Быстрая сортировка

QuickSort

Wikipedia: Tony Hoare

Sir Charles Antony Richard Hoare (/hɔːr/; born 11 January 1934), also known as **C. A. R. Hoare**, is a British computer scientist who has made foundational contributions to programming languages, algorithms, operating systems, formal verification, and concurrent computing.[3]
Hoare developed the sorting algorithm quicksort in 1959–1960.

Education and early life

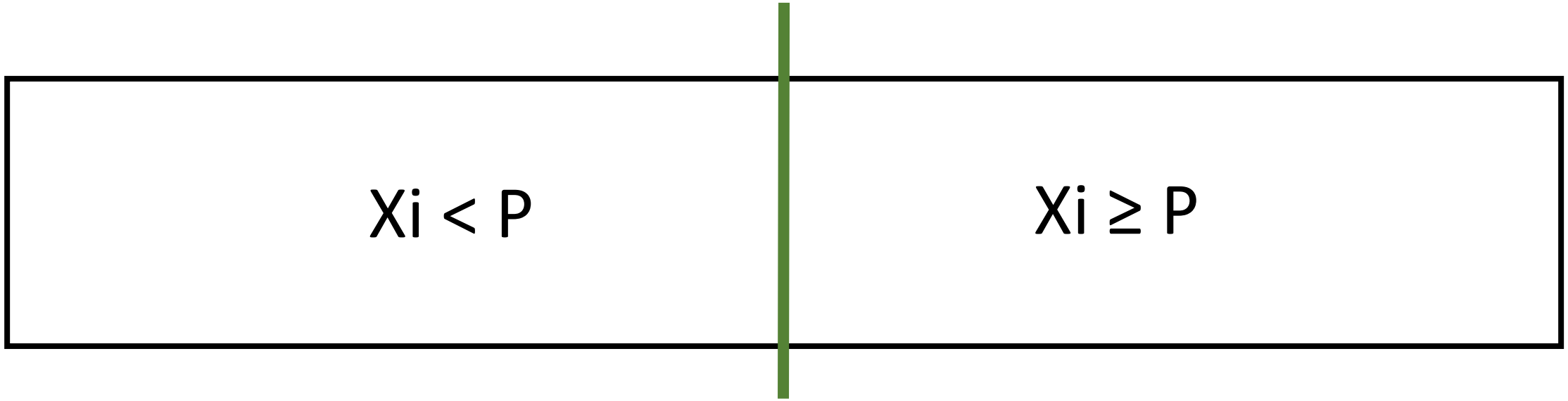
He went to Moscow State University as a British Council exchange student,[13] where he studied machine translation under Andrey Kolmogorov.[14]

Sir Tony Hoare
FRS FREng

Tony Hoare in 2011

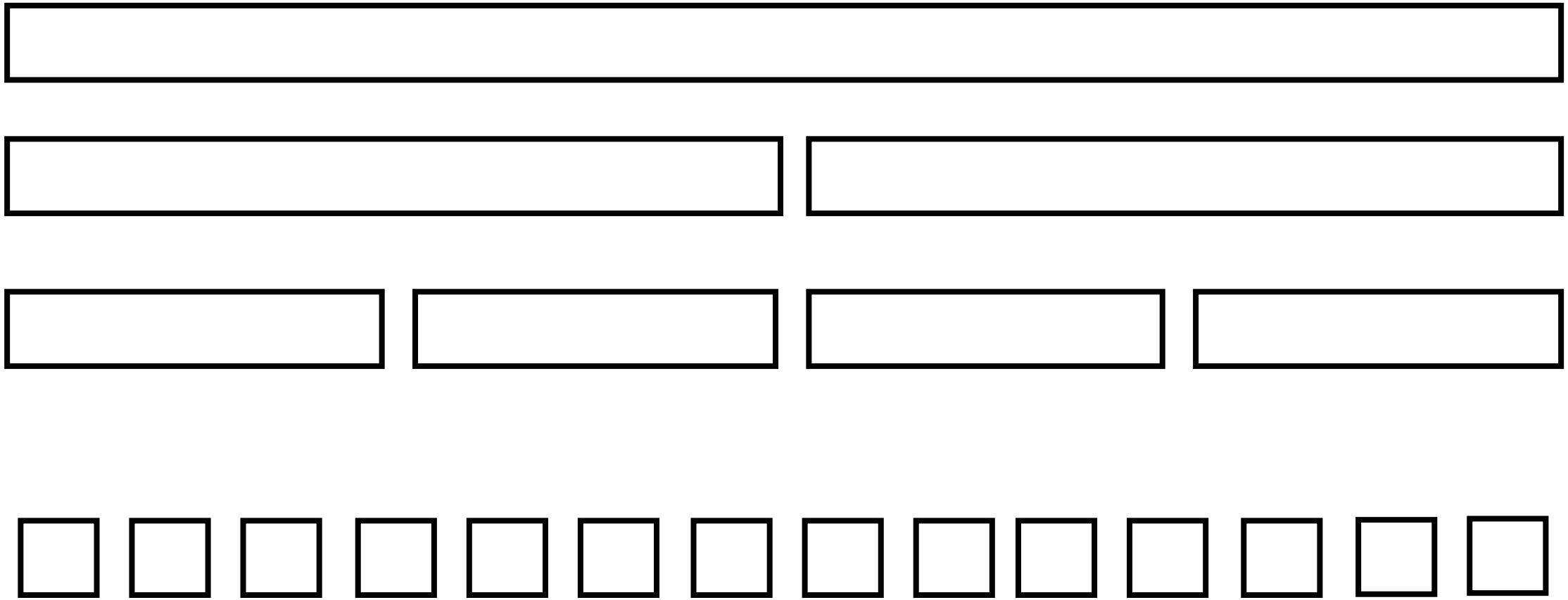
Born	Charles Antony Richard Hoare 11 January 1934 (age 91) Colombo, British Ceylon
Education	Merton College, Oxford (BA, PgDip) Moscow State University
Known for	Quicksort

Разбиение массива / Partition

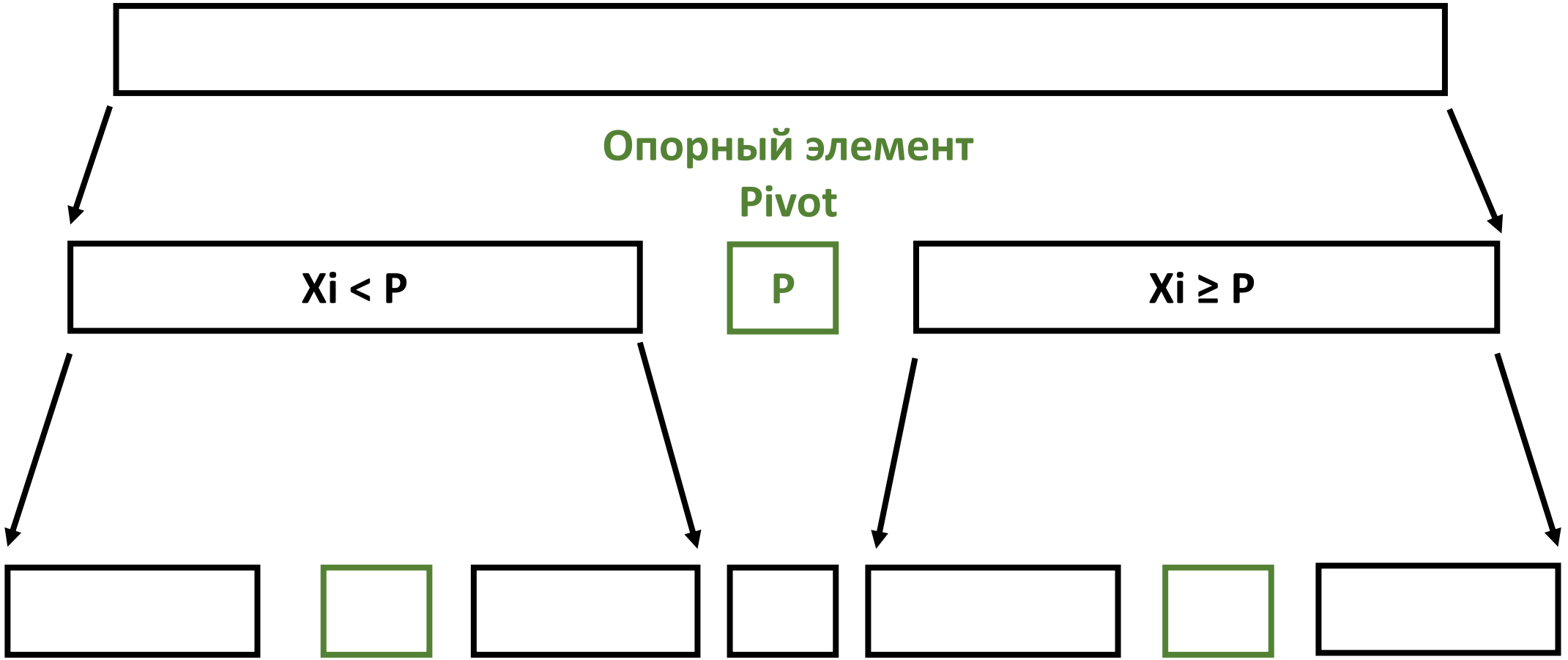


Граница между подмассивами

Вычислительная сложность



Рекурсивное разбиение / Partition



Наивная реализация

```
1  def qsort(a):
2      if len(a) <= 1:
3          return a
4      p = a[0]
5      left = [x for x in a[1:] if x < p]
6      right  = [x for x in a[1:] if x >= p]
7      return qsort(left) + [p] + qsort(right)
8  a = [2, 1, 4, 6, 5, 3]
9  print(a)
10 print(qsort(a))
```

Quicksort

By C. A. R. Hoare

A description is given of a new method of sorting in the random-access store of a computer. The method compares very favourably with other known methods in speed, in economy of storage, and in ease of programming. Certain refinements of the method, which may be useful in the optimization of inner loops, are described in the second part of the paper.

Part One: Theory

The sorting method described in this paper is based on the principle of resolving a problem into two simpler subproblems. Each of these subproblems may be resolved to produce yet simpler problems. The process is repeated until all the resulting problems are found to be trivial. These trivial problems may then be solved by known methods, thus obtaining a solution of the original more complex problem.

ALGORITHM 64 QUICKSORT

C. A. R. HOARE
Elliott Brothers Ltd., Borehamwood, Hertfordshire, Eng.

```
procedure  quicksort (A,M,N); value M,N;
                array A; integer M,N;
comment  Quicksort is a very fast and convenient method of
sorting an array in the random-access store of a computer. The
entire contents of the store may be sorted, since no extra space is
required. The average number of comparisons made is 2(M−N) ln
(N−M), and the average number of exchanges is one sixth this
amount. Suitable refinements of this method will be desirable for
its implementation on any actual computer;
begin      integer I,J;
           if M < N then      partition (A,M,N,I,J);
                              quicksort (A,M,J);
                              quicksort (A, I, N)
end        quicksort
```

ALGORITHM 63 PARTITION C. A. R. HOARE

```
procedure  partition (A,M,N,I,J);  value M,N;
                array A;  integer M,N,I,J;
begin      real X;  integer F;
           F := random (M,N);  X := A[F];
           I := M;  J := N;
up:        for I := I step 1 until N do
                      if X < A [I] then go to down;
           I := N;
down:      for J := J  step −1 until M do
                      if A[J]<X then go to change;
           J := M;
change:    if I < J then          exchange (A[I], A[J]);
                                  I := I + 1; J := J − 1;
                                  go to up
else       if I < F then          exchange (A[I], A[F]);
                                  I := I + 1
else       if F < J then          exchange (A[F], A[J]);
                                  J := J − 1
end        partition
```

I and J are output variables,

$M \leqq J < I \leqq N$ provided $M < N$

$A[R] \leqq X$ for $M \leqq R \leqq J$

$A[R] = X$ for $J < R < I$

$A[R] \geqq X$ for $I \leqq R \leqq N$

Быстрая сортировка

- Внутренняя сортировка (ОЗУ)
- Использование памяти: in-place
- Принцип «Разделяй и властвуй»
- Разбиение и рекурсия
  - Quicksort()
  - Partition()
- Опорный элемент P
  - Pivot / Bound
  - Dividing line / Wall

Разделение на подмассивы

- Два указателя
- Левый указатель
  - Lower / left pointer
  - Начало подмассива
  - Увеличиваем, пока x[i] < A
- Правый указатель
  - Upper / right pointer
  - Конец подмассива
  - Уменьшаем, пока x[i] >= A
- Обмен значений
- Продолжаем до «перекрытия» указателей

Рекурсия

- Разделение массива:
  - Левый подмассив
  - Опорный элемент
  - Правый подмассив
- Рекурсивное разделение
  - Базовый случай
  - Размер подмассива = 0 или 1

Входной массив

- Наилучший случай
- Наихудший случай
- «Средний» случай

Крайние случаи

- Опорный элемент
  - Минимальный
  - Максимальный
- Все элементы одинаковые
- Элементы по возрастанию / убыванию

Оптимизация алгоритма

- Вероятностный алгоритм
  - *Randomized algorithm*
  - Предварительное перемешивание массива
  - Случайный выбор опорного элемента
- «Медианная сортировка»
  - Опорный элемент – медиана трех элементов
- Разбиение на три подмассива
- Схема разбиения
  - Lomuto partition scheme
  - Hoare partition scheme

- Совершенство достигается не тогда, когда уже нечего прибавить, но когда уже ничего нельзя отнять.
  - Антуан Мари Жан-Батист Роже де Сент-Экзюпери. Планета людей
- Perfection is finally attained not when there is no longer anything to add, but when there is no longer anything to take away.
- Il semble que la perfection soit atteinte non quand il n’y a plus rien à ajouter, mais quand il n’y a plus rien à retrancher.
  - Antoine de Saint Exupéry

Ссылки

- Hoare, C. A. R. (1961). Algorithm 63, Partition; Algorithm 64, Quicksort; ACM Communications, Vol. 4, p. 321.
  - https://dl.acm.org/doi/pdf/10.1145/366622.366644
- Hoare C. A. R. (1962) Quicksort. The Computer Journal, Volume 5, Issue 1, p. 10–16.
  - https://www.sci-hub.ru/10.1093/comjnl/5.1.10
- Sedgewick R. Implementing Quicksort programs. ACM Communications. Vol. 21 (10), 1978, pp. 847-857.
  - https://sedgewick.io/wp-content/themes/sedgewick/papers/1978Implementing.pdf
- Bentley J.L, McIlroy M.D. Engineering a sort function. Software: Practice and Experience, vol. 23(11), 1993, pp. 1249–1265.
  - https://cs.fit.edu/~pkc/classes/writing/papers/bentley93engineering.pdf
- Кормен Т. и др. Алгоритмы: построение и анализ. Гл.8. Быстрая сортировка
- Кнут Д. Искусство программирования т.3. Сортировка и поиск. 5.2.2.Обменная сортировка
- Википедия. Быстрая сортировка
  - https://ru.wikipedia.org/wiki/Быстрая_сортировка
- Гарвард. CS50 на русском. Короткие видео. Быстрая сортировка
  - https://clck.ru/3N6X2z
- Young&&Yandex. Тренировки по алгоритмам 4.0. Сортировки: быстрая, слиянием и поразрядная
  - https://vk.cc/cNFwER