурсивных функций^[9].

Класс вычислимых функций был записан в образ, напоминающий построение некоторой аксиоматической теории на базе системы аксиом. Сначала были выбраны простейшие функции, вычисление которых очевидно. Затем были сформулированы правила (операторы) построения новых функций на основе уже существующих. Необходимый класс функций состоит из всех функций, которые можно получить из простейших применением операторов.

Подобно тезису Тьюринга в теории вычислимых функций была выдвинута гипотеза, которая называется тезис Чёрча:

« Числовая функция тогда и только тогда алгоритмически исчисляется, когда она частично рекурсивна. »

Доказательство того, что класс вычислимых функций совпадает с исчисляемыми по Тьюрингу, происходит в два шага: сначала доказывают вычисление простейших функций на машине Тьюринга, а затем — вычисление функций, полученных в результате применения операторов.

Таким образом, неформально алгоритм можно определить как четкую систему инструкций, определяющих дискретный детерминированный процесс, который ведёт от начальных данных (на входе) к искомому результату (на выходе), если он существует, за конечное число шагов; если искомого результата не существует, алгоритм или никогда не завершает работу, либо заходит в тупик.

Нормальный алгоритм (алгорифм) Маркова

Нормальный алгоритм (алгорифм в авторском написании) Маркова — это система последовательных применений подстановок, которые реализуют определённые процедуры получения новых слов из базовых, построенных из символов некоторого алфавита. Как и машина Тьюринга, *нормальные алгоритмы* не выполняют самих вычислений: они лишь выполняют преобразование слов путём замены букв по заданным правилам^[10].

Нормально вычислимой называют функцию, которую можно реализовать нормальным алгоритмом. То есть алгоритмом, который каждое слово из множества допустимых данных функции превращает в её начальные значения^[11].

Создатель теории нормальных алгоритмов А. А. Марков выдвинул гипотезу, которая получила название принцип нормализации Маркова:

« Для нахождения значений функции, заданной в некотором алфавите, тогда и только тогда существует некоторый алгоритм, когда функция нормально исчисляемая. »

Подобно тезисам Тьюринга и Черча, принцип нормализации Маркова не может быть доказан математическими средствами.

Стохастические алгоритмы

Однако приведённое выше формальное определение алгоритма в некоторых случаях может быть слишком строгим. Иногда возникает потребность в использовании случайных величин^[12]. Алгоритм, работа которого определяется не только исходными данными, но и значениями, полученными из генератора случайных чисел, называют *стохастическим* (или рандомизированным, от англ. *randomized algorithm*)^[13]. Стохастические алгоритмы часто бывают эффективнее детерминированных, а в отдельных случаях — единственным способом решить задачу^[12].

На практике вместо генератора случайных чисел используют генератор псевдослучайных чисел.

Однако следует отличать стохастические алгоритмы и методы, которые дают с высокой вероятностью правильный результат. В отличие от метода, алгоритм даёт корректные результаты даже после продолжительной работы.

Некоторые исследователи допускают возможность того, что стохастический алгоритм даст с некоторой заранее известной вероятностью неправильный результат. Тогда стохастические алгоритмы можно разделить на два типа^[14]:

- алгоритмы типа Лас-Вегас всегда дают корректный результат, но время их работы не определено.
- алгоритмы типа Монте-Карло, в отличие от предыдущих, могут давать неправильные результаты с известной вероятностью.

Другие формализации

Для некоторых задач названные выше формализации могут затруднять поиск решений и осуществление исследований. Для преодоления препятствий были разработаны как модификации «классических» схем, так и созданы новые модели алгоритма. В частности, можно назвать:

- многоленточная и недетерминированная машины Тьюринга;
- регистровая и РАМ-машина — прототип современных компьютеров и виртуальных машин;
- конечные и клеточные автоматы

Виды алгоритмов

Виды алгоритмов как логико-математических средств отражают указанные компоненты человеческой деятельности и тенденции, а сами алгоритмы в зависимости от цели, начальных условий задачи, путей её решения. Следует подчеркнуть принципиальную разницу между алгоритмами вычислительного характера, преобразующими некоторые входные данные в выходные (именно их формализацией являются упомянутые выше машины Тьюринга, Поста, РАМ, нормальные алгорифмы Маркова и рекурсивные функции), и интерактивными алгоритмами (уже у Тьюринга встречается С-машина, от англ. *choice* — выбор, ожидающая внешнего воздействия, в отличие от классической А-машины, где все начальные данные заданы до начала вычисления и выходные данные недоступны до окончания вычисления). Последние предназначены для взаимодействия с некоторым объектом управления и призваны обеспечить корректную выдачу управляющих воздействий в зависимости от складывающейся ситуации, отражаемой поступающими от объекта управления сигналами^{[15][16]}. В некоторых случаях алгоритм управления вообще не предусматривает окончания работы (например, поддерживает бесконечный цикл ожидания событий, на которые выдается соответствующая реакция), несмотря на это, являясь полностью правильным.

Можно также выделить алгоритмы:

- **Механические алгоритмы**, или иначе **детерминированные, жесткие** (например, алгоритм работы машины, двигателя и т. п.) — задают определённые действия, обозначая их в единственной и достоверной последовательности, обеспечивая тем самым однозначный требуемый или искомый результат, если выполняются те условия процесса, задачи, для которых разработан алгоритм.
- **Гибкие алгоритмы**, например, стохастические, то есть вероятностные и эвристические.
- **Вероятностный** (стохастический) алгоритм даёт программу решения задачи несколькими путями или способами, приводящими к вероятному достижению результата.
- **Эвристический алгоритм** (от греческого слова «эврика») — алгоритм, использующий различные разумные соображения без строгих обоснований^[17].

- **Линейный алгоритм** — набор команд (указаний), выполняемых последовательно во времени друг за другом.
- **Разветвляющийся алгоритм** — алгоритм, содержащий хотя бы одно условие, в результате проверки которого может осуществляться разделение на несколько альтернативных ветвей алгоритма.
- **Циклический алгоритм** — алгоритм, предусматривающий многократное повторение одного и того же действия (одних и тех же операций). К циклическим алгоритмам сводится большинство методов вычислений, перебора вариантов. Цикл программы — последовательность команд (серия, тело цикла), которая может выполняться многократно.
- **Вспомогательный (подчинённый) алгоритм (процедура)** — алгоритм, ранее разработанный и целиком используемый при алгоритмизации конкретной задачи. В некоторых случаях при наличии одинаковых последовательностей указаний (команд) для различных данных с целью сокращения записи также выделяют вспомогательный алгоритм. На всех этапах подготовки к алгоритмизации задачи широко используется структурное представление алгоритма.
- **Структурная блок-схема, граф-схема алгоритма** — графическое изображение алгоритма в виде схемы связанных между собой с помощью стрелок (линий перехода) блоков — графических символов, каждый из которых соответствует одному шагу алгоритма. Внутри блока дается описание соответствующего действия. Графическое изображение алгоритма широко используется перед программированием задачи вследствие его наглядности, так как зрительное восприятие обычно облегчает процесс написания программы, её корректировки при возможных ошибках, осмысливание процесса обработки информации. Можно встретить даже такое утверждение: «Внешне алгоритм представляет собой схему — набор прямоугольников и других символов, внутри которых записывается, что вычисляется, что вводится в машину и что выдается на печать и другие средства отображения информации».

Нумерация алгоритмов

Нумерация алгоритмов играет важную роль в их исследовании и анализе^[18]. Поскольку любой алгоритм можно задать в виде конечного слова (представить в виде конечной последовательности символов некоторого алфавита), а множество всех конечных слов в конечном алфавите счётное, то множество всех алгоритмов также счётное. Это означает существование взаимно однозначного отображения между множеством натуральных чисел и множеством алгоритмов, то есть возможность присвоить каждому алгоритму номер.

Нумерация алгоритмов является одновременно и нумерацией всех алгоритмически исчисляемых функций, причем любая функция может иметь бесконечное количество номеров.

Существование нумерации позволяет работать с алгоритмами так же, как с числами. Особенно полезна нумерация в исследовании алгоритмов, работающих с другими алгоритмами.

Алгоритмически неразрешимые задачи

Формализация понятия алгоритма позволила исследовать существование задач, для которых не существует алгоритмов поиска решений. Впоследствии была доказана невозможность алгоритмического вычисления решений ряда задач, что делает невозможным их решение на любом вычислительном устройстве. Функцию f называют вычислимой (англ. *computable*), если существует машина Тьюринга, которая вычисляет значение f для всех элементов множества определения

функции. Если такой машины не существует, функцию f называют невычислимой. Функция будет считаться невычислимой, даже если существуют машины Тьюринга, способные вычислить значение для подмножества из всего множества входных данных^[19].

Случай, когда результатом вычисления функции f является логическое выражение «истина» или «ложь» (или множество $\{0, 1\}$), называют задачей, которая может быть решаемой или нерешаемой, в зависимости от вычислимости функции f ^[19].

Важно точно указывать допустимое множество входных данных, поскольку задача может быть решаемой для одного множества и нерешаемой для другого.

Одной из первых задач, для которой была доказана нерешаемость, является проблема остановки. Формулируется она следующим образом:

«Имея описание программы для машины Тьюринга, требуется определить, завершит ли работу программа за конечное время или будет работать бесконечно, получив некоторые входные данные.»^[20]

Доказательство неразрешимости проблемы остановки важно тем, что к ней можно свести другие задачи. Например, простую проблему остановки можно свести к задаче остановки на пустой строке (когда нужно определить для заданной машины Тьюринга, остановится ли она, будучи запущенной на пустой строке), доказав тем самым неразрешимость последней.^[19]

Анализ алгоритмов

Вместе с распространением информационных технологий увеличился риск программных сбоев. Одним из способов избежания ошибок в алгоритмах и их реализациях служат доказательства корректности систем математическими средствами.

Использование математического аппарата для анализа алгоритмов и их реализаций называют формальными методами. Формальные методы предусматривают применение формальных спецификаций и, обычно, набора инструментов для синтаксического анализа и доказательства свойств спецификаций. Абстрагирование от деталей реализации позволяет установить свойства системы независимо от её реализации. Кроме того, точность и однозначность математических утверждений позволяет избежать многозначности и неточности естественных языков^[21].

По гипотезе Ричарда Мейса, «избежание ошибок лучше устранения ошибок»^[22]. По гипотезе Хоара, «доказательство программ решает проблему корректности, документации и совместимости»^[23]. Доказательство корректности программ позволяет выявлять их свойства по отношению ко всему диапазону входных данных. Для этого понятие корректности было разделено на два типа:

- **Частичная корректность** — программа даёт правильный результат для тех случаев, когда она завершается.
- **Полная корректность** — программа завершает работу и выдаёт правильный результат для всех элементов из диапазона входных данных.

Во время доказательства корректности сравнивают текст программы со спецификацией желаемого соотношения входных-выходных данных. Для доказательств типа Хоара спецификация имеет вид утверждений, которые называют предусловиями и постусловиями. В совокупности с самой программой их ещё называют тройкой Хоара. Эти утверждения записывают как

$\{P\}Q\{S\}$

где P — это предусловия, должны выполняться перед запуском программы Q , а R — постусловия, истинные после завершения работы программы.

Формальные методы были успешно применены для широкого круга задач, в частности: разработке электронных схем, искусственного интеллекта, автоматических систем на железной дороге, верификации микропроцессоров, спецификации стандартов и спецификации и верификации программ^[24].

Время работы

Распространённым критерием оценки алгоритмов является время работы и порядок роста продолжительности работы в зависимости от объёма входных данных.^[25]

Для каждой конкретной задачи составляют некоторое число, которое называют её размером. Например, размером задачи вычисления произведения матриц может быть наибольший размер матриц-множителей, для задач на графах размером может быть количество ребер графа.

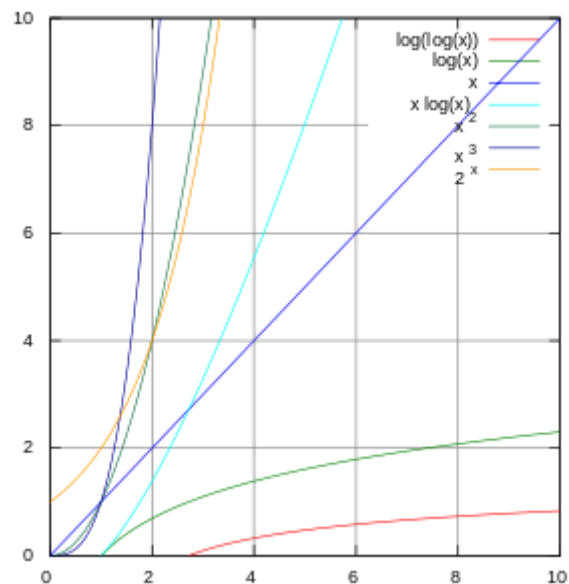
Время, которое тратит алгоритм как функция от размера задачи n , называют временной сложностью этого алгоритма $T(n)$. Асимптотику поведения этой функции при увеличении размера задачи называют *асимптотической временной сложностью*, а для её обозначения используют нотацию «O» большое. Например, если алгоритм обрабатывает входные данные размером n за время cn^2 , где c — некоторая константа, то говорят, что временная сложность такого алгоритма $O(n^2)$.

Асимптотическая сложность важна тем, что является характеристикой алгоритма, а не его конкретной реализации: «оптимизацией» операций, без замены алгоритма, можно изменить только мультипликативный коэффициент c , но не асимптотику. Как правило, именно асимптотическая сложность является главным фактором, который определяет размер задач, которые алгоритм способен обработать.

Часто во время разработки алгоритма пытаются уменьшить асимптотическую временную сложность для наихудших случаев. На практике же бывают случаи, когда достаточным является алгоритм, который «обычно» работает быстро.

Грубо говоря, анализ средней асимптотической временной сложности можно разделить на два типа: аналитический и статистический. Аналитический метод даёт более точные результаты, но сложен в использовании на практике. Зато статистический метод позволяет быстрее осуществлять анализ сложных задач^[26].

В следующей таблице приведены распространённые асимптотические сложности с комментариями^[27].



Графики функций, приведённых в таблице ниже.

Сложность	Комментарий	Примеры
$O(1)$	Устойчивое время работы не зависит от размера задачи	Ожидаемое время поиска в <u>хеш-таблице</u>
$O(\log \log n)$	Очень медленный рост необходимого времени	Ожидаемое время работы интерполирующего поиска n элементов
$O(\log n)$	Логарифмический рост — удвоение размера задачи увеличивает время работы на постоянную величину	Вычисление x^n ; <u>Двоичный поиск</u> в массиве из n элементов
$O(n)$	Линейный рост — удвоение размера задачи удвоит и необходимое время	Сложение/вычитание чисел из n цифр; <u>Линейный поиск</u> в массиве из n элементов
$O(n \log n)$	Линеаритмичный рост — удвоение размера задачи увеличит необходимое время чуть более, чем вдвое	<u>Сортировка слиянием</u> или <u>кучей</u> n элементов; нижняя граница сортировки сопоставлением n элементов
$O(n^2)$	Квадратичный рост — удвоение размера задачи увеличивает необходимое время в четыре раза	Элементарные <u>алгоритмы сортировки</u>
$O(n^3)$	Кубичный рост — удвоение размера задачи увеличивает необходимое время в восемь раз	Обычное умножение матриц
$O(c^n)$	Экспоненциальный рост — увеличение размера задачи на 1 приводит к c -кратному увеличению необходимого времени; удвоение размера задачи увеличивает необходимое время в квадрат	Некоторые <u>задачи коммивояжёра</u> , <u>алгоритмы поиска</u> полным перебором

Наличие исходных данных и некоторого результата

Алгоритм — это точно определённая инструкция, последовательно применяя которую к исходным данным, можно получить решение задачи. Для каждого алгоритма есть некоторое множество объектов, допустимых в качестве исходных данных. Например, в алгоритме деления вещественных чисел делимое может быть любым, а делитель не может быть равен нулю.

Алгоритм служит, как правило, для решения не одной конкретной задачи, а некоторого класса задач. Так, алгоритм сложения применим к любой паре натуральных чисел. В этом выражается его свойство массовости, то есть возможности применять многократно один и тот же алгоритм для любой задачи одного класса.

Для разработки алгоритмов и программ используется **алгоритмизация** — процесс систематического составления алгоритмов для решения поставленных прикладных задач. Алгоритмизация считается обязательным этапом в процессе разработки программ и решении задач на ЭВМ. Именно для прикладных алгоритмов и программ принципиально важны детерминированность, результативность и массовость, а также правильность результатов решения поставленных задач.

Алгоритм — это понятное и точное предписание исполнителю совершить последовательность действий, направленных на достижение цели.

Представление алгоритмов

Формы записи алгоритма:

- словесная или вербальная: на естественном языке;
- на алгоритмическом языке: языке программирования или псевдокоде;

- в машинном коде (код процессора ЭВМ);
- в математической нотации (см. выше представленные варианты);
- схематическая:
 - графическая (например, блок-схемы и ДРАКОН-схемы);
 - структурограммы (диаграммы Насси-Шнейдермана).

Обычно сначала (на уровне идеи) алгоритм описывается словами, но по мере приближения к реализации он обретает всё более формальные очертания и формулировку на языке, понятном исполнителю (например, машинный код).

Эффективность алгоритмов

Хотя в определении алгоритма требуется лишь конечность числа шагов, требуемых для достижения результата, на практике выполнение огромного количества шагов приводит к длительному выполнению программ, также обычно есть другие ограничения (на размер программы, на допустимые действия). В связи с этим вводят такие понятия, как сложность алгоритма (временная, по размеру программы, вычислительная и другие).

Для каждой задачи может существовать множество алгоритмов, приводящих к цели. Увеличение эффективности алгоритмов составляет одну из задач информатики, начиная с 1940-х годов в связи с этим построен ряд более эффективных в асимптотическом смысле алгоритмов для традиционных задач (например, алгоритмы быстрого умножения, алгоритм Чудновского для вычисления числа π).

Пример

Алгоритм Евклида — эффективный метод вычисления наибольшего общего делителя (НОД). Назван в честь греческого математика Евклида; один из древнейших алгоритмов, который используют до сих пор^[28].

Описан в «Началах» Евклида (примерно 300 лет до н. э.), а именно в книгах VII и X. В седьмой книге описан алгоритм для целых чисел, а в десятой — для длин отрезков.

Существует несколько вариантов алгоритма, ниже записанный в псевдокоде рекурсивный вариант:

```

функция нод(a, b)
  если b = 0
    возврат a
  иначе
    возврат нод(b, a mod b)

```

НОД чисел 1599 и 650:

Шаг 1	$1599 = 650 \cdot 2 + 299$
Шаг 2	$650 = 299 \cdot 2 + 52$
Шаг 3	$299 = 52 \cdot 5 + 39$
Шаг 4	$52 = 39 \cdot 1 + 13$
Шаг 5	$39 = 13 \cdot 3 + 0$



1599


Иллюстрация выполнения алгоритма Евклида для
вычисления НОД чисел 1599 и 650.

См. также

- [Список алгоритмов](#)
- [Метаалгоритм](#)
- [Теория алгоритмов](#)

Примечания

1. Семёнов А. Л. АЛГОРИТМ (<https://bigenc.ru/mathematics/text/1810305>). Большая российская энциклопедия. Электронная версия (2016). Дата обращения: 29 октября 2018.
2. Kleene 1943 in Davis 1965: 274
3. Rosser 1939 in Davis 1965: 225
4. Кнут, 2006, 1.1. Алгоритмы.
5. Robert Andrew Wilson, Frank C. Keil. The MIT Encyclopedia of the Cognitive Sciences (<https://books.google.com/books?id=-wt1aZrGXLYC&pg=PA11&>). — MIT Press, 2001. — С. 11. — 1106 с. — ISBN 9780262731447.
6. (Игошин, с. 317)
7. В. А. Успенский: «Математика — это гуманитарная наука» (https://elementy.ru/nauchno-populyarnaya_biblioteka/432214/V_A_Uspenskiy_Matematika_eto_gumanitarnaya_nauka) (рус.) // Троицкий вариант. — 2014. — 28 января (№ 2(146)). — С. 4—6.
8. Basics: The Turing Machine (with an interpreter!) (http://scienceblogs.com/goodmath/2007/02/basics_the_turing_machine_with_1.php). *Good Math, Bad Math* (9 февраля 2007).
Архивировано (https://www.webcitation.org/659x9ts9V?url=http://scienceblogs.com/goodmath/2007/02/basics_the_turing_machine_with_1.php) 2 февраля 2012 года.
9. (Игошин, раздел 33)
10. Энциклопедия кибернетики, т. 2, с. 90—91.
11. (Игошин, раздел 34)
12. «Probabilistic algorithms should not be mistaken with methods (which I refuse to call algorithms), which produce a result which has a high probability of being correct. It is essential that an algorithm produces correct results (discounting human or computer errors), even if this happens after a very long time». Henri Cohen. *A Course in Computational Algebraic Number Theory* (https://archive.org/details/coursecomputatio00cohe_507) (англ.). — Springer-Verlag, 1996. — Р. 2 (https://archive.org/details/coursecomputatio00cohe_507/page/n19). — ISBN 3-540-55640-0.

13. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein. Introduction to Algorithms (неопр.). — 2-е. — MIT Press, 2001. — С. 531. — ISBN 0-262-03293-7.
14. (Раздел 12, с. 12—22 в Atallah, 2010)
15. Dina Goldin, Peter Wegner. The origins of the Turing Thesis Myth (<ftp://ftp.cs.brown.edu/pub/techreports/04/cs04-14.pdf>) , CS-04-14, October 2004
16. Interactive Computation Is More Powerful Than Non Interactive (<http://c2.com/cgi/wiki?InteractiveComputationIsMorePowerfulThanNonInteractive>), c2.com
17. М. Гэри, Д. Джонсон, Вычислительные машины и труднорешаемые задачи, М.: Мир, 1982, С. 155.
18. (Игошин, раздел 36)
19. Peter Linz. An Introduction to Formal Languages and Automata (англ.). — Jones and Bartlett Publishers, 2000. — ISBN 0-7637-1422-4.
20. computability and complexity, *Encyclopedia of computer Science and Technology*, Facts On File, 2009, ISBN 978-0-8160-6382-6
21. (O'Regan, раздел 4.5)
22. (раздел 5.3.6 в Enders, 2003)
23. (раздел 5.3.7 в Enders, 2003)
24. (O'Regan, с. 119)
25. А. Ахо, Дж. Хопкрофт, Дж. Ульман. Построение и анализ вычислительных алгоритмов = The Design and Analysis of Computer Algorithms. — Москва: «Мир», 1979.
26. (раздел 11 в Atallah, 2010)
27. (раздел 1 в Atallah, 2010)
28. Knuth D. The Art of Computer Programming, Volume 2: Seminumerical Algorithms (англ.). — 3rd. — Addison-Wesley, 1997. — ISBN 0201896842.

Литература

- Томас Х. Кормен, Чарльз И. Лейзерсон, Рональд Л. Ривест, Клиффорд Штайн. Алгоритмы: построение и анализ, 3-е издание = Introduction to Algorithms, Third Edition. — М.: «Вильямс», 2013. — 1328 с. — ISBN 978-5-8459-1794-2.
- Дональд Кнут. Искусство программирования, том 1. Основные алгоритмы = The Art of Computer Programming, vol. 1. Fundamental Algorithms. — 3-е изд.. — М.: «Вильямс», 2006. — С. 720. — ISBN 0-201-89683-4.
- Томас Х. Кормен. Алгоритмы: вводный курс = Algorithms Unlocked. — М.: «Вильямс», 2014. — 208 с. — ISBN 978-5-8459-1868-0.
- Игошин В. И. Математическая логика и теория алгоритмов. — 2-е изд., стер.. — М.: ИЦ «Академия», 2008. — 448 с. — ISBN 5-7695-1363-2.

Ссылки

- «Слово „алгоритм“: происхождение и развитие», В. В. Шилов, Журнал «Потенциал» — источник исторических сведений.
- Об алгоритме (<http://www.krugosvet.ru/articles/125/1012581/1012581a1.htm>) (недоступная ссылка с 22-05-2013 [2916 дней] — *история* (https://web.archive.org/web/*/http://www.krugosvet.ru/articles/125/1012581/1012581a1.htm), *копия* (<https://web.archive.org/web/20070712/http://www.krugosvet.ru/articles/125/1012581/1012581a1.htm>)) в энциклопедии «Кругосвет»

Эта страница в последний раз была отредактирована 24 апреля 2021 в 03:48.

Текст доступен по лицензии Creative Commons Attribution-ShareAlike; в отдельных случаях могут действовать дополнительные условия.

Wikipedia® — зарегистрированный товарный знак некоммерческой организации Wikimedia Foundation, Inc.