ВикипедиЯ

Сортировка пузырьком

Материал из Википедии — свободной энциклопедии

Сортировка простыми обменами, **сортиро́вка пузырько́м** (англ. *bubble sort*) — простой алгоритм сортировки. Для понимания и реализации этот алгоритм — простейший, но эффективен он лишь для небольших массивов. Сложность алгоритма: $O(n^2)$.

Алгоритм считается учебным и практически не применяется вне учебной литературы, вместо него на практике применяются более эффективные алгоритмы сортировки. В то же время метод сортировки обменами лежит в основе некоторых более совершенных алгоритмов, таких как шейкерная сортировка, пирамидальная сортировка и быстрая сортировка.

Содержание

Алгоритм

Алгоритм состоит из повторяющихся проходов по сортируемому массиву. За каждый проход элементы последовательно сравниваются попарно и, если порядок в паре неверный, выполняется обмен элементов. Проходы по массиву повторяются $N-1$ раз или до тех пор, пока на очередном проходе не окажется, что обмены больше не нужны, что означает — массив отсортирован. При каждом проходе алгоритма по внутреннему циклу, очередной наибольший элемент массива ставится на своё место в конце массива рядом с предыдущим «наибольшим элементом», а наименьший элемент перемещается на одну позицию к началу массива («всплывает» до нужной позиции, как пузырёк в воде. Отсюда и название алгоритма).

Реализация

Сложность: $O(n^2)$.

Наихудший случай:

- Число сравнений в теле цикла равно $(N-1)\frac{N}{2}$.

- Число сравнений в заголовках циклов равно $(N-1)\frac{N}{2}$.
- Суммарное число сравнений равно $(N-1)N$.
- Число присваиваний в заголовках циклов равно $(N-1)\frac{N}{2}$.
- Число обменов равно $(N-1)\frac{N}{2}$, что в $\frac{N}{2}$ раз больше, чем в сортировке выбором.

Наилучший случай (на вход подаётся уже отсортированный массив):

- Число сравнений в теле цикла равно $(N-1)\frac{N}{2}$.
- Число сравнений в заголовках циклов равно $(N-1)\frac{N}{2}$.
- Суммарное число сравнений равно $(N-1)N$.
- Число обменов равно 0.

Особенность данного алгоритма заключается в следующем: после первого завершения внутреннего цикла максимальный элемент массива всегда находится на N-ой позиции. При втором проходе, следующий по значению максимальный элемент находится на $N-1$ месте. И так далее. Таким образом, на каждом следующем проходе число обрабатываемых элементов уменьшается на 1 и нет необходимости «обходить» весь массив от начала до конца каждый раз.

Так как подмассив из одного элемента не нуждается в сортировке, то для сортировки требуется делать не более $N-1$ итераций внешнего цикла. Поэтому в некоторых реализациях внешний цикл всегда выполняется ровно $N-1$ и не отслеживается, были или не были обмены на каждой итерации.

Введение индикатора (флажка F) действительно произошедших во внутреннем цикле обменов уменьшает число лишних проходов в случаях с частично отсортированными массивами на входе. Перед каждым проходом по внутреннему циклу флажок сбрасывается в 0, а после действительно произошедшего обмена устанавливается в 1. Если после выхода из внутреннего цикла флажок равен 0, то обменов не было, то есть массив отсортирован и можно досрочно выйти из программы сортировки.

Псевдокод ещё более улучшенного алгоритма с проверкой действительно произошедших обменов во внутреннем цикле.

На входе: массив $A[N]$, состоящий из N элементов, с нумерацией от $A[1]$ до $A[N]$

```
ЦИКЛ ДЛЯ J=1 ДО N-1 ШАГ 1
  F=0
  ЦИКЛ ДЛЯ I=1 ДО N-J ШАГ 1
    ЕСЛИ A[I] > A[I+1] ТО ОБМЕН A[I],A[I+1]:F=1
  СЛЕДУЮЩЕЕ I
  ЕСЛИ F=0 ТО ВЫХОД ИЗ ЦИКЛА
СЛЕДУЮЩЕЕ J
```

```
FOR J=1 TO N-1 STEP 1
  F=0
  FOR I=1 TO N-J STEP 1
    IF A[I]>A[I+1] THEN SWAP A[I],A[I+1]:F=1
  NEXT I
  IF F=0 THEN EXIT FOR
NEXT J
```

В случае досрочного выхода из сортировки в этом алгоритме делается один избыточный проход без обменов.

Наихудший случай (не улучшается):

- Число сравнений в теле цикла равно $(N-1)\frac{N}{2}$.

- Число сравнений в заголовках циклов $(N-1)\frac{N}{2}$.
- Суммарное число сравнений равно $(N-1)N$.
- Число присваиваний в заголовках циклов равно $(N-1)\frac{N}{2}$.
- Число обменов равно $(N-1)\frac{N}{2}$.

Наилучший случай (улучшается):

- Число сравнений в теле цикла равно $(N-1)$.
- Число сравнений в заголовках циклов $(N-1)$.
- Суммарное число сравнений равно $2(N-1)$.
- Число обменов равно 0.

Время сортировки 10000 коротких целых чисел на одном и том же программно-аппаратном комплексе (операция сравнения ≈3.4мкс, обмена ≈2.3мкс) сортировкой выбором составило ≈40сек., ещё более улучшенной сортировкой пузырьком ≈30сек, а быстрой сортировкой ≈0,027сек.

$O(n \cdot n)$ больше, чем $O(n \cdot \log n)$ у сортировки слиянием, но при малых n разница не очень большая, а программный код очень прост, поэтому вполне допустимо применение сортировки пузырьком для множества задач с массивами малой размерности на простаивающих и малозагруженных машинах.

Алгоритм можно немного улучшить, сделав следующее:

- Внутренний цикл можно модифицировать так, чтобы он поочерёдно просматривал массив то с начала, то с конца. Модифицированный таким образом алгоритм называется сортировкой перемешиванием или шейкерной сортировкой. Сложность при этом $O(n \cdot n)$ не уменьшается.

В сортировке пузырьком, при каждом проходе по внутреннему циклу, можно добавить определение очередного минимального элемента и помещение его в начало массива, то есть объединить алгоритмы сортировки пузырьком и сортировки выбором, при этом число проходов по внутреннему циклу сокращается вдвое, но более чем вдвое увеличивается число сравнений и добавляется один обмен после каждого прохода по внутреннему циклу.

Псевдокод объединённого алгоритма сортировки пузырьком и сортировки выбором (устойчивая реализация):

```
FOR J=1 TO N-1 STEP 1
   F=0
   MIN=J
   FOR I=J TO N-J STEP 1
      IF Y[I]>Y[I+1] THEN SWAP Y[I],Y[I+1]:F=1
      IF Y[I]<Y[MIN] THEN MIN=I
      NEXT I
   IF F=0 THEN EXIT FOR
   IF MIN<>J THEN SWAP Y[J],Y[MIN]
   NEXT J
```

C

```
1 int      *bubble_sort(int *array, int array_size)
2 {
3  int i = 0;
```

```
 4  int buf;
 5  char swap_cnt = 0;
 6
 7  if (array_size == 0)
 8      return (0);
 9  while (i < array_size)
10  {
11      if (i + 1 != array_size && array[i] > array[i + 1])
12      {
13          buf = array[i];
14          array[i] = array[i + 1];
15          array[i + 1] = buf;
16          swap_cnt = 1;
17      }
18      i++;
19      if (i == array_size && swap_cnt == 1)
20      {
21          swap_cnt = 0;
22          i = 0;
23      }
24  }
25  return (array);
26 }
```

Пример работы алгоритма

Возьмём массив с числами «5 1 4 2 8» и отсортируем значения по возрастанию, используя сортировку пузырьком. Выделены те элементы, которые сравниваются на данном этапе.

Первый проход:

(**5** **1** 4 2 8) (**1** **5** 4 2 8), Здесь алгоритм сравнивает два первых элемента и меняет их местами.
(1 **5** **4** 2 8) (1 **4** **5** 2 8), Меняет местами, так как $5 > 4$
(1 4 **5** **2** 8) (1 4 **2** **5** 8), Меняет местами, так как $5 > 2$
(1 4 2 **5** **8**) (1 4 2 **5** **8**), Теперь, ввиду того, что элементы стоят на своих местах ($8 > 5$), алгоритм не меняет их местами.

6 5 3 1 8 7 2 4

Наглядная демонстрация алгоритма.

Второй проход:

(**1** **4** 2 5 8) (**1** **4** 2 5 8)
(1 **4** **2** 5 8) (1 **2** **4** 5 8), Меняет местами, так как $4 > 2$
(1 2 **4** **5** 8) (1 2 **4** **5** 8)

Теперь массив полностью отсортирован, но алгоритму это неизвестно. Поэтому ему необходимо сделать полный проход и определить, что перестановок элементов не было.

Третий проход:

(**1** **2** 4 5 8) (**1** **2** 4 5 8)
(1 **2** **4** 5 8) (1 **2** **4** 5 8)

Теперь массив отсортирован и алгоритм может быть завершён.

Примечания

Все приведённые реализации алгоритма имеют недостаток: каждый раз во внешнем цикле длина проходимого участка уменьшается на единицу, в то время как следующий проход должен идти до места последнего обмена.

Ссылки

- http://sorting.at/ Анимация алгоритмов сортировки
- Динамическая визуализация 7 алгоритмов сортировки с открытым исходным кодом (https://airtucha.github.io/SortVis/)

Литература

- *Левитин А. В.* Глава 3. Метод грубой силы: Пузырьковая сортировка // Алгоритмы. Введение в разработку и анализ — М.: Вильямс, 2006. — С. 144–146. — 576 с. — ISBN 978-5-8459-0987-9

Источник — https://ru.wikipedia.org/w/index.php?title=Сортировка_пузырьком&oldid=95595792

Эта страница в последний раз была отредактирована 13 октября 2018 в 20:19.

Текст доступен по лицензии Creative Commons Attribution-ShareAlike; в отдельных случаях могут действовать дополнительные условия.
Wikipedia® — зарегистрированный товарный знак некоммерческой организации Wikimedia Foundation, Inc.