ВикипедиЯ

Алгоритм сортировки

Материал из Википедии — свободной энциклопедии

Алгоритм сортировки — это алгоритм для упорядочивания элементов в списке. В случае, когда элемент списка имеет несколько полей, поле, служащее критерием порядка, называется ключом сортировки. На практике в качестве ключа часто выступает число, а в остальных полях хранятся какие-либо данные, никак не влияющие на работу алгоритма.

Содержание

История

Первые прототипы современных методов сортировки появились уже в XIX веке. К 1890 году для ускорения обработки данных переписи населения в США американец Герман Холлерит создал первый статистический табулятор — электромеханическую машину, предназначенную для автоматической обработки информации, записанной на перфокартах[1]. У машины Холлерита имелся специальный «сортировальный ящик» из 26 внутренних отделений. При работе с машиной от оператора требовалось вставить перфокарту и опустить рукоятку. Благодаря пробитым на перфокарте отверстиям замыкалась определённая электрическая цепь, и на единицу увеличивалось показание связанного с ней циферблата. Одновременно с этим открывалась одна из 26 крышек сортировального ящика, и в соответствующее отделение перемещалась перфокарта, после чего крышка

закрывалась. Данная машина позволила обрабатывать около 50 карт в минуту, что ускорило обработку данных в 3 раза. К переписи населения 1900 года Холлерит усовершенствовал машину, автоматизировав подачу карт[1]. Работа сортировальной машины Холлерита основывалась на методах поразрядной сортировки. В патенте на машину обозначена сортировка «по отдельности для каждого столбца», но не определён порядок. В другой аналогичной машине, запатентованной в 1894 году Джоном Гором, упоминается сортировка со столбца десятков[2]. Метод сортировки, начиная со столбца единиц, впервые появляется в литературе в конце 1930-х годов[3]. К этому времени сортировальные машины уже позволяли обрабатывать до 400 карт в минуту[4].

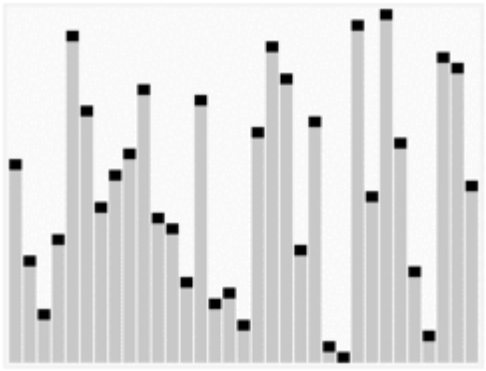
Табулятор Холлерита с «сортировальным ящиком»

EDVAC

В дальнейшем история алгоритмов оказалась связана с развитием электронно-вычислительных машин. По некоторым источникам, именно программа сортировки стала первой программой для вычислительных машин. Некоторые конструкторы ЭВМ, в частности разработчики EDVAC, называли задачу сортировки данных наиболее характерной нечисловой задачей для вычислительных машин. В 1945 году Джон фон Нейман для тестирования ряда команд для EDVAC разработал программы сортировки методом слияния. В том же году немецкий инженер Конрад Цузе разработал программу для сортировки методом простой вставки. К этому времени уже появились быстрые специализированные сортировальные машины, в сопоставлении с которыми и оценивалась эффективность разрабатываемых ЭВМ[4]. Первым опубликованным обсуждением сортировки с помощью вычислительных машин стала лекция Джона Мокли, прочитанная им в 1946 году. Мокли показал, что сортировка может быть полезной также и для численных расчетов, описал методы сортировки простой вставки и бинарных вставок, а также поразрядную сортировку с частичными проходами. Позже организованная им совместно с инженером Джоном Эккертом компания «Eckert–Mauchly Computer Corporation» выпустила некоторые из самых ранних электронных вычислительных машин BINAC и UNIVAC[5]. Наряду с отмеченными алгоритмами внутренней сортировки, появлялись алгоритмы внешней сортировки, развитию которых способствовал ограниченный объём памяти первых вычислительных машин[4]. В частности, были предложены методы сбалансированной двухпутевой поразрядной сортировки и сбалансированного двухпутевого слияния[5].

К 1952 году на практике уже применялись многие методы внутренней сортировки, но теория была развита сравнительно слабо[6]. В октябре 1952 года Даниэль Гольденберг привёл пять методов сортировки с анализом наилучшего и наихудшего случаев для каждого из них. В 1954 году Гарольд Сьюворд развил идеи Гольденберга, а также проанализировал методы внешней сортировки. Говард Демут в 1956 году рассмотрел три абстрактные модели задачи сортировки: с использованием циклической памяти, линейной памяти и памяти с произвольным доступом. Для каждой из этих задач автор предложил оптимальные или почти оптимальные методы сортировки, что помогло связать теорию с практикой[7]. Из-за малого числа людей, связанных с

вычислительной техникой, эти доклады не появлялись в «открытой литературе». Первой большой обзорной статьёй о сортировке, появившейся в печати в 1955 году, стала работа Дж. Хоскена, в которой он описал всё имевшееся на тот момент оборудование специального назначения и методы сортировки для ЭВМ, основываясь на брошюрах фирм-изготовителей. В 1956 году Э. Френд в своей работе проанализировал математические свойства большого числа алгоритмов внутренней и внешней сортировки, предложив некоторые новые методы[8].



Быстрая сортировка Хоара

После этого было предложено множество различных алгоритмов сортировки: например, вычисление адреса в 1956 году; слияние с вставкой, обменная поразрядная сортировка, каскадное слияние и метод Шелла в 1959 году, многофазное слияние и вставки в дерево в 1960 году, осциллирующая сортировка и быстрая сортировка Хоара в 1962 году, пирамидальная сортировка Уильямса и обменная сортировка со слиянием Бэтчера в 1964 году. В конце 60-х годов произошло и интенсивное развитие теории сортировки[9]. Появившиеся позже алгоритмы во многом являлись вариациями уже известных методов. Получили распространение адаптивные методы сортировки, ориентированные на более быстрое выполнение в случаях, когда входная последовательность удовлетворяет заранее установленным критериям[9].

Формулировка задачи

Пусть требуется упорядочить *N* элементов: $R_1, R_2, \ldots, R_n$. Каждый элемент представляет из себя запись R_j, содержащую некоторую информацию и ключ K_j, управляющий процессом сортировки. На множестве ключей определено отношение порядка «<» так, чтобы для любых трёх значений ключей a, b, c выполнялись следующие условия[10]:

- закон трихотомии: либо $a < b$, либо $a > b$, либо $a = b$;
- закон транзитивности: если $a < b$ и $b < c$, то $a < c$.

Данные условия определяют математическое понятие линейного или совершенного упорядочения, а удовлетворяющие им множества поддаются сортировке большинством методов[10].

Задачей сортировки является нахождение такой перестановки записей $p(1)p(2)\ldots p(n)$ с индексами $\{1, 2, \ldots, N\}$, после которой ключи расположились бы в порядке неубывания[10]:

$$K_{p(1)} \leqslant K_{p(2)} \leqslant \cdots \leqslant K_{p(n)}$$

Сортировка называется устойчивой, если не меняет взаимного расположения элементов с одинаковыми ключами[10]:

$$p(i) < p(j) \text{ для любых } K_{p(i)} = K_{p(j)} \text{ и } i < j.$$

Методы сортировки можно разделить на внутренние и внешние. Внутренняя сортировка используется для данных, помещающихся в оперативную память, за счёт чего является более гибкой в плане структур данных. Внешняя сортировка применяется, когда данные в оперативную память не помещаются, и ориентирована на

достижение результата в условиях ограниченных ресурсов[11].

Оценка алгоритма сортировки

Алгоритмы сортировки оцениваются по скорости выполнения и эффективности использования памяти:

- **Время** — основной параметр, характеризующий быстродействие алгоритма. Называется также вычислительной сложностью. Для упорядочения важны *худшее*, *среднее* и *лучшее* поведение алгоритма в терминах мощности входного множества $\mathbf{A}$. Если на вход алгоритму подаётся множество $\mathbf{A}$, то обозначим $n = |\mathbf{A}|$. Для типичного алгоритма хорошее поведение — это $O(n \log n)$ и плохое поведение — это $O(n^2)$. Идеальное поведение для упорядочения — $O(n)$. Алгоритмы сортировки, использующие только абстрактную операцию сравнения ключей всегда нуждаются по меньшей мере в сравнениях. Тем не менее, существует алгоритм сортировки Хана (Yijie Han) с вычислительной сложностью $O(n \log \log n \log \log \log n)$, использующий тот факт, что пространство ключей ограничено (он чрезвычайно сложен, а за O-обозначением скрывается весьма большой коэффициент, что делает невозможным его применение в повседневной практике). Также существует понятие сортирующих сетей. Предполагая, что можно одновременно (например, при параллельном вычислении) проводить несколько сравнений, можно отсортировать n чисел за $O(\log^2 n)$ операций. При этом число n должно быть заранее известно;
- **Память** — ряд алгоритмов требует выделения дополнительной памяти под временное хранение данных. Как правило, эти алгоритмы требуют $O(\log n)$ памяти. При оценке не учитывается место, которое занимает исходный массив и независящие от входной последовательности затраты, например, на хранение кода программы (так как всё это потребляет $O(1)$). Алгоритмы сортировки, не потребляющие дополнительной памяти, относят к *сортировкам на месте*.

Оптимальность $O(n \log(n))$

Задача сортировки в общем случае предполагает, что единственной обязательно наличествующей операцией для элементов является сравнение. Это делает невозможным реализацию алгоритма Хана, использующего арифметические действия. Рассмотрим схему алгоритма, когда единственным возможным действием над элементами является их сравнение.

Пусть по ходу работы алгоритмом производится k сравнений. Ответом на сравнение двух элементов a и b может быть один из двух вариантов ($a < b$ или $a > b$). Значит, всего возможно 2^k вариантов комбинаций ответов на k вопросов.

Количество перестановок из n элементов равно $n!$. Для того, чтобы можно было провести сюръекцию из множества ответов на сравнения *на* множество перестановок, количество сравнений должно быть не меньше, чем $\log_2 n!$ (это обеспечивается тем, что сравнение — единственная разрешённая операция[12]).

Прологарифмировав формулу Стирлинга, можно обнаружить, что $\log_2 n! = \log_2 \left(\sqrt{2\pi n}\left(\frac{n}{e}\right)^n\right) = \log_2 \sqrt{2\pi n} + n \log_2 n - n \log_2 e \sim n \log_2 n = O(n \log n)$

Свойства и классификация

- **Устойчивость** (англ. *stability*) — устойчивая сортировка не меняет взаимного расположения элементов с одинаковыми ключами[13].
- **Естественность поведения** — эффективность метода при обработке уже упорядоченных или частично упорядоченных данных. Алгоритм ведёт себя естественно, если учитывает эту характеристику входной последовательности и работает лучше.
- **Использование операции сравнения.** Алгоритмы, использующие для сортировки сравнение элементов между собой, называются основанными на сравнениях. Минимальная трудоемкость *худшего случая* для этих алгоритмов составляет $O(n \cdot \log n)$, но они отличаются гибкостью применения. Для специальных случаев (типов данных) существуют более эффективные алгоритмы.

Ещё одним важным свойством алгоритма является его сфера применения. Здесь основных типов упорядочения два:

- **Внутренняя сортировка** оперирует массивами, целиком помещающимися в оперативной памяти с произвольным доступом к любой ячейке. Данные обычно упорядочиваются на том же месте без дополнительных затрат.
  - В современных архитектурах персональных компьютеров широко применяется подкачка и кэширование памяти. Алгоритм сортировки должен хорошо сочетаться с применяемыми алгоритмами кэширования и подкачки.
- **Внешняя сортировка** оперирует запоминающими устройствами большого объёма, но не с произвольным доступом, а последовательным (упорядочение файлов), то есть в данный момент «виден» только один элемент, а затраты на перемотку по сравнению с памятью неоправданно велики. Это накладывает некоторые дополнительные ограничения на алгоритм и приводит к специальным методам упорядочения, обычно использующим дополнительное дисковое пространство. Кроме того, доступ к данным во внешней памяти производится намного медленнее, чем операции с оперативной памятью.
  - Доступ к носителю осуществляется последовательным образом: в каждый момент времени можно считать или записать только элемент, следующий за текущим.
  - Объём данных не позволяет им разместиться в ОЗУ.

Также алгоритмы классифицируются по:

- потребности в дополнительной памяти или её отсутствию
- потребности в знаниях о структуре данных, выходящих за рамки операции сравнения, или отсутствию таковой

Список алгоритмов сортировки

В этой таблице n — это количество записей, которые необходимо упорядочить, а k — это количество уникальных ключей.

Алгоритмы устойчивой сортировки

- Сортировка пузырьком (англ. *Bubble sort*) — для каждой пары индексов производится обмен, если элементы расположены не по порядку. Сложность алгоритма: $O(n^2)$.
- Сортировка перемешиванием (англ. *Cocktail sort*). Сложность алгоритма: $O(n^2)$.
- Сортировка вставками (Insertion sort) — определяем, где текущий элемент должен

находиться в упорядоченном списке, и вставляем его туда. Сложность алгоритма: $O(n^2)$.

- Гномья сортировка (Gnome sort; первоначально опубликована под названием «глупая сортировка» [stupid sort] за простоту реализации) — сходна с сортировкой вставками. Сложность алгоритма — $O(n^2)$; рекурсивная версия требует дополнительно $O(n^2)$ памяти.
- Сортировка слиянием (Merge sort) — выстраиваем первую и вторую половину списка отдельно, а затем объединяем упорядоченные списки. Сложность алгоритма: $O(n \log n)$. Требуется $O(n)$ дополнительной памяти.
- Сортировка с помощью двоичного дерева (англ. *Tree sort*). Сложность алгоритма: $O(n \log n)$. Требуется $O(n)$ дополнительной памяти.
- Сортировка Timsort (англ. *Timsort*) — комбинированный алгоритм (используется сортировка вставками и сортировка слиянием). Сложность алгоритма: $O(n \log n)$. Требуется $O(n)$ дополнительной памяти. Разработан для использования в языке Python[14].
- Сортировка подсчётом (Counting sort). Сложность алгоритма: $O(n + k)$. Требуется $O(n + k)$ дополнительной памяти.
- Блочная сортировка (Корзинная сортировка, Bucket sort) — требуется $O(k)$ дополнительной памяти и знание о природе сортируемых данных, выходящее за рамки функций «переставить» и «сравнить». Сложность алгоритма: $O(n)$.
- Поразрядная сортировка (она же *цифровая сортировка*, Radix sort) — сложность алгоритма: $O(nk)$; требуется $O(k)$ дополнительной памяти.

Алгоритмы неустойчивой сортировки

- Сортировка выбором (англ. *Selection sort*) — поиск наименьшего или наибольшего элемента и помещение его в начало или конец упорядоченного списка. Сложность алгоритма: $O(n^2)$.
- Сортировка расчёской (Comb sort) — сложность алгоритма: $O(n^2)$; улучшение сортировки пузырьком.
- Сортировка Шелла (Shell sort) — улучшение сортировки вставками. Сложность алгоритма меняется в зависимости от выбора последовательности длин промежутков; при определённом выборе (см. статью), возможно обеспечить сложность $O(n^{\frac{4}{3}})$ или $O(n \log^2 n)$.
- Пирамидальная сортировка (сортировка кучи, Heapsort) — сложность алгоритма: $O(n \log n)$; превращаем список в кучу, берём наибольший элемент и добавляем его в конец списка
- Плавная сортировка (Smoothsort) — сложность алгоритма $O(n \log n)$
- Быстрая сортировка (Quicksort), в варианте с минимальными затратами памяти — сложность алгоритма: $O(n \log n)$ — среднее время, $O(n^2)$ — худший случай; широко известен как быстрейший из известных для упорядочения больших случайных списков; с разбиением исходного набора данных на две половины так, что любой элемент первой половины упорядочен относительно любого элемента второй половины; затем алгоритм применяется рекурсивно к каждой половине. При использовании $O(n)$ дополнительной памяти, можно сделать сортировку устойчивой.
- Интроспективная сортировка (Introsort) — сложность алгоритма: $O(n \log n)$, сочетание быстрой и пирамидальной сортировки. Пирамидальная сортировка применяется в случае, если глубина рекурсии превышает $\log n$.
- Терпеливая сортировка (Patience sorting) — сложность алгоритма: $O(n \log n)$ — наихудший случай, требует дополнительно $O(n)$ памяти, также находит самую

длинную увеличивающуюся подпоследовательность
- Stooge sort — рекурсивный алгоритм сортировки с временной сложностью $O(n^{\log_{1,5} 3}) \approx O(n^{2.71})$.

Непрактичные алгоритмы сортировки

- Bogosort (также *глупая сортировка*, stupid sort) — $O(n \cdot n!)$ в среднем. Произвольно перемешать массив, проверить порядок.
- Сортировка перестановкой — $O(n \cdot n!)$ — худшее время. Для каждой пары осуществляется проверка верного порядка и генерируются всевозможные перестановки исходного массива.
- Bead Sort — $O(n)$ или $O(\sqrt{n})$, требуется специализированное аппаратное обеспечение
- Блинная сортировка (Pancake sorting) — $O(n)$, требуется специализированное аппаратное обеспечение

Алгоритмы, не основанные на сравнениях

- Блочная сортировка (Корзинная сортировка, Bucket sort)
- Лексикографическая или поразрядная сортировка (Radix sort)
- Сортировка подсчётом (Counting sort)

Прочие алгоритмы сортировки

- Топологическая сортировка
- Внешняя сортировка

Сортировка строк

Одним из наиболее частых приложений алгоритмов сортировки является сортировка строк. Обычно она производится так: сначала множество строк сортируется по первому символу каждой строки, затем каждое подмножество строк, имеющих одинаковый первый символ, сортируется по второму символу, и так до тех пор, пока все строки не будут упорядочены. При этом отсутствующий символ (при сравнении строки длины N со строкой длины N+1) считается меньше любого символа.

Применение данного метода к строкам, представляющим собой числа в естественной записи, выдаёт контринтуитивные результаты: например, «9» оказывается больше, чем «11», так как первый символ первой строки имеет бóльшее значение, чем первый символ второй. Для исправления этой проблемы алгоритм сортировки может преобразовывать сортируемые строки в числа и сортировать их как числа. Такой алгоритм называется «числовой сортировкой», а описанный ранее — «строковой сортировкой».

См. также

- О-большое
- Временная сложность алгоритма

Примечания

1. Кнут, 2007, с. 416.
2. Кнут, 2007, с. 417.
3. Кнут, 2007, с. 417-418.
4. Кнут, 2007, с. 418.
5. Кнут, 2007, с. 419.
6. Кнут, 2007, с. 420.
7. Кнут, 2007, с. 420-421.
8. Кнут, 2007, с. 421.
9. Кнут, 2007, с. 422.
10. Кнут, 2007, с. 22.
11. Кнут, 2007, с. 23.
12. *Дональд Кнут.* 5.3.1. Сортировка с минимальным числом сравнений // Искусство программирования. — 2-е. — Вильямс, 2002.
13. Кнут, 2007.
14. Hetland, 2010.

Литература

- *Кнут Д. Э.* Искусство программирования. Том 3. Сортировка и поиск (https://books.google.ru/books?id=92rW-nktlbgC&printsec=frontcover&dq=editions:spjjKVwoQ3QC&hl=ru&sa=X&ved=0ahUKEwiUjLbc88rQAhVCfiwKHUJlBQoQuwUIIjAB#v=onepage&q&f=false) = The Art of Computer Programming. Volume 3. Sorting and Searching / под ред. В. Т. Тертышного (гл. 5) и И. В. Красикова (гл. 6). — 2-е изд. — Москва: Вильямс, 2007. — Т. 3. — 832 с. — ISBN 5-8459-0082-1.
- *Томас Х. Кормен, Чарльз И. Лейзерсон, Рональд Л. Ривест, Клиффорд Штайн.* Алгоритмы: построение и анализ = INTRODUCTION TO ALGORITHMS. — 2-е изд. — М.: «Вильямс», 2006. — С. 1296. — ISBN 5-8459-0857-4.
- *Роберт Седжвик.* Фундаментальные алгоритмы на C. Анализ/Структуры данных/Сортировка/Поиск = Algorithms in C. Fundamentals/Data Structures/Sorting/Searching. — СПб.: ДиаСофтЮП, 2003. — С. 672. — ISBN 5-93772-081-4.
- *Magnus Lie Hetland.* Python Algorithms: Mastering Basic Algorithms in the Python Language. — Apress, 2010. — 336 с. — ISBN 978-1-4302-3237-7.

Ссылки

- Теория, задачи, тестирующая система (http://informatics.mccme.ru/moodle/course/view.php?id=3)
- Алгоритмы сортировки на algolist.manual.ru (http://algolist.manual.ru/sort/index.php)
- Анимированное сравнение алгоритмов сортировки (http://www.sorting-algorithms.com) (англ.)

случаях могут действовать дополнительные условия.