2025, **26** (4), 548–568. doi 10.26089/NumMet.v26r436

doi 10.26089/NumMet.v26r436

УДК 004.421;
004.272.2

# Обзор методов описания структуры алгоритмов

**А. С. Антонов**
Московский государственный университет имени М. В. Ломоносова,
Научно-исследовательский вычислительный центр,
Москва, Российская Федерация
ORCID: 0000-0003-2820-7196, e-mail: asa@parallel.ru

**Аннотация:** Алгоритмы являются основой всего вычислительного процесса в рамках традиционной цепочки "задача–метод–алгоритм–реализация", а эффективное использование их свойств, структуры, особенностей и характеристик может дать многократное сокращение времени их реализации на конкретной вычислительной системе. В данной обзорной статье рассматриваются методы, используемые для описания структуры вычислительных алгоритмов. Также в статье рассматриваются доступные в сети Интернет коллекции описаний алгоритмов, дается краткая характеристика, алгоритмы из каких областей и каким образом в них описываются.

**Ключевые слова:** задача, метод, алгоритм, реализация, параллелизм, суперкомпьютер, AlgoWiki.

**Благодарности:** Исследование выполнено за счет гранта Российского научного фонда № 25–11–00181, https://rscf.ru/project/25-11-00181/.

**Для цитирования:** Антонов А.С. Обзор методов описания структуры алгоритмов // Вычислительные методы и программирование. 2025. **26**, № 4. 548–568. doi 10.26089/NumMet.v26r436.

# Review of methods for describing the structure of algorithms

**Alexander S. Antonov**
Lomonosov Moscow State University, Research Computing Center,
Moscow, Russia
ORCID: 0000-0003-2820-7196, e-mail: asa@parallel.ru

**Abstract:** Algorithms are the basis of the entire computational process within the traditional chain "problem–method–algorithm–implementation", and the effective use of their properties, structure, features and characteristics can provide a multiple reduction in the time of their implementation on a specific computing system. This review paper examines the methods used to describe the structure of computational algorithms. It also examines collections of algorithm descriptions available online, providing a brief description of the areas from which algorithms are described and how they are described.

**Keywords:** problem, method, algorithm, implementation, parallelism, supercomputer, AlgoWiki.

**Acknowledgements:** The study was supported by a grant from the Russian Science Foundation No 25–11–00181, https://rscf.ru/en/project/25-11-00181/.

**For citation:** A. S. Antonov, "Review of methods for describing the structure of algorithms," Numerical Methods and Programming. **26** (4), 548–568 (2025). doi 10.26089/NumMet.v26r436.

2025, **26** (4), 548–568. doi 10.26089/NumMet.v26r436

**1. Введение. Понятие алгоритма.** Происхождение понятия “алгоритм” не вполне очевидно, но большинство исследователей считает, что слово “алгоритм” происходит от имени арабского ученого аль-Хорезми (Al-Khwarizmi), жившего в 8–9 веках нашей эры [1]. Известно много разных определений понятия “алгоритм”, в одном из самых известных Дональд Кнут (Donald Knuth) к определению “алгоритм — это ... набор конечного числа правил, задающих последовательность выполнения операций для решения задачи определенного типа” (“a finite set of rules that gives a sequence of operations for solving a specific type of problem”) добавляет описание таких особенностей, как конечность, определенность, ввод, вывод и эффективность [2].

Выдающийся советский математик А. А. Марков сформулировал свое определение этого понятия: “Алгоритм — это точное предписание, определяющее вычислительный процесс, ведущий от варьируемых исходных данных к искомому результату” [3]. Далее А. Н. Колмогоров уточнил и дополнил это определение: “Алгоритмом принято называть систему вычислений, которая для некоторого класса математических задач из записи A “условий” задачи позволяет при помощи однозначно определенной последовательности операций, совершаемых “механически”, без вмешательства творческих способностей человека, получить запись B “решения” задачи” [4].

Более четкие определения понятия “алгоритм” дают справочные издания, например философский словарь под редакцией М. М. Розенталя: “Алгоритм — точное предписание о выполнении в определенном порядке некоторой системы операций, ведущих к решению всех задач данного типа” [5], в IEEE Standard Dictionary of Electrical and Electronics Terms сказано: “Алгоритм — это предписанный набор четко определенных правил или процессов для решения задачи за конечное число шагов” (“Algorithm is a prescribed set of well-defined rules or processes for the solution of a problem in a finite number of steps”) [6], a National Institute of Standards and Technology (NIST) в словаре Dictionary of Algorithms and Data Structures [7] определяет понятие “алгоритм” как “Вычислимая последовательность шагов для достижения желаемого результата” (“A computable set of steps to achieve a desired result”).

В учебниках понятие “алгоритм” определяют более широко, не ограничиваясь только областью вычислений. Так, в пособии для студентов вуза приводится такое определение: “Алгоритм — конечный набор инструкций, приводящий от начальных данных к искомому результату” [8], а в учебнике “Информатика и информационные технологии” для 9–11 классов школы сказано: “Алгоритм — строго детерминированная последовательность действий, описывающая процесс преобразования объекта из начального состояния в конечное, записанная с помощью понятных исполнителю команд” [9].

Приведем также несколько определений понятия “алгоритм” из других источников:

- “Алгоритм — это конечная, определенная, эффективная процедура с некоторым результатом” (“An algorithm is a finite, definite, effective procedure, with some output”) [10].
- “Алгоритм (algorithm) — это любая корректно определенная вычислительная процедура, на вход (input) которой подается некоторая величина или набор величин, и результатом выполнения которой является выходная (output) величина или набор значений” [11].
- “Алгоритм — это процедура, которая принимает любой из возможных входных экземпляров [задачи] и преобразует его в соответствии с требованиями, указанными в условии задачи” [12].

Постараемся выделить из этих определений общее и главное, что присуще вычислительному алгоритму независимо от особенностей конкретного определения:

- Алгоритм предназначен для решения некоторой задачи (класса задач).
- Алгоритм состоит из множества некоторых действий (операций).
- Структура алгоритма задается последовательностью этих действий (операций).
- Выполняя конечную последовательность предписанных действий, исполнитель приходит от входных данных к результатам.

Любой предлагаемый способ описания структуры вычислительных алгоритмов должен позволять описывать перечисленные выше сущности. Также существенным в описании алгоритмов является выделение свойств, связанных с параллелизмом, необходимых для их эффективной реализации на параллельных вычислительных системах как настоящих, так и будущих.

Дальнейшая структура данной обзорной статьи следующая. В разделе 2 рассматривается связь между решением вычислительной задачи и выбором алгоритма. Раздел 3 посвящен обзору существующих способов описания структуры вычислительных алгоритмов. В разделе 4 даются краткие характеристики коллекций описаний алгоритмов в сети Интернет. В заключении (раздел 5) подводятся итоги и делаются выводы из выполненного обзора.

2025, **26** (4), 548–568. doi 10.26089/NumMet.v26r436

**2. От методов решения задачи к выбору алгоритма.** Понятно, что алгоритмы появились не сами по себе, их возникновение обусловлено необходимостью решения конкретных вычислительных задач. В книге [11] сказано: "Алгоритм также можно рассматривать как инструмент, предназначенный для решения корректно поставленной вычислительной задачи (computational problem). В постановке задачи в общих чертах задаются отношения между входом и выходом. В алгоритме описывается конкретная вычислительная процедура, с помощью которой удается добиться выполнения указанных отношений".

Для того чтобы перейти от уровня исследования поставленной задачи к уровню выбора алгоритма, надо выбрать математический метод решения данной задачи. Так возникает уровень метода, промежуточный между уровнями задачи и алгоритма. Например, для решения эллиптических уравнений можно использовать прямой метод, основанный на преобразовании Фурье, или один из итерационных методов. Обычно в описании метода содержится его математическое описание, обоснование сходимости метода, получение решения, обоснование единственности решения и т.д. Каждый из методов решения задачи обладает своими характеристиками, которые оказывают непосредственное влияние на дальнейший выбор алгоритма и его реализацию в виде программы. В [13] термин "алгоритм" поясняется следующим образом: "Термин *алгоритм* используется в компьютерных науках для описания метода решения задачи, пригодного для реализации в виде компьютерной программы".

С другой стороны, каждый алгоритм может иметь множество различных программных реализаций, обладающих своими свойствами и предназначенных для разных программно-аппаратных платформ. Свойства и характеристики программных реализаций в значительной степени характеризуются анализом динамики их исполнения на соответствующих вычислительных системах.

Таким образом, логично формируется цепочка понятий "задача–метод–алгоритм–реализация", которая может являться основой для описания любой предметной области, в которой применяются вычислительные алгоритмы. На каждом из этапов этой цепочки принимаются важные решения, в конечном итоге кардинально влияющие на время получения решения поставленной задачи. Но фундаментальную роль в процессе решения вычислительных задач играют именно алгоритмы, что подчеркивает также Д. Кнут: "Алгоритмы — это нити, связывающие воедино большинство разделов компьютерной науки" ("Algorithms are the threads that tie together most of the subfields of computer science"). В данной статье рассматриваются способы описания структуры именно вычислительных алгоритмов.

**3. Способы описания структуры алгоритмов.** Возможных способов описания структуры вычислительных алгоритмов довольно много. Надо иметь в виду, что все авторы понимают описание структуры алгоритмов по-своему, описывают с разными целями, используют для их описания разные методики. В данном разделе рассмотрим наиболее часто используемые из известных способов.

**3.1. Словесное описание.** Словесное описание предполагает детализацию работы алгоритма на естественном языке. Это дает наибольшую свободу, но приводит к тому, что без наложения дополнительных ограничений описания даже похожих алгоритмов могут существенно отличаться друг от друга, что приводит к практической невозможности их сопоставления.

Словесное описание структуры алгоритмов используется в Открытой энциклопедии свойств алгоритмов AlgoWiki [14] и во многих других коллекциях описаний структуры алгоритмов, приводимых в разделе 4. В энциклопедии AlgoWiki, кроме того, различаются "Общее описание алгоритма" и "Математическое описание алгоритма". Оба эти раздела заполняются на естественном языке, но первый предполагается практически полностью описательным, а во втором используется более формальное изложение с выражением шагов алгоритма в виде математических формул. На рис. 1 показаны эти два способа описания алгоритма для метода Гивенса (вращений) QR-разложения квадратной матрицы[1]

В классической книге [2] алгоритмы также описываются на частично формализованном естественном языке. Задаваемые ограничения позволяют делать описания алгоритмов в чем-то похожими, легче понимаемыми и где-то приближают такие описания к описаниям с использованием псевдокода (раздел 3.6.1). Классическим источником словесных описаний множества вычислительных алгоритмов является также книга [15].

**3.2. Блок-схемы.** Очень часто для описания структуры алгоритмов используют те или иные варианты блок-схем (flowcharts). В блок-схемах каждому типу операций, выполняемых в алгоритме, соответ-

[1]URL wiki-страницы https://algowiki-project.org/ru/Метод_Гивенса_(вращений)_QR-разложения_квадратной_матрицы_(вещественный_точечный_вариант).

2025, **26** (4), 548–568. doi 10.26089/NumMet.v26r436

### 1.1 Общее описание алгоритма [править]

**Метод Гивенса** (в отечественной математической литературе называется также **методом вращений**) используется для разложения матриц в виде $A = QR$ ($Q$ - унитарная, $R$ — правая треугольная матрица)[1]. При этом матрица $Q$ хранится и используется не в явном виде, а в виде произведения матриц вращения. Каждая из матриц вращения (Гивенса)

номера столбцов: $i-1$ $i$ $i+1$ $j-1$ $j$ $j+1$

$$T_{ij} = \begin{bmatrix} 1 & \cdots & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \ddots & & \vdots & & & \vdots & & \vdots & & \\ 0 & \cdots & 1 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ 0 & \cdots & 0 & c & 0 & \cdots & 0 & -s & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 0 & 1 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots & & \ddots & & \vdots & & \vdots & \vdots \\ 0 & \cdots & 0 & 0 & 0 & \cdots & 1 & 0 & 0 & \cdots & 0 \\ 0 & \cdots & 0 & s & 0 & \cdots & 0 & c & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & & \vdots & & & \vdots & & & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 1 \end{bmatrix}$$

может быть определена парой индексов и одним параметром. Это позволяет в стандартной реализации метода Гивенса хранить результаты разложения на месте матрицы $A$ без использования дополнительных массивов. $QR$-разложение матрицы $A$ может быть использовано в разных целях: как для решения СЛАУ вида $Ax = b$, так и в качестве одного из шагов так называемого $QR$-алгоритма нахождения собственных чисел матрицы.

На каждом шаге алгоритма две строки преобразуемой матрицы подвергаются преобразованию вращения. При этом параметр такого преобразования подбирается так, чтобы один из поддиагональных элементов преобразуемой матрицы стал нулевым. Сначала в нуль последовательно обращаются элементы 1-го столбца, затем 2-го, и т.д. до n-1-го, после чего получившаяся матрица - это и есть матрица $R$. Сам шаг алгоритма распадается на две части: подбор параметра вращения и выполнение вращения над двумя строками матрицы. Поскольку слева от "рабочего" столбца элементы вращаемых строк равны 0, то там выполнять вращение не нужно. Вращение элементов строк в текущем столбце выполняется одновременно с подбором параметра вращения. Таким образом, вторая часть шага заключается в выполнении вращения двумерных векторов, составленных из элементов вращаемых строк в столбцах справа от "рабочего". Каждое такое вращение эквивалентно перемножению двух комплексных чисел (или в сумме 4 умножениям, 1 сложению и 1 вычитанию действительных), одно из которых имеет модуль 1. Подбор параметра вращения по двум элементам "рабочего" столбца является более сложной операцией, что связано в том числе и с необходимостью минимизации влияния ошибок округления. Обычно для хранения матрицы вращения используется тангенс половинного угла $t$, с которым простыми формулами (т.н. "боевыми формулами тригонометрии") связаны косинус $c$ и синус $s$ самого угла:

$c = (1-t^2)/(1+t^2), s = 2t/(1+t^2)$

Именно $t$ обычно и хранится в соответствующем элементе массива.

### 1.2 Математическое описание алгоритма [править]

Для получения $QR$-разложения квадратной матрицы $A$ последнюю приводят к правой треугольной ($R$ - от слова right) последовательностью умножений её слева на матрицы вращения $T_{12}, T_{13}, \ldots, T_{1n}, T_{23}, T_{24}, \ldots, T_{2n}, \ldots, T_{n-2n}, T_{n-1n}$.

Каждая из матриц $T_{ij}$ задаёт вращение в 2-мерном подпространстве, связанном с $i$-й и $j$-й компонентами столбцов, остальные компоненты не меняются. При этом вращение подбирается так, чтобы элемент новой матрицы в $j$-й строке и $i$-м столбце стал нулевым. Поскольку вращения и тождественные преобразования нулевых векторов оставляют их нулевыми, то последующие вращения сохраняют полученные ранее нули слева и сверху от текущего обнуляемого элемента.

В конце процесса имеем $R = T_{n-1n}T_{n-2n}T_{n-2n-1}\ldots T_{13}T_{12}A$.

Так как матрицы вращения унитарны, то естественно получается $Q = (T_{n-1n}T_{n-2n}T_{n-2n-1}\ldots T_{13}T_{12})^* = T_{12}^*T_{13}^*\ldots T_{1n}^*T_{23}^*T_{24}^*\ldots T_{2n}^*\ldots T_{n-2n}^*T_{n-1n}^*$ и $A = QR$.

В вещественном случае матрицы вращения ортогональны и $Q = (T_{n-1n}T_{n-2n}T_{n-2n-1}\ldots T_{13}T_{12})^T = T_{12}^TT_{13}^T\ldots T_{1n}^TT_{23}^TT_{24}^T\ldots T_{2n}^T\ldots T_{n-2n}^TT_{n-1n}^T$.

Для завершения математического описания остаётся записать вычисление[2] матрицы вращения $T_{ij}$ и формулы её умножения на текущую промежуточную матрицу.

Пусть в позиции $(i,i)$ преобразуемой матрицы стоит число $x$, а в позиции $(i,j)$ - число $y$.

Тогда для минимизации влияния ошибок округления сначала вычисляется равномерная норма вектора $z = max(|x|,|y|)$.

Если она равна 0, то поворот не требуется: $t = s = 0, c = 1$.

Если $z = |x|$, то вычисляем $y_1 = y/x$ и далее $c = \frac{1}{\sqrt{1+y_1^2}}$, $s = -cy_1$, $t = \frac{1-\sqrt{1+y_1^2}}{y_1}$, а в позиции $(i,i)$ после поворота будет число $x\sqrt{1+y_1^2}$.

Если же $z = |y|$, то вычисляем $x_1 = x/y$ и далее $t = x_1 - x_1^2 sign(x_1)$, $s = \frac{sign(x_1)}{\sqrt{1+x_1^2}}$, $c = sx_1$, а в позиции $(i,i)$ после поворота будет число $y\sqrt{1+x_1^2}sign(x)$.

После получения элементов $c$ и $s$ матрицы вращения $T_{ij}$ запись преобразования для каждого столбца правее $i$-го выглядит просто. Если в $k$-м столбце в позиции с номером $i$ стоит x, а в позиции с номером $j$ стоит y, то их новые значения будут $cx - sy$ и $sx + cy$, соответственно, как если бы комплексное число с действительной частью $x$ и мнимой $y$ умножили на комплексное число $(c, s)$.

Рис. 1. Общее и математическое описание алгоритма в рамках энциклопедии AlgoWiki

Fig. 1. General and mathematical description of the algorithm in the AlgoWiki encyclopedia

ствует геометрическая фигура, представленная в виде некоторого символа из предопределенного набора. Данные символы соединяются стрелками, определяющими последовательность выполнения действий алгоритма.

Блок-схемы для описания структуры алгоритмов стали использоваться уже давно, они встречались еще в трудах Джона фон Неймана [16].

В книге [2] описания алгоритмов на естественном языке зачастую сопровождаются блок-схемами, дополненными примечаниями, которые доказывают правильность работы алгоритма.

В книге [8] для похожего понятия используется термин “граф-схема алгоритма (ГСА)”.

Правила формального описания схем алгоритмов задаются в России ГОСТ 19.701–90 “Схемы алгоритмов, программ, данных и систем” [17]. В нем регламентированы конфигурация и размеры блоков, а также порядок графического оформления блок-схем. На рис. 2 приведен пример блок-схемы из описания данного ГОСТ.

2025, **26** (4), 548–568. doi 10.26089/NumMet.v26r436

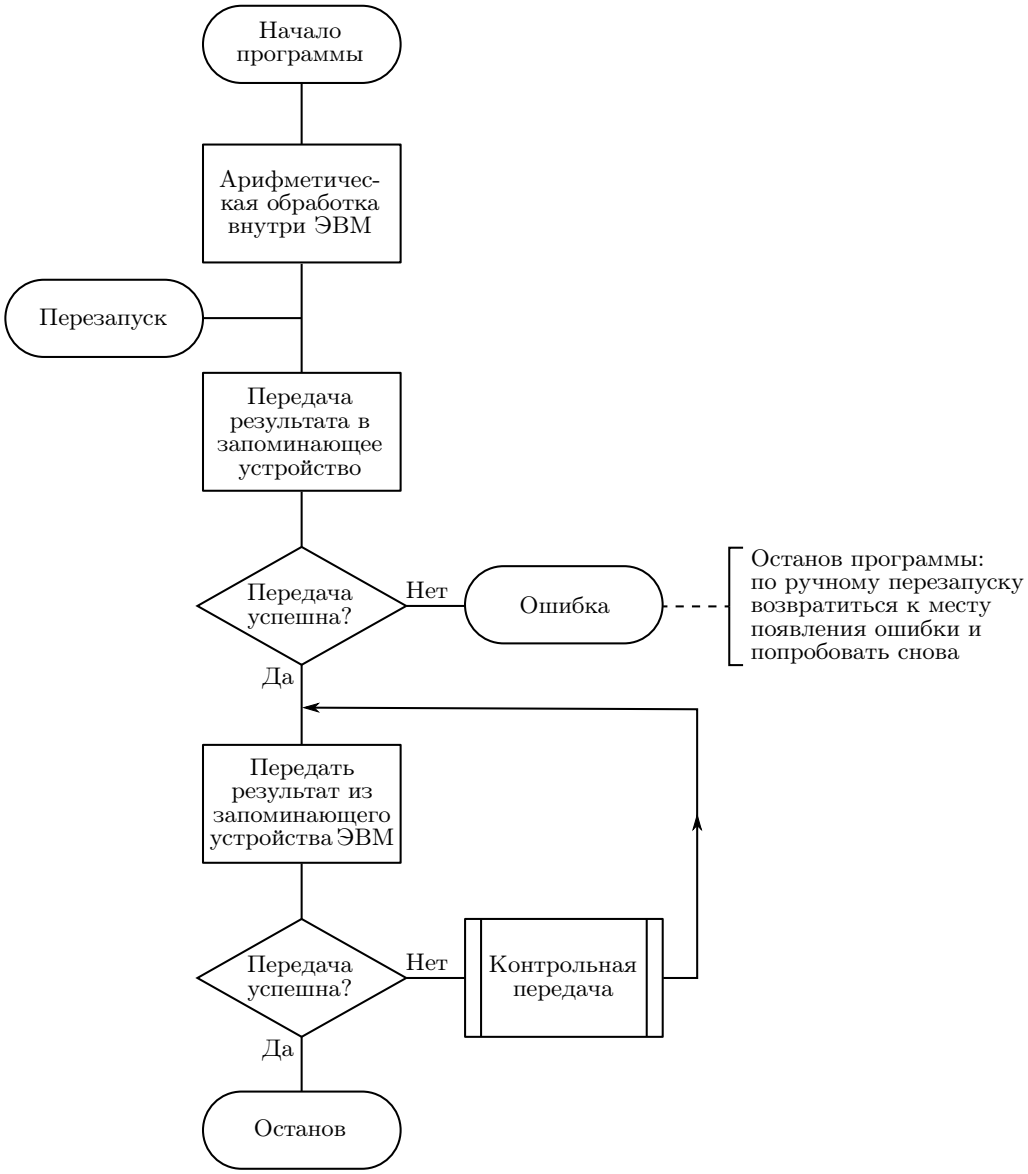


Рис. 2. Пример блок-схемы из описания ГОСТ 19.701–90 [17]

Fig. 2. Flowchart example from the description of GOST 19.701–90 [17]

ГОСТ 19.701–90 соответствует международному стандарту оформления алгоритмов ISO 5807:1985 “Information processing — Documentation symbols and conventions for data, program and system flowcharts, program network charts and system resources charts” [18], поэтому блок-схемы алгоритмов, оформленные согласно ГОСТ 19.701–90, и в других странах должны пониматься однозначно.

Использование блок-схем для описания структуры алгоритма гораздо нагляднее, чем использование естественного языка, а также позволяет избежать неоднозначности трактования. Приводимые в описаниях алгоритмов блок-схемы в целом соответствуют международным стандартам, что делает их достаточно легко понимаемыми. Для больших и сложных алгоритмов блок-схемы могут оказаться слишком громоздкими. Не всегда просто отобразить в виде блок-схемы особенности параллельных алгоритмов (в ГОСТ 19.701–90 для обозначения параллельных действий предусмотрена только одна простая конструкция), для этого их иногда расширяют и дополняют новыми сущностями. Также для параллельных алгоритмов могут больше подойти диаграммы деятельности (Activity diagram) на языке UML (Unified Modeling Language) [19], в которых, например, для описания параллельных областей могут использоваться управляющие узлы разделения (fork) и слияния (join).

2025, **26** (4), 548–568. doi 10.26089/NumMet.v26r436

В некоторых странах получили распространение диаграммы Насси–Шнейдермана [20], в которых не используются соединительные стрелки, а структура программы передается вложенностью структурных блоков.

**3.3. Вычислимые функции, конечные автоматы.** Несколько вариантов описания структуры алгоритмов связаны с введением формальных определений этого понятия при помощи специальных математических конструкций (машина Поста [21], машина Тьюринга [22], рекурсивно-вычислимые функции Черча [23]) и с доказательством утверждений об эквивалентности этих моделей. А. Н. Колмогоров показывает [4], что понятие алгоритма сводится к понятию вычислимой функции, причем в любой из этих формальных моделей.

Перечисленные формальные модели в принципе пригодны для описания структуры алгоритмов, однако являются скорее теоретическими конструкциями, крайне редко используемыми для этой цели на практике.

**3.4. Типовые алгоритмические структуры (шаблоны).** Один из подходов к описанию структуры алгоритмов заключается в выделении так называемых типовых алгоритмических структур — структур, на основе которых построено значительное число задач из различных предметных областей. Эти структуры хорошо известны или могут быть достаточно легко изучены. Основная идея заключается в том, что изучение процесса отображения относительно небольшого числа данных структур на архитектуру целевой высокопроизводительной вычислительной системы позволит сделать выводы об отображении множества реальных задач, построенных на их основе.

В работах исследователей из Berkeley University of California было предложено [24, 25] 13 таких типовых алгоритмических структур (в их исследованиях — “dwarfs”, позже “motifs”) для обозначения алгоритмических конструкций, которые выделяют некоторый часто встречаемый шаблон вычислений и коммуникаций.

В других исследованиях подобные конструкции называют “шаблоны” или “скелетоны” (“patterns” / “skeletons”) [26, 27]. При этом алгоритм составляется из наборов таких шаблонов и их композиций. Ясно, что основным недостатком такого подхода является невозможность реализации произвольного алгоритма.

**3.5. Языки программирования.** Несмотря на принципиальное отличие алгоритмов и программ, очень часто именно программы на каком-то языке программирования используют для описания структуры алгоритма. Н. Вирт писал: “... программы суть конкретные формулировки абстрактных алгоритмов, основанные на конкретных представлениях и структурах данных” [28].

Во многих источниках для описания структуры алгоритмов используют язык C (C++) или некоторые его подмножества [12, 13, 29, 30 и др.]. В книге “Numerical Recipes” [15] реализации алгоритмов приводятся также на языке C++, хотя в предыдущих редакциях приводились программы на языках программирования C, Фортран и Паскаль.

В книге [31] для записи большого количества примеров используется язык Фортран, в [28] — Паскаль / Модула–2 / Оберон.

Для описания параллельных алгоритмов к базовому языку программирования часто добавляют технологии OpenMP [32], MPI [33] или CUDA [34].

Иногда используют специальные языки параллельного программирования, например в [35] используется язык Nesl, а в книге [36] — язык C*.

Использование традиционных языков программирования для описания структуры алгоритмов делает описание формализованным и понятным для любого, знакомого с синтаксисом и семантикой данного языка. Однако традиционные языки программирования навязывают использование конструкций, не относящихся напрямую к описанию алгоритма. Поэтому данный подход хорош как иллюстративный вариант, но не всегда подходит для объяснения сути алгоритма.

**3.6. Специальные языки для описания алгоритмов.** Иногда для описания структуры алгоритмов используются специально созданные языки. Они не являются языками программирования, описанные с их использованием конструкции не могут быть откомпилированы и выполнены на реальном компьютере. Однако их создают таким образом, чтобы максимально удобно описывать как раз структуру вычислительных алгоритмов.

2025, **26** (4), 548–568. doi 10.26089/NumMet.v26r436

*§ 3.6.1. Псевдокод.* Одним из наиболее распространенных способов описания структуры алгоритмов является использование псевдокода. Строгого определения, что же такое псевдокод, не существует, в книге [12] не вполне серьезно сказано: “Псевдокод — язык программирования, который никогда не выдает сообщений о синтаксических ошибках”.

От источника к источнику используемые псевдокоды отличаются [11, 30, 37–40], но по сути почти всегда похожи друг на друга. В некоторых случаях псевдокод рассматривается как некое ограничение на традиционный язык программирования. Например, в [39] сказано: “Наш псевдокод можно рассматривать как краткую запись для подмножества C++” (“Our pseudocode can be viewed as a concise notation for a subset of C++”).

В книге [37] сказано: “Особенности псевдокода не играют существенной роли, если он реализует основные структуры управления, общие для всех алгоритмов. Такие структуры, как циклы вида `for` или `while`, механизм ветвления вида `if`, `case` или `switch`, присутствуют в любом языке программирования, и любой такой язык подойдет для наших целей”. В этом смысле псевдокоды, используемые разными авторами, примерно эквивалентны. Рис. 3 демонстрирует пример псевдокода из книги [11].

```
INSERTION_SORT(A)
1  for j ← 2 to length[A]
2     do key ← A[j]
3        ▷ Вставка элемента A[j] в отсортированную
         ▷ последовательность A[1..j − 1]
4        j ← j − 1
5        while j > 0 и A[i] > key
6           do A[i + 1] ← A[i]
7              i ← i − 1
8        A[i + 1] ← key
```

Рис. 3. Пример псевдокода (из книги [11])

Fig. 3. Pseudocode example (from the book [11])

Псевдокод представляет собой что-то среднее между естественным языком и традиционным языком программирования. Алгоритмы на псевдокоде читаются лучше, чем традиционные программы. Но по сравнению с естественным языком псевдокод накладывает дополнительные ограничения и включает математическую нотацию.

Для выражения параллелизма в алгоритмах псевдокод часто модифицируется, чтобы включать соответствующие механизмы [36, 41–43]. В книге [38] в псевдокод добавляются также абстрактные конструкции для реализации модели пересылок данных send-receive.

Существует такая разновидность псевдокода, как Program Design Language (PDL) [44], которая имеет более строгие правила описания структуры алгоритма, тем самым приближаясь к традиционным языкам программирования.

Использование для описания структуры алгоритмов псевдокода делает описание более строгим, чем просто словесное описание (раздел 3.1), но в то же время не требует знания и строгого соблюдения синтаксиса конкретного языка программирования (раздел 3.5). При этом авторы описания сохраняют необходимую им гибкость, модифицируя псевдокод в зависимости от тех свойств алгоритма, которые собираются продемонстрировать. Все это определяет широкое использование псевдокодов на практике.

*§ 3.6.2. Нотация ДРАКОН.* Нотация ДРАКОН (Дружелюбный русский алгоритмический язык, который обеспечивает наглядность) [45] была предложена для описания алгоритмов в критических областях, в частности, в системах реального времени. Она разрабатывалась как часть космической программы “Буран”, начиная с 1986 г. Это визуальный язык, который формализует блок-схемы алгоритмов (раздел 3.2) для их использования в разработке реальных приложений. Используемые в ней “дракон-схемы” устраняют некоторые недостатки блок-схем. На рис. 4 показан пример простой дракон-схемы для цикла while. Нотация ДРАКОН имеет как графический, так и текстовый синтаксис, что расширяет ее область применимости, обеспечивая гибкость и универсальность используемых выразительных средств.

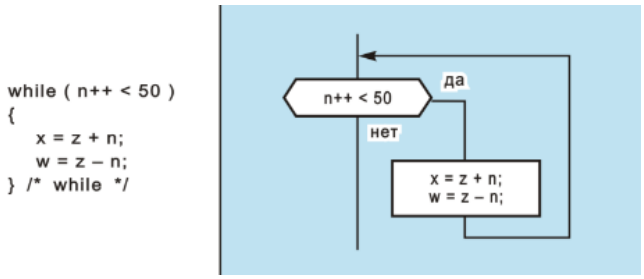


Рис. 4. Пример дракон-схемы для цикла while (из книги [45])

Fig. 4. An example of a drakon diagram for a while loop (from the book [45])

2025, **26** (4), 548–568. doi 10.26089/NumMet.v26r436

*§ 3.6.3. Язык Algolang.* В рамках создания системы визуализации информационных графов AlgoView [46] был предложен язык описания информационной структуры, названный Algolang [47]. Он не является языком программирования, не содержит привычных объектов — переменных, операций присваивания, циклов и т.д. Фактически на языке Algolang описывается информационный граф, но без необходимости описания каждой отдельной вершины или дуги. В целом, структура описания информационного графа соотносится с описанием структуры алгоритма любым другим способом, например на традиционном языке программирования.

Рис. 5 демонстрирует описание на языке Algolang информационной структуры небольшого фрагмента программы на языке C. В описании не остается фактически ничего, кроме структуры информационного графа: операторы `<block>` задают множества вершин графа, а операторы `<in>` — координаты начала дуги, ведущей в данную вершину.

Таким образом, язык Algolang позволяет абстрагироваться от любых особенностей традиционных языков программирования, в то же время точно и компактно описывая информационную структуру ана-

```
for(i = 2; i <= n+1; ++i)
    C[i] = C[i-2] * e;
for(i = 2; i <= n+1; ++i)
    for(j = 2; j <= m+1; ++j)
       B[i][j] = B[i-1][j-1] + C[i];
for(i = 2; i <= n+1; ++i){
    A[i][1][1]=B[i][m];
    for(j = 2; j <= m+1; ++j)
       for(k = 1; k <= n-1; ++k)
          A[i][j][k] = A[i][j][k] + A[i][j-1][k];
}
```

```
<?xml version = "1.0" encoding = "UTF-8"?>
<algo>
    <params>
        <param name = "N" type = "int" value = "5"></param>
        <param name = "M" type = "int" value = "4"></param>
    </params>
    <block id = "0" dims = "1">
        <arg name = "i" val = "2..N+1"></arg>
        <vertex condition = "" type = "1">
            <in src = "i - 2"></in>
        </vertex>
    </block>
    <block id = "1" dims = "2">
        <arg name = "i" val = "2..N+1"></arg>
        <arg name = "j" val = "2..M+1"></arg>
        <vertex condition = "" type = "1">
            <in src = "i - 1, j - 1"></in>
            <in bsrc = "0" src = "i"></in>
        </vertex>
    </block>
    <block id = "2" dims = "3">
        <arg name = "i" val = "2..N+1"></arg>
        <arg name = "j" val = "1..M+1"></arg>
        <arg name = "k" val = "1..N-1"></arg>
        <vertex condition = "(j == 1) and (k == 1)" type = "1">
            <in bsrc = "1" src = "i, M"></in>
        </vertex>
        <vertex condition = "(j > 1) or (k > 1)" type = "1">
            <in src = "i, j - 1, k"></in>
        </vertex>
    </block>
</algo>
```

Рис. 5. Пример на языке C и его описание на языке Algolang (из статьи [47])

Fig. 5. Example in C language and its description in Algolang language (from the paper [47])

2025, **26** (4), 548–568. doi 10.26089/NumMet.v26r436

лизируемого алгоритма. В отличие от псевдокода, описание на языке Algolang строго описано и может быть проанализировано, а соответствующий информационный граф автоматически визуализируется в рамках системы AlgoView [48].

**3.7. Графовые модели.** Еще одним способом описания структуры алгоритма является представление его в виде графа. Существует достаточно много различных графовых представлений, отличающихся тем, чему соответствуют вершины и дуги конкретного графа. В [31] показано, что информационная структура алгоритма определяется графом информационных зависимостей (называемым также “графом алгоритма”). Он представляет собой граф, вершины которого соответствуют срабатываниям операций алгоритма, а дуги — информационным зависимостям между ними.

Похожие объекты используются и в других исследованиях. Так, в [30] похожий объект называется “Dependence Graph (DG)”. В некоторых источниках используют понятие “Directed Acyclic Graphs (DAG)”. Например, в [38] отличие DAG заключается в том, что его вершины соответствуют либо операциям программы, либо их операндам, а дуги идут из вершин, соответствующих операндам, в вершины, соответствующие использующим их операциям. В книге [36] для информационного графа используется термин “Computation network”, а в [42] — “circuit model”.

Иногда рассматривается представление информационного графа в виде матрицы смежности (adjacency matrix) [30] — квадратной матрицы, элемент $(i, j)$ которой равен 1, если вершина $i$ графа зависит от вершины $j$, и 0 в противном случае.

Возможны различные варианты задания информационного графа алгоритма. Одним из подходящих способов может быть использование языка описания информационной структуры алгоритмов Algolang (раздел 3.6.3). Преимуществом при этом может оказаться возможность трехмерной интерактивной визуализации информационного графа, предоставляемая системой AlgoView. На рис. 6 показана визуализация информационного графа, описанного на рис. 5.

**4. Коллекции описаний алгоритмов.** С развитием сети Интернет стало появляться большое количество общедоступных коллекций описаний алгоритмов. Однако набор описываемых свойств алгоритмов в этих коллекциях чаще всего крайне ограничен, а единая структура описания алгоритмов почти всегда отсутствует. Далее рассматриваются наиболее известные коллекции описаний алгоритмов в сети Интернет, для каждой из них приводится краткая характеристика, перечисляются классы описываемых алгоритмов и указывается, по какой схеме проводится их описание и какие основные свойства описываются. В обзор не включены просто библиотеки подпрограмм, реализующие некоторые классы алгоритмов.

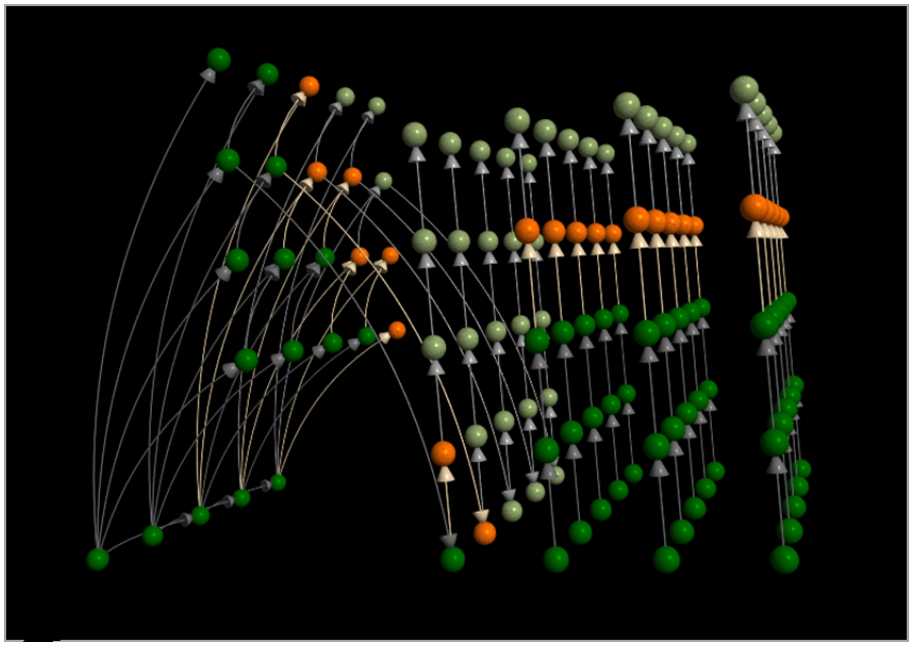

Рис. 6. Визуализация информационного графа (из статьи [47])

Fig. 6. Visualization of the information graph (from the paper [47])

2025, **26** (4), 548–568. doi 10.26089/NumMet.v26r436

**4.1. Список алгоритмов в англоязычной Википедии.** В англоязычной части Википедии страница “List of algorithms” [49] поясняет понятие алгоритма и содержит ссылки на описания достаточно большого количества “хорошо известных” (“well-known”) алгоритмов из самых разных предметных областей.

*§ 4.1.1. Классы описываемых алгоритмов.* Описываемые алгоритмы разбиты на следующие крупные классы: автоматизированное планирование, комбинаторные алгоритмы, вычислительная математика, вычислительная наука, вычислительная техника, теория информации и обработка сигналов, программная инженерия, алгоритмы баз данных, алгоритмы распределенных систем, сетевые технологии, алгоритмы операционных систем. В свою очередь, каждая из этих крупных областей разбивается на более мелкие подобласти, а далее до описаний конкретных алгоритмов.

*§ 4.1.2. Описание структуры алгоритмов.* По каждому алгоритму приводится свое описание, без единой схемы для разных алгоритмов. В результате приводится большое количество информации, но для разных алгоритмов описаны разные свойства. Для описания структуры разных алгоритмов используется словесное описание, псевдокод, участки кода на различных языках программирования. Для части алгоритмов визуализируется динамика их исполнения на некоторых наборах данных.

**4.2. Список алгоритмов в русскоязычной Википедии.** В русскоязычной Википедии приводится описание структуры заметно меньшего количества алгоритмов[2], чем в англоязычной части.

*§ 4.2.1. Классы описываемых алгоритмов.* Из крупных классов алгоритмов выделены: алгоритмы поиска на графах, алгоритмы теории чисел, теоретико-числовые алгоритмы, квантовые алгоритмы, симметричные криптосистемы, криптосистемы с открытым ключом, хеш-функции. Далее приводится алфавитный список из еще 33 более мелких подкатегорий и 120 алгоритмов, входящих в эти категории.

*§ 4.2.2. Описание структуры алгоритмов.* Точно так же, как в англоязычной части Википедии (раздел 4.1), по каждому алгоритму приводится свое описание, без единой схемы. В результате приводится большое количество информации, но для разных алгоритмов описаны разные свойства. Для описания структуры разных алгоритмов используется словесное описание, псевдокод, участки кода на различных языках программирования. Для части алгоритмов визуализируется динамика их исполнения на некоторых наборах данных. Для некоторых алгоритмов приводятся оценки свойств алгоритмов, таких как вычислительная сложность и затраты по памяти, например, на рис. 7 показана врезка со свойствами алгоритма сортировки пузырьком со страницы русскоязычной Википедии[3].


Рис. 7. Свойства алгоритма сортировки пузырьком

Fig. 7. Properties of the bubble sort algorithm

**4.3. GeeksforGeeks.** На странице Data Structures and Algorithms проекта GeeksforGeeks [50] приводятся описания большого количества алгоритмов, для каждого из которых приводятся программные реализации, написанные на языках C++, C, Java, Python, C#, JavaScript. Авторы позиционируют свой проект как обучающий, поэтому все разделы классификации алгоритмов сопровождаются тестом для проверки освоения информации при самостоятельном знакомстве.

*§ 4.3.1. Классы описываемых алгоритмов.* Классы описываемых алгоритмов: работа с массивами, алгоритмы поиска, алгоритмы сортировки, хеширование, метод двух указателей, метод скользящего окна, метод префиксной суммы, строки, рекурсия, матрицы/сетки, связанные списки, стеки, очереди, деки, деревья, кучи, графы, жадные алгоритмы, динамическое программирование, расширенные структуры данных и алгоритмы.

[2]URL страницы https://ru.wikipedia.org/wiki/Категория:Алгоритмы

[3]URL страницы https://ru.wikipedia.org/wiki/Сортировка_пузырьком

2025, **26** (4), 548–568. doi 10.26089/NumMet.v26r436

*§ 4.3.2. Описание структуры алгоритмов.* Для каждого алгоритма приводятся: его словесное описание, реализации на различных языках программирования; для части алгоритмов приводятся оценки их вычислительной сложности и затрат по памяти.

**4.4. Modernes.** В проекте Modernes [51] приводятся реализации параллельных версий некоторых известных простых алгоритмов, написанные на языке C++. Вызов параллельных версий производится либо с помощью перегрузки вызова стандартной функции, либо в отдельных случаях явным вызовом написанной новой функции. Используя политику выполнения, можно указать, должен ли алгоритм выполняться последовательно, параллельно или параллельно и векторно.

*§ 4.4.1. Классы описываемых алгоритмов.* На данный момент приведены параллельные реализации около 100 алгоритмов, включая различные варианты поиска, сортировки и т.д.

*§ 4.4.2. Описание структуры алгоритмов.* Никакие свойства алгоритмов в рамках проекта отдельно не описываются. Описания самих алгоритмов ссылаются на Википедию (4.1).

**4.5. Алгоритмика.** На сайте проекта Алгоритмика [52] находятся материалы различных курсов по программированию, проводящихся в Tinkoff Generation. Проект открытый, все исходные коды на языке программирования C++ доступны на GitHub [53].

*§ 4.5.1. Классы описываемых алгоритмов.* Описываются алгоритмы, реализующие сортировки, бинарный поиск, задачи комбинаторики, линейной алгебры, задачи на графах, задачи геометрии и многое другое.

*§ 4.5.2. Описание структуры алгоритмов.* Для рассматриваемых алгоритмов приводятся словесное описание и их реализации в виде фрагментов программ на языке C++. Из описываемых свойств можно выделить асимптотическую вычислительную сложность. Также для отдельных алгоритмов рассматриваются возможные оптимизации и описываются их применения, для итерационных алгоритмов дается оценка скорости сходимости.

**4.6. MAXimal :: algo.** На странице MAXimal :: algo [54], являющейся частным проектом, посвященным олимпиадному программированию, представлены описания 145 алгоритмов.

*§ 4.6.1. Классы описываемых алгоритмов.* В проекте приводятся описания алгоритмов из следующих областей: алгебра, графы, геометрия, алгоритмы на последовательностях, линейная алгебра, численные методы, комбинаторика, теория игр и др.

*§ 4.6.2. Описание структуры алгоритмов.* Ко всем алгоритмам даны словесные описания и приводятся программы на языке программирования C++. Для многих алгоритмов приводятся оценки асимптотики вычислительной сложности.

**4.7. Algorithms for Competitive Programming.** Проект Algorithms for Competitive Programming [55] описывает себя как расширение и перевод на английский язык проекта 4.6.

*§ 4.7.1. Классы описываемых алгоритмов.* Классы представленных в проекте алгоритмов: алгебра, структуры данных, динамическое программирование, обработка строк, линейная алгебра, комбинаторика, численные методы, геометрия, графы, разное.

*§ 4.7.2. Описание структуры алгоритмов.* Для каждого алгоритма приводятся: его словесное описание, код реализации на языке C++, оценки вычислительной сложности, описания возможных модификаций и оптимизаций.

**4.8. The Stony Brook Algorithm Repository.** Описания алгоритмов на странице The Stony Brook Algorithm Repository [56] рассматриваются в качестве сопровождающего материала для книги [12]. Всего описываются алгоритмы решения 75 фундаментальных задач в комбинаторных алгоритмах.

*§ 4.8.1. Классы описываемых алгоритмов.* На данном сайте описываются алгоритмы из следующих областей: работа со структурами данных, численные задачи, комбинаторные задачи, граф: задачи с полиномиальным временем, граф: сложные задачи, вычислительная геометрия, задачи о множествах и строках.

2025, **26** (4), 548–568. doi 10.26089/NumMet.v26r436

*§ 4.8.2. Описание структуры алгоритмов.* Приводятся словесные описания алгоритмов, а также ссылки на различные библиотеки, содержащие их реализации. Никакие свойства алгоритмов отдельно не рассматриваются.

**4.9. Алгоритмы и структуры данных на странице вики-конспектов ИТМО.** На сайте проекта[4] представлены конспекты, которые написаны самостоятельно студентами кафедры компьютерных технологий Университета ИТМО.

*§ 4.9.1. Классы описываемых алгоритмов.* На странице описаны алгоритмы, разбитые на следующие группы: амортизационный анализ, персистентные структуры данных, приоритетные очереди, система непересекающихся множеств, поисковые структуры данных, запросы на отрезках, дерево Фенвика, задача о наименьшем общем предке, хеширование, сортировки, сортирующие сети, алгоритмы поиска, динамическое программирование, криптографические алгоритмы, связь между структурами данных, алгоритмы во внешней памяти.

*§ 4.9.2. Описание структуры алгоритмов.* Для каждого алгоритма приводится его словесное описание, во многих случаях сопровождаемое фрагментами некоторых его реализаций на языке C++, в некоторых случаях приводятся оценки асимптотических свойств алгоритма и сравнения с другими алгоритмами, решающими ту же самую задачу. Единой схемы описания структуры алгоритмов нет.

**4.10. Algocode wiki.** Сайт проекта Algocode [57] представляет собой wiki-страницу кружка Яндекс по изучению алгоритмов.

*§ 4.10.1. Классы описываемых алгоритмов.* На странице представлены алгоритмы, разбитые на следующие классы: поиски, сортировки, динамическое программирование, графы, геометрия, математика, структуры данных, строковые алгоритмы, оптимизации, нестандартные алгоритмы.

*§ 4.10.2. Описание структуры алгоритмов.* Для каждого алгоритма приводится его словесное описание, для многих приводятся реализации на языке C++, а также некоторые свойства, например вычислительная сложность. Единой схемы описания структуры алгоритмов нет.

**4.11. Алгоритмы — IT wiki ru.** Раздел по алгоритмам wiki по ИТ на русском языке [58] является частью проекта, созданного и поддерживаемого сообществом энтузиастов и профессионалов. Авторы ссылаются на описания алгоритмов из книги [11].

*§ 4.11.1. Классы описываемых алгоритмов.* В разделе по алгоритмам на данный момент есть только описания нескольких видов алгоритмов сортировок.

*§ 4.11.2. Описание структуры алгоритмов.* В данном проекте приводятся словесные описания алгоритмов и некоторые их реализации на языках Python, C++ и Java. В описаниях приводятся результаты анализа временно́й и пространственной сложности. Структура описания состоит из четырех разделов: алгоритм, анализ сложности, реализация, заключение.

**4.12. Algorithm Wiki.** Проект Algorithm Wiki [59] описывает себя “Краудсорсинговый ресурс об алгоритмах и их разработке”. Других описаний проекта и политики его использования не приводится.

*§ 4.12.1. Классы описываемых алгоритмов.* В проекте описываются алгоритмы из следующих областей: биоинформатика, комбинаторика, базы данных, обработка изображений, численный анализ, операционные системы, робототехника, обработка сигналов, статистика, криптография.

*§ 4.12.2. Описание структуры алгоритмов.* Для описываемых алгоритмов приводятся: словесное описание, описание их параметров, ссылки на различные реализации; также приводятся оценки вычислительной сложности, объема используемых данных и некоторые другие характеристики. Структура описания отличается от алгоритма к алгоритму.

**4.13. Algorithm Wiki | Interactive algorithms.** Данный проект [60] является экспериментом по созданию описаний интерактивных алгоритмов в Интернете. Алгоритмы на этой странице не просто иллюстрируются анимацией, они построены на реальном интерпретаторе, а их визуализации основаны на реальном выполняемом коде на языке Javascript.

*§ 4.13.1. Классы описываемых алгоритмов.* Среди основных классов описываемых в данном проекте алгоритмов выделены: работа со структурами данных, поиск, сортировка, сравнительная сортировка, другие виды сортировки, деревья, графы, математика, другие.

[4]URL страницы https://neerc.ifmo.ru/wiki/index.php?title=Алгоритмы_и_структуры_данных

2025, **26** (4), 548–568. doi 10.26089/NumMet.v26r436

*§ 4.13.2. Описание структуры алгоритмов.* Кроме краткого словесного описания алгоритмов, приводится их программная реализация на языке C++, для некоторых алгоритмов приводятся оценки временно́й и пространственной сложности. Главной особенностью проекта является пошаговая визуализация процесса выполнения алгоритма. Рис. 8 демонстрирует фрагмент одного из вариантов такой визуализации для алгоритма Дейкстры поиска кратчайших путей между вершинами графа [61]. Четкой структуры описания различных алгоритмов не предусмотрено.

**4.14. Algorithms — Wikibooks, open books for an open world.** Проект [62] в значительной степени базируется на курсе лекций, прочитанных в University of California San Diego.

*§ 4.14.1. Классы описываемых алгоритмов.* Основные классы представленных алгоритмов: разделяй и властвуй, рандомизация, поиск с возвратом, динамическое программирование, жадные алгоритмы, поиск восхождением к вершине, алгоритмы невзвешенных графов, аппроксимация расстояний.

*§ 4.14.2. Описание структуры алгоритмов.* Приводятся: словесное описание алгоритмов, их реализация на псевдокоде; также в отдельных алгоритмах описываются различные свойства, включая вычислительную сложность, оценки объема данных, различные особенности алгоритмов и их реализаций.

**4.15. Algorithms | Brilliant Math & Science Wiki.** Описание структуры алгоритмов от Brilliant.org [63].

*§ 4.15.1. Классы описываемых алгоритмов.* Классы представленных в проекте алгоритмов: алгоритмы сортировки, графовые алгоритмы, алгоритмы кратчайшего пути, алгоритмы сопоставления строк, алгоритмы максимального потока, алгоритмы вычислительной геометрии, теоретико-числовые алгоритмы, алгоритмы быстрого преобразования Фурье, матричные алгоритмы, алгоритмы “разделяй и властвуй”, жадные алгоритмы, алгоритмы динамического программирования, рекурсивные алгоритмы, алгоритмы грубой силы, алгоритмы обратного хода, вероятностные и рандомизированные алгоритмы, алгоритмы аппроксимации, многопоточные алгоритмы, алгоритмы линейного программирования.

*§ 4.15.2. Описание структуры алгоритмов.* Структура описания алгоритмов включает в себя словесное описание, формальное определение, псевдокод, реализацию. Основные описываемые для алгоритмов свойства: вычислительная сложность, пространственная сложность, корректность. Для многих алгоритмов приводится анимированная демонстрация их работы. Единой структуры описания алгоритмов нет, хотя определенные шаги в этом направлении сделаны.

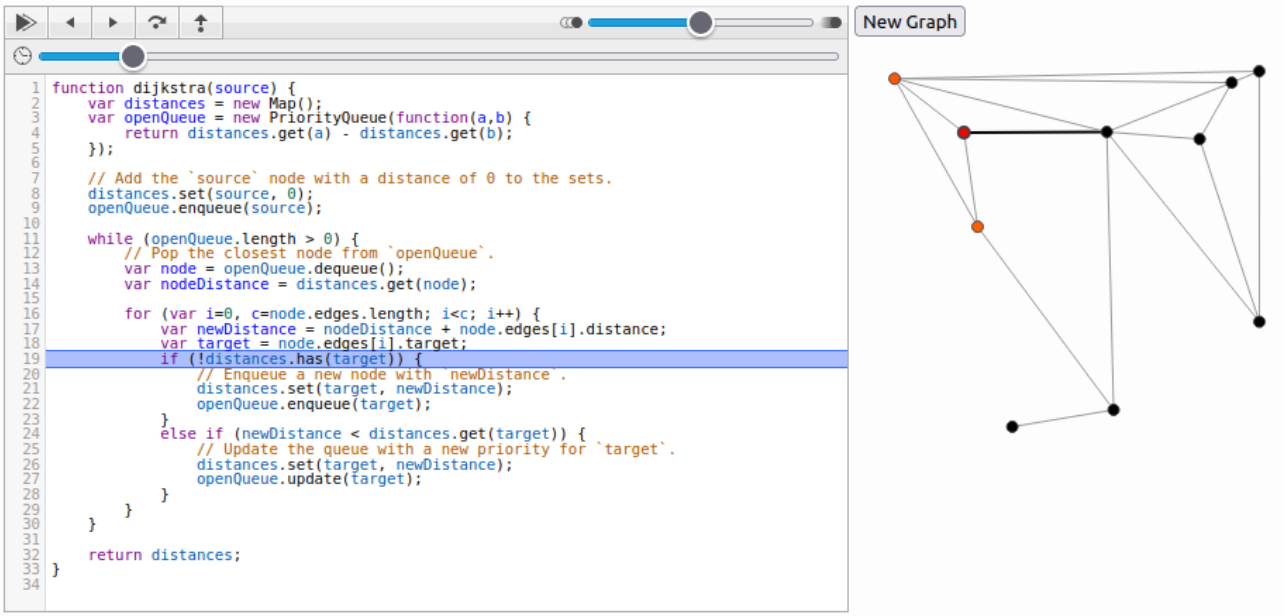


Рис. 8. Визуализация алгоритма Дейкстры поиска кратчайших путей между вершинами графа (со страницы [61])

Fig. 8. Visualization of Dijkstra’s algorithm Dijkstra’s algorithm for finding the shortest paths between nodes in a graph (from the page [61])

2025, **26** (4), 548–568. doi 10.26089/NumMet.v26r436

**4.16. Scriptol.** На странице [64] представлена классификация примерно 300 различных алгоритмов. Однако для всех алгоритмов приведена только краткая характеристика, неполное описание приведено только для малой части из них.

*§ 4.16.1. Классы описываемых алгоритмов.* Классы указанных в классификации алгоритмов: автоматы, искусственный интеллект, биоинформатика и хемоинформатика, сжатие данных, криптография, геометрия, графы, графика, списки, массивы и деревья, логическое программирование, математика, обработка матриц, оптика, оптимизация, синтаксический анализ, прогнозирование (статистика), квантовая математика, генераторы случайных чисел, науки, обработка сигналов, программная инженерия, тексты, утилиты, разное.

*§ 4.16.2. Описание структуры алгоритмов.* Никакой единой структуры описания алгоритмов нет. Для части алгоритмов приводятся коды их реализаций на языках Java, JavaScript, PHP, C, C++, для некоторых алгоритмов указаны ссылки на их описания в виде опубликованных статей или фрагментов лекций.

**4.17. Открытая энциклопедия свойств алгоритмов AlgoWiki.** Проект Открытой энциклопедии свойств алгоритмов AlgoWiki [14, 65, 66] нацелен на создание методов эффективного использования компьютерных систем с параллельной архитектурой. Для этого в энциклопедии описываются свойства алгоритмов с особым акцентом на свойства, связанные с параллелизмом [67].

*§ 4.17.1. Классы описываемых алгоритмов.* Основой для описания любой предметной области в рамках Открытой энциклопедии свойств алгоритмов AlgoWiki является иерархическое представление в виде цепочек описаний “задача–метод–алгоритм–реализация” [68, 69].

Уровень Задачи может соответствовать описанию как конкретных решаемых практических проблем, так и задач в чисто математической постановке. На самом верхнем уровне группы описываемых задач такие: задачи алгебры, алгоритмы на списках и массивах, вычислительная геометрия, исследование и моделирование компьютеров, прикладные задачи из разных областей, криптографические алгоритмы. Далее эти группы разбиваются на более мелкие конкретные решаемые задачи.

Уровень Метода описывает разные подходы к решению рассматриваемых задач. Каждый метод обладает своими характеристиками, и в определенных условиях может быть выгодно использовать один из них.

Уровень Алгоритма является базовым уровнем энциклопедии AlgoWiki. Каждый метод решения задачи может предусматривать возможность использования того или иного алгоритма. Подробно описываемые свойства алгоритмов позволяют выбрать наиболее подходящий из алгоритмов для имеющейся высокопроизводительной вычислительной системы.

Наконец, уровень Реализации [70] предусматривает возможность использования различных программных реализаций для одного и того же алгоритма. Реализации могут отличаться по особенностям использования свойств самих реализуемых алгоритмов, по вариантам используемых программных технологий, а также и по особенностям используемых вычислительных платформ.

На рис. 9 показан фрагмент описания классификации алгоритмов в проекте AlgoWiki[5], на котором хорошо видны цепочки описаний “задача–метод–алгоритм–реализация”. Значком с буквой “З” обозначены описания задач, “М” — описания методов, “А” — описания алгоритмов и “Р” — описания программных реализаций.

*§ 4.17.2. Описание структуры алгоритмов.* В проекте AlgoWiki была разработана единая универсальная структура описания любых вычислительных алгоритмов[6]. Изначально эта структура объединяла описания самих алгоритмов и их программных реализаций, но позже для программных реализаций были выделены отдельные сущности[7].

В результате для описания структуры алгоритмов в настоящий момент используется схема, представленная на рис. 10, а для описания структуры программных реализаций — схема, представленная на рис. 11.

Разработанные в проекте Открытой энциклопедии свойств алгоритмов AlgoWiki структуры описания свойств алгоритмов и их программных реализаций позволяют по единой схеме описывать структуру

[5]URL страницы https://algowiki-project.org/ru/Классификация_алгоритмов

[6]URL страницы https://algowiki-project.org/ru/Структура_описания_свойств_алгоритмов

[7]URL страницы https://algowiki-project.org/ru/Структура_описания_свойств_программных_реализаций

2025, **26** (4), 548–568. doi 10.26089/NumMet.v26r436


Рис. 9. Фрагмент страницы “Классификация алгоритмов”

Fig. 9. Fragment from the page “Algorithm classification”

любых вычислительных алгоритмов. Согласно данной структуре в проекте описаны несколько десятков различных вычислительных алгоритмов.

**5. Заключение.** Приведен обзор существующих методов описания структуры алгоритмов, от описательных до формальных. Показаны преимущества, недостатки и области применимости разных методов.

Рассмотрены доступные в сети Интернет коллекции описаний алгоритмов. Для каждой из них указаны классы описываемых алгоритмов и перечислены применяемые методы описания их структуры и основные описываемые свойства.

Обзор показал, что почти во всех существующих коллекциях алгоритмов используется словесное описание и приводятся реализации на различных языках программирования. Другие методы описания структуры алгоритмов применяются крайне редко. Наиболее часто описываемым свойством алгоритмов является оценка их вычислительной сложности. В большинстве коллекций не предусмотрена единая схема описания структуры алгоритмов, чаще всего каждый алгоритм описывается по своей схеме с некоторыми общими элементами.

2025, **26** (4), 548–568. doi 10.26089/NumMet.v26r436

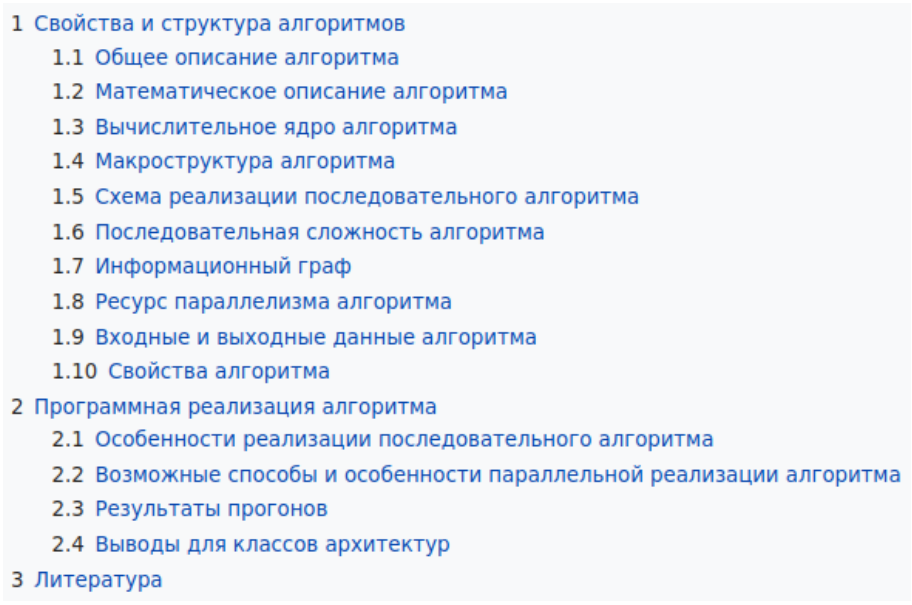


Рис. 10. Структура описания свойств алгоритмов в проекте AlgoWiki

Fig. 10. Description of algorithm properties and structure in the AlgoWiki project

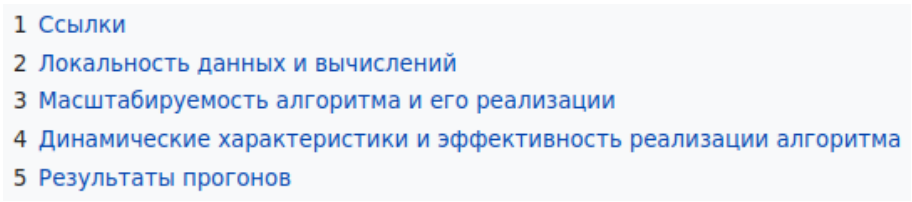


Рис. 11. Структура описания свойств программных реализаций в проекте AlgoWiki

Fig. 11. Description of implementation properties and structure in the AlgoWiki project

Исключением является проект Открытой энциклопедии свойств алгоритмов AlgoWiki (раздел 4.17), в котором предложена единая универсальная схема описания структуры любых вычислительных алгоритмов. В эту схему включены многие важные свойства алгоритмов, в том числе те, которые характеризуют их параллельные свойства (информационный граф, ресурс параллелизма и др.). Также энциклопедия AlgoWiki предлагает наиболее строго проработанную схему описания предметной области в виде цепочек описаний “задача–метод–алгоритм–реализация”, что позволяет распределить алгоритмы в рамках возникающей уникальной классификации.

Таким образом, из проведенного анализа следует, что именно схема, предложенная в Открытой энциклопедии свойств алгоритмов AlgoWiki, наиболее полно и многогранно описывает и раскрывает свойства вычислительных алгоритмов. В других коллекциях алгоритмов можно отметить отдельные интересные элементы, но стройной структуры описания алгоритмов они не предлагают.

## Список литературы

1. *Boyer C.B.* The arabic hegemony // A History of Mathematics (Second ed.). New York: Wiley, 1991.
2. *Кнут Д.* Искусство программирования. Том 1. Основные алгоритмы. 3-е издание. М: Вильямс, 2001.
3. *Марков А.А.* Теория алгорифмов. Тр. МИАН СССР. 1954. **42**. 3—375. https://www.mathnet.ru/rus/tm1178. (Дата обращения: 30 ноября 2025).
4. *Колмогоров А.Н., Успенский В.А.* К определению алгоритма. УМН. 1958. **13**, № 4. 3—28. https://www.mathnet.ru/rm7453. (Дата обращения: 30 ноября 2025).
5. Философский словарь. Изд. 3 / Ред. Розенталь М.М. М.: Политиздат, 1975.
6. *Standards Coordinating Committee 10, Terms and Definitions* The IEEE Standard Dictionary of Electrical and Electronics Terms. J. Radatz, Ed. New York: IEEE, 1996.
7. Dictionary of algorithms and data structures. https://xlinux.nist.gov/dads/. (Дата обращения: 30 ноября 2025).

2025, **26** (4), 548–568. doi 10.26089/NumMet.v26r436


8. *Селиванова И.А., Блинов В.А.* Фундаментальные алгоритмы на C++. Построение и анализ алгоритмов обработки данных: учебно-методическое пособие. Екатеринбург: Издательство Уральского университета. 2015.
9. *Угринович Н.Д.* Информатика и информационные технологии. Учебник для 10–11 классов. 3-е изд. М.: БИНОМ. Лаборатория знаний, 2006.
10. *Woodhouse D., Johnstone G., et al.* Computer science. Milton, Qld., Jacaranda Wiley. 1984.
11. *Кормен Т.Х., Лейзерсон Ч.И., Ривест Р.Л., Штайн К.* Алгоритмы: построение и анализ, 2-е изд.: Пер. с англ. М.: Издательский дом Вильямс, 2011.
12. *Скиена С.* Алгоритмы. Руководство по разработке. 2-е изд.: Пер. с англ. Спб.: БХВ-Петербург, 2011.
13. *Седжвик Р.* Фундаментальные алгоритмы на C++. Анализ/Структуры данных/Сортировка/Поиск. Киев: Изд-во ДиаСофт, 2001.
14. Открытая энциклопедия свойств алгоритмов. https://algowiki-project.org. (Дата обращения: 30 ноября 2025).
15. *Press W.H.* Numerical recipes 3rd edition: the art of scientific computing. Cambridge: Cambridge University Press, 2007. https://numerical.recipes/. (Дата обращения: 30 ноября 2025).
16. *von Neumann J.* The collected works of John von Neumann, Volume V. New York: Macmillan, 1963.
17. ГОСТ 19.701-90. https://protect.gost.ru/document.aspx?control=7&id=137637. (Дата обращения: 30 ноября 2025).
18. ISO 5807:1985. https://www.iso.org/ru/standard/11955.html. (Дата обращения: 30 ноября 2025).
19. Теория и практика UML. Диаграмма деятельности. https://it-gost.ru/articles/view_articles/96. (Дата обращения: 13 октября 2025).
20. *Nassi I., Shneiderman B.* Flowchart techniques for structured programming // ACM SIGPLAN Notices **8** (8). 12–26 (1973). doi 10.1145/953349.953350.
21. *Post E.L.* Finite combinatory processes — formulation I // J. Symbolic Logic **1** (3). 103—105 (1936).
22. *Turing A.M.* On computable numbers with an application to the Entscheidungsproblem. // Proc. London Math. Society. 1937. ser. 2, vol. 42. 230–265. ibid. ser. 2, vol. 43. 544–546.
23. *Church A.* An unsolvable problem of elementary number theory // American J. Math. **58**, N 2. 345–363 (1936).
24. *Asanović K., Bodik R., Catanzaro B., et al.* The landscape of parallel computing research: a view from Berkeley. EECS Technical Report UCB/EECS-2006-183. 2006. https://people.eecs.berkeley.edu/~krste/papers/BerkeleyView.pdf. (Дата обращения: 30 ноября 2025).
25. *Asanović K., Bodik R., Demmel J., et al.* The parallel computing laboratory at U.C. Berkeley: a research agenda based on the Berkeley view. Technical Report No. UCB/EECS-2008-23. 2008. https://www2.eecs.berkeley.edu/Pubs/TechRpts/2008/EECS-2008-23.pdf. (Дата обращения: 30 ноября 2025).
26. *Cole M.I.* Algorithmic skeletons: structured management of parallel computation. MIT Press, 1991.
27. *Rabhi F.A., Gorlatch S.* Patterns and skeletons for parallel and distributed computing. London: Springer, 2003. doi 10.1007/978-1-4471-0097-3.
28. *Вирт Н.* Алгоритмы и структуры данных. Новая версия для Оберона + CD. Пер. с англ. Ткачев Ф.В. М.: ДМК Пресс, 2010.
29. *Reingold E.M.* Basic techniques for design and analysis of algorithms. Computer science handbook / ed. A.B. Tucker. Chapman and Hall/CRC, 2004.
30. *Gebali F.* Algorithms and parallel computing. Wiley, 2011. doi 10.1002/9780470932025.
31. *Воеводин В.В., Воеводин Вл.В.* Параллельные вычисления. СПб.: БХВ–Петербург, 2002.
32. OpenMP: Home. https://www.openmp.org/. (Дата обращения: 30 ноября 2025).
33. MPI Forum. https://www.mpi-forum.org/. (Дата обращения: 30 ноября 2025).
34. CUDA Zone. https://developer.nvidia.com/cuda-zone. (Дата обращения: 30 ноября 2025).
35. *Blelloch G.E.* Programming parallel algorithms. // Communications of the ACM. 1996. **39**, N 3. 85–97.
36. *Smith J.R.* The design and analysis of parallel algorithms. Oxford: Oxford University Press, 1993.
37. *Макконнелл Дж.* Основы современных алгоритмов. 2-е дополненное издание. М.: Техносфера, 2004.
38. *JáJá J.* An introduction to parallel algorithms. Addison Wesley Longman Publishing Co., Inc., USA, 1992. doi 10.1016/S0898-1221(99)90305-X.
39. *Mehlhorn K., Sanders P.* Algorithms and data structures: the basic toolbox. Springer Science & Business Media, 2008. doi 10.1007/978-3-540-77978-0.
40. *Akl S.G.* The design and analysis of parallel algorithms. London: Prentice Hall, 1989.
41. *Alsuwaiyel M.H.* Parallel algorithms. World Scientific Publishers, 2022. doi 10.1142/12744.
42. *Blelloch G.E., Maggs B.M.* Parallel algorithms. Computer science handbook / ed. A.B. Tucker/ 2nd ed. pp. 232–272. Pittsburgh: Carnegie Mellon University, 2004.
43. *Casanova H., Legrand A., Robert Y.* Parallel algorithms. New York: CRC PRESS, 2008.
44. *McConnell S.* Code complete: a practical handbook of software construction, 2nd ed. Microsoft Press, USA, 2004.
45. *Пароннджанов В.Д.* Алгоритмические языки и программирование: ДРАКОН. М.: Изд-во Юрайт, 2022.

2025, **26** (4), 548–568. doi 10.26089/NumMet.v26r436


46. *Antonov A.S., Volkov N.I.* Information graph visualization using AlgoView Software Tool // Lobachevskii J. Math. 2020. **41**, N 8. 1427–1434. doi 10.1134/S199508022008003X.
47. *Antonov A.S., Volkov N.I.* Study of the algorithms information structure as the basis of a training workshop // Communications in Computer and Information Science. 2021. **1510**. 404–414. doi 10.1007/978-3-030-92864-3_31.
48. *Skryabin G., Gadieva T., Antonov A.* A New Version of the AlgoView System for 3D Visualization and interactive analysis of information graphs of algorithms // Communications in Computer and Information Science. 2024. **2241**. 19–33. doi 10.1007/978-3-031-73372-7_2.
49. List of algorithms — Wikipedia. https://en.wikipedia.org/wiki/List_of_algorithms. (Дата обращения: 30 ноября 2025).
50. DSA tutorial — learn data structures and algorithms — GeeksforGeeks. https://www.geeksforgeeks.org/dsa/dsa-tutorial-learn-data-structures-and-algorithms/. (Дата обращения: 30 ноября 2025).
51. Parallel algorithms of the standard template library. https://www.modernescpp.com/index.php/parallel-algorithm-of-the-standard-template-library/. (Дата обращения: 30 ноября 2025).
52. Algorithmica. https://algorithmica.org/. (Дата обращения: 30 ноября 2025).
53. GitHub — algorithmica-org/algorithmica: a computer science textbook. https://github.com/algorithmica-org/algorithmica. (Дата обращения: 30 ноября 2025).
54. MAXimal :: algo. http://e-maxx.ru/algo/. (Дата обращения: 30 ноября 2025).
55. Main page — algorithms for competitive programming. https://cp-algorithms.com/. (Дата обращения: 30 ноября 2025).
56. Algorithm repository. https://www.algorist.com/algorist.html. (Дата обращения: 30 ноября 2025).
57. Algocode wiki. https://wiki.algocode.ru. (Дата обращения: 30 ноября 2025).
58. Алгоритмы — IT wiki ru. https://www.it-wiki.com.ru/algorithms/. (Дата обращения: 13 октября 2025).
59. Algorithm Wiki. https://algorithm-wiki.csail.mit.edu/. (Дата обращения: 13 октября 2025).
60. Algorithm Wiki | Interactive algorithms. https://thimbleby.gitlab.io/algorithm-wiki-site. (Дата обращения: 30 ноября 2025).
61. Dijkstra's algorithm | Algorithm Wiki. https://thimbleby.gitlab.io/algorithm-wiki-site/wiki/Dijkstras_algorithm/. (Дата обращения: 30 ноября 2025).
62. Algorithms — Wikibooks, open books for an open world. https://en.wikibooks.org/wiki/Algorithms. (Дата обращения: 30 ноября 2025).
63. Algorithms | Brilliant Math & Science Wiki. https://brilliant.org/wiki/algorithm/. (Дата обращения: 30 ноября 2025).
64. List of algorithms. https://www.scriptol.com/programming/list-algorithms.php. (Дата обращения: 30 ноября 2025).
65. *Воеводин Вл.В.* Открытая энциклопедия свойств алгоритмов AlgoWiki: от мобильных платформ до экзафлопсных суперкомпьютерных систем // Вычислительные методы и программирование. 2015. **16**, № 1. 99–111. doi 10.26089/NumMet.v16r111.
66. *Voevodin Vl.V., Antonov A.S., Dongarra J.* AlgoWiki: an open encyclopedia of parallel algorithmic features // Supercomputing Frontiers and Innovations. 2015. **2**, N 1. 4–18. doi 10.14529/jsfi150101.
67. *Voevodin Vl., Antonov A., Dongarra J.* Why is it hard to describe properties of algorithms? // Procedia Computer Science. 2016. **101**. 4–7. doi 10.1016/j.procs.2016.11.002.
68. *Antonov A., Frolov A., Konshin I., Voevodin Vl.* Hierarchical domain representation in the AlgoWiki encyclopedia: from problems to implementations // Communications in Computer and Information Science. 2018. **910**. 3–15. Cham: Springer. doi 10.1007/978-3-319-99673-8_1.
69. *Popov A., Nikitenko D., Antonov A., Voevodin Vl.* Formal model of problems, methods, algorithms and implementations in the advancing AlgoWiki open encyclopedia // CEUR Workshop Proc. 2018. **2281**. 1–11. http://ceur-ws.org/Vol-2281/paper-01.pdf. (Дата обращения: 30 ноября 2025).
70. *Антонов А.С.* Выделение явного уровня реализации алгоритмов для использования в проекте Algo500 // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2023. **12**, № 1. 89–100. doi 10.14529/cmse230105.


*Получена* 27 октября 2025 г. | *Принята* 27 ноября 2025 г. | *Опубликована* 5 декабря 2025 г.

### Информация об авторе

*Александр Сергеевич Антонов* — к.ф.-м.н., вед. науч. сотр.; Московский государственный университет имени М. В. Ломоносова, Научно-исследовательский вычислительный центр, Ленинские горы, 1, стр. 4, 119234, Москва, Российская Федерация.

2025, **26** (4), 548–568. doi 10.26089/NumMet.v26r436

## References


1. C. B. Boyer, The Arabic Hegemony. *A History of Mathematics (Second ed.)* (Wiley, New York, 1991).
2. D. Knuth, *The Art of Computer Programming, vol.1. Fundamental Algorithms. 3d ed.* (Addison-Wesley, Reading, Massachusetts, 1997).
3. A. A. Markov, Theory of algorithms. *Proc. of the Steklov Institute of Mathematics of the USSR* Vol. 42. 3—375 (1954) [in Russian].
4. A. N. Kolmogorov, V. A. Uspenskiy, "On the definition of an algorithm," Russian Math. Surveys. **13** (4). 3—28 (1958). https://www.mathnet.ru/rm7453. Cited November 30, 2025.
5. Philosophical Dictionary. 3rd ed. / Ed. M. M. Rosental (1975).
6. Standards Coordinating Committee 10, Terms and Definitions. *The IEEE Standard Dictionary of Electrical and Electronics Terms. J. Radatz, Ed.* (IEEE, New York, 1996).
7. Dictionary of Algorithms and Data Structures. https://xlinux.nist.gov/dads/. Cited November 30, 2025.
8. I. A. Selivanova and V. A. Blinov, *Fundamental algorithms in C++. Construction and analysis of data processing algorithms: a teaching aid* (Ural University Publishing House, Ekaterinburg, 2015).
9. N. D. Ugrinovich, *Computer science and information technology. Textbook for grades 10–11. 3rd ed.* (BINOM. Knowledge laboratory, Moscow, 2006) [in Russian].
10. D. Woodhouse, G. Johnstone, et al., *Computer science* ( Milton, Qld., Jacaranda Wiley, 1984).
11. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms, 2nd ed.* (The MIT Press, Cambridge, Massachusetts, 2001).
12. S. S. Skiena, *The Algorithm Design Manual. 2nd ed.* (Springer, New York, 2008).
13. R. Sedgewick, *Algorithms in C++. Fundamentals/Data Structures/Sorting/Searching. 3rd ed.* (Addison Wesley Longman, 1999).
14. Open Encyclopedia of Algorithmic Features. https://algowiki-project.org. Cited November 30, 2025.
15. W. H. Press, *Numerical Recipes 3rd Edition: The Art of Scientific Computing* (Cambridge University Press, Cambridge, 2007). https://numerical.recipes/. Cited November 30, 2025.
16. J. von Neumann, *The Collected Works of John von Neumann, Volume V* (Macmillan, New York, 1963).
17. GOST 19.701-90. https://protect.gost.ru/document.aspx?control=7&id=137637. Cited November 30, 2025.
18. ISO 5807:1985. https://www.iso.org/ru/standard/11955.html. Cited November 30, 2025.
19. UML: Theory and Practice. Activity Diagram. https://it-gost.ru/articles/view_articles/96. Cited October 13, 2025.
20. I. Nassi and B. Shneiderman, "Flowchart techniques for structured programming," ACM SIGPLAN Notices **8** (8), 12–26 (1973). doi 10.1145/953349.953350.
21. E. L. Post, "Finite Combinatory Processes — Formulation I," J. Symbolic Logic **1** (3), 103—105 (1936).
22. A. M. Turing, "On Computable Numbers with an Application to the Entscheidungsproblem," Proc. London Math. Society. ser. 2, vol. 42, 230–265. ibid. ser. 2, vol. 43, 544–546. (1937).
23. A. Church, "An Unsolvable Problem of Elementary Number Theory," American J. Math. **58** (2), 345–363 (1936).
24. K. Asanović, R. Bodik, B. Catanzaro, et al., *The Landscape of Parallel Computing Research: A View from Berkeley* EECS Technical Report UCB/EECS-2006-183 (2006). https://people.eecs.berkeley.edu/~krste/papers/BerkeleyView.pdf. Cited November 30, 2025.
25. K. Asanović, R. Bodik, J. Demmel, et al., *The Parallel Computing Laboratory at U.C. Berkeley: A Research Agenda Based on the Berkeley View* Technical Report No. UCB/EECS-2008-23 (2008). https://www2.eecs.berkeley.edu/Pubs/TechRpts/2008/EECS-2008-23.pdf. Cited November 30, 2025.
26. M. I. Cole, *Algorithmic Skeletons: Structured Management of Parallel Computation* (MIT Press, 1991).
27. F. A. Rabhi and S. Gorlatch (ed.), *Patterns and Skeletons for Parallel and Distributed Computing* (Springer, London, 2003). doi 10.1007/978-1-4471-0097-3.
28. N. Wirth, *Algorithms and Data Structures. Oberon version + CD* (1985. Oberon version: 2004).
29. E. M. Reingold, *Basic Techniques for Design and Analysis of Algorithms. Computer science handbook / ed. A. B. Tucker* (Chapman and Hall/CRC, 2004).
30. F. Gebali, *Algorithms and Parallel Computing* (Wiley, 2011). doi 10.1002/9780470932025.
31. V. V. Voevodin and Vl. V. Voevodin, *The Parallel Computing* (BHV-Petersburg, St. Petersburg, 2002) [in Russian].
32. OpenMP: Home. https://www.openmp.org/. Cited November 30, 2025.
33. MPI Forum. https://www.mpi-forum.org/. Cited November 30, 2025.
34. CUDA Zone. https://developer.nvidia.com/cuda-zone. Cited November 30, 2025.

2025, **26** (4), 548–568. doi 10.26089/NumMet.v26r436


35. G. E. Blelloch, "Programming Parallel Algorithms," Communications of the ACM **39** (3), 85–97 (1996).
36. J. R. Smith, *The Design and Analysis of Parallel Algorithms* (Oxford University Press, Oxford, 1993).
37. McConnell J., *Fundamentals of Modern Algorithms. 2nd ed.* (Moscow: Tekhnosfera, 2004) [in Russian].
38. J. JáJá, *An Introduction to Parallel Algorithms* (Addison Wesley Longman Publishing Co., Inc., USA, 1992). doi 10.1016/S0898-1221(99)90305-X.
39. K. Mehlhorn and P. Sanders, *Algorithms and Data Structures: The Basic Toolbox* (Springer Science & Business Media, 2008). doi 10.1007/978-3-540-77978-0.
40. S. G. Akl, *The Design and Analysis of Parallel Algorithms* (Prentice Hall, London, 1989).
41. M. H. Alsuwaiyel, *Parallel algorithms* (World Scientific Publishers, 2022). doi 10.1142/12744.
42. G. E. Blelloch and B. M. Maggs, Parallel Algorithms. *Computer science handbook / ed. A.B. Tucker/ 2nd ed.* pp. 232–272. (Carnegie Mellon University, Pittsburgh, 2004).
43. H. Casanova, A. Legrand, and Y. Robert, *Parallel Algorithms* (CRC PRESS, New York, 2008).
44. S. McConnell, *Code Complete: A Practical Handbook of Software Construction, 2nd ed.* (Microsoft Press, USA, 2004).
45. V. D. Parondzhanov, *Algorithmic Languages and Programming: DRAKON* (Publishing house Yurait, Moscow, 2022) [in Russian].
46. A. S. Antonov and N. I. Volkov, "Information Graph Visualization Using AlgoView Software Tool," Lobachevskii J. Math. **41** (8), 1427–1434 (2020). doi 10.1134/S199508022008003X.
47. A. S. Antonov and N. I. Volkov, "Study of the Algorithms Information Structure as the Basis of a Training Workshop," Communications in Computer and Information Science. **1510**, 404–414 (2021). doi 10.1007/978-3-030-92864-3_31.
48. G. Skryabin, T. Gadieva, and A. Antonov, "A New Version of the AlgoView System for 3D Visualization and Interactive Analysis of Information Graphs of Algorithms," Communications in Computer and Information Science. **2241**, 19–33 (2024). doi 10.1007/978-3-031-73372-7_2.
49. List of algorithms — Wikipedia. https://en.wikipedia.org/wiki/List_of_algorithms. Cited November 30, 2025.
50. DSA Tutorial — Learn Data Structures and Algorithms — GeeksforGeeks. https://www.geeksforgeeks.org/dsa/dsa-tutorial-learn-data-structures-and-algorithms/. Cited November 30, 2025.
51. Parallel Algorithms of the Standard Template Library. https://www.modernescpp.com/index.php/parallel-algorithm-of-the-standard-template-library/. Cited November 30, 2025.
52. Algorithmica. https://algorithmica.org/. Cited November 30, 2025.
53. GitHub — Algorithmica-org/Algorithmica: A Computer Science Textbook. https://github.com/algorithmica-org/algorithmica. Cited November 30, 2025.
54. MAXimal :: algo. http://e-maxx.ru/algo/. Cited November 30, 2025.
55. Main Page — Algorithms for Competitive Programming. https://cp-algorithms.com/. Cited November 30, 2025.
56. Algorithm Repository. https://www.algorist.com/algorist.html. Cited November 30, 2025.
57. Algocode wiki. https://wiki.algocode.ru. Cited November 30, 2025.
58. Algorithms — IT wiki ru. https://www.it-wiki.com.ru/algorithms/. Cited October 13, 2025.
59. Algorithm Wiki. https://algorithm-wiki.csail.mit.edu/. Cited October 13, 2025.
60. Algorithm Wiki | Interactive algorithms. https://thimbleby.gitlab.io/algorithm-wiki-site. Cited November 30, 2025.
61. Dijkstra's algorithm | Algorithm Wiki. https://thimbleby.gitlab.io/algorithm-wiki-site/wiki/Dijkstras_algorithm/. Cited November 30, 2025.
62. Algorithms — Wikibooks, Open Books for an Open World. https://en.wikibooks.org/wiki/Algorithms. Cited November 30, 2025.
63. Algorithms | Brilliant Math & Science Wiki. https://brilliant.org/wiki/algorithm/. Cited November 30, 2025.
64. List of Algorithms. https://www.scriptol.com/programming/list-algorithms.php. Cited November 30, 2025.
65. Vl. V. Voevodin, "An Open AlgoWiki Encyclopedia of Algorithmic Features: from Mobile to Extreme Scale," Numerical Methods and Programming **16** (1), 99–111 (2015). doi 10.26089/NumMet.v16r111.
66. Vl. V. Voevodin, A. S. Antonov, and J. Dongarra, "AlgoWiki: an Open Encyclopedia of Parallel Algorithmic Features," Supercomputing Frontiers and Innovations **2** (1), 4–18 (2015). doi 10.14529/jsfi150101.
67. Vl. Voevodin, A. Antonov, and J. Dongarra, "Why is It Hard to Describe Properties of Algorithms?," Procedia Computer Science. **101**, 4–7 (2016). doi 10.1016/j.procs.2016.11.002.

2025, **26** (4), 548–568. doi 10.26089/NumMet.v26r436

68. A. Antonov, A. Frolov, I. Konshin, and Vl. Voevodin, "Hierarchical Domain Representation in the AlgoWiki Encyclopedia: From Problems to Implementations," in *Communications in Computer and Information Science* (Springer, Cham, 2018), **910**, pp. 3–15. doi 10.1007/978-3-319-99673-8_1.
69. A. Popov, D. Nikitenko, A. Antonov, and Vl. Voevodin, "Formal Model of Problems, Methods, Algorithms and Implementations in the Advancing AlgoWiki Open Encyclopedia," CEUR Workshop Proc. **2281**, 1–11 (2018). https://ceur-ws.org/Vol-2281/paper-01.pdf. Cited November 30, 2025.
70. A. S. Antonov, "Extraction of an Explicit Level of Algorithms Implementation for Use in the Algo500 Project," Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering **12** (1), 89–100 (2023). doi 10.14529/cmse230105.

| *Received* | *Accepted* | *Published* |
|---|---|---|
| *October 27, 2025* | *November 27, 2025* | *December 5, 2025* |

### Information about the author

*Alexander S. Antonov* — Ph.D., Leading Researcher; Lomonosov Moscow State University, Research Computing Center, Leninskie Gory, 1, building 4, 119234, Moscow, Russia.