Битонная сортировка.

Битонная сортировка (англ. Bitonic sorter) — параллельный алгоритм сортировки данных, метод для создания сортировочной сети. Разработан американским информатиком Кеннетом Батчером в 1968 году.

Определение:

битонная последовательность – последовательность элементов

$\{ a_n \} = (a_1; a_2; a_3; \dots; a_n)$,

в которой либо элементы до *k-го* образуют неубывающую последовательность, а после него – невозрастающую последовательность, либо приводится к такому виду с помощью циклического сдвига:

$a_1 \le a_2 \le a_3 \le \dots \le a_k \ge a_{k+1} \ge \dots \ge a_n$

Примеры битонных последовательностей:

$\{ a_n \} = (1; 2; 3; 4; 3; 2; 1)$,

$\{ a_n \} = (1; 2; 3; 4; 5; 6; 7)$,

$\{ a_n \} = (1; 10; 9; 8; 0; -3; -10)$.

1. Алгоритм сортировки битонной последовательности длины $n = 2^{k-1}$, $k \in \mathrm{N}$.

1. Если $n = 1$, то алгоритм завершается.

2. Разобьем данную последовательность на две:

$\{ b_n \} = (min(a_1, a_{m+1}); min(a_2, a_{m+2}); \dots; min(a_m, a_n));$

$\{ c_n \} = (max(a_1, a_{m+1}); max(a_2, a_{m+2}); \dots; max(a_m, a_n));$

где *m = n / 2.*

Заметим несколько фактов:

- $b_n \le c_n$, так как $min(a, b) \le max(a, b)$;
- $b_n -$ битонная последовательность.
- $c_n -$ битонная последовательность.
  Доказательство аналогично предыдущему.

3. Таким образом, если отсортировать последовательности b_n и c_n, ответом будет являться последовательность $\{ a_n \} = (b_1; b_2; \dots; b_m; c_1; c_2; \dots; c_m)$. Отсортировать последовательности можно рекурсивно этим же алгоритмом.

Асимптотика алгоритма 1:

Будем считать, что алгоритм состоит из $k = log_2 n$ фаз, на каждой из которых сортируются части массива длины $\frac{n}{2^{k-1}}$. Таким образом, на первой фазе выполняется $\frac{n}{2}$ сравнений. Далее массив делится на 2 равные части, в каждой из которых выполняется $\frac{n}{4}$ сравнений. И так далее, нетрудно видеть, что на каждой фазе, кроме последней, массивов вдвое больше, чем на предыдущей, а сравнений в каждом из них выполняется в 2 раза меньше. Значит всего выполняется $\frac{n}{2}k$ операций. Значит асимптотика данного алгоритма равна $O(nk) = O(nlogn)$.

2. Алгоритм преобразования последовательности длины $n = 2^{k-1}$, $k \in \mathrm{N}$ В битонную.

1. Если $n \leq 2$, то алгоритм завершается, так как любая последовательность длины такой длины является битонной.

2. Разделим последовательность на две:
$\{ b_n \} = (a_1;\ a_2; \dots;\ a_m)$;
$\{ c_n \} = (a_{m+1};\ a_{m+2}; \dots;\ a_n)$;

где *m = n / 2.*

Запустим этот алгоритм для построения битонных последовательностей из b_n и c_n.

3. Отсортируем алгоритмом 1 последовательность b_n по возрастанию и последовательность c_n по убыванию.

4. Возвращаем объединение двух последовательностей:

$\{ a_n \} = (b_1; b_2; \dots; b_m; c_1; c_2; \dots; c_m)$.

$b_1 \leq b_2 \leq \dots \leq b_m; c_1 \geq c_2 \geq \dots \geq c_m$;

Очевидно, что при любом результате сравнения $\mathrm{b_m}$ и c_1 последовательность $\{ a_n \}$ является битонной.

Асимптотика алгоритма 2:

Алгоритм рекурсивный, глубина рекурсии, очевидно, равна $log_2 n$. На каждом уровне выполняется $log_2 n$ операций (см. асимптотику алгоритма 1). Итоговая асимптотика: $O(nlog^2 n)$.

3. Алгоритм битонной сортировки.

1. Дополним массив минимальным количеством элементов, равных бесконечности, так, чтобы количество элементов в нем стало равно $n = 2^{k-1}$, $k \in \mathrm{N}$.

2. Преобразуем нашу последовательность алгоритмом 2 в битонную.

3. Отсортируем весь массив алгоритмом 1.

4. Выбросим элементы, равные бесконечности, из конца массива.

Асимптотика алгоритма битонной сортировки:

Очевидно, первая и четвертая части выполняются за $O(n)$. Для остальных асимптотика доказана: третья часть - $O(nlogn)$, вторая - $O(nlog^2n)$.

Асимптотика всего алгоритма: $O(nlog^2n + n + n + nlogn) = O(nlog^2n)$.

Пример:

$\{ a_n \} = (1; 5; 7; 2; 5; 6; 3; 4) \rightarrow (1; 5; 7; 2; 5; 6; 4; 3) \rightarrow (1; 2; 7; 5; 5; 6; 4; 3) \rightarrow (1; 2; 5; 7; 6; 5; 4; 3) \rightarrow (1; 2; 4; 3; 6; 5; 5; 7) \rightarrow (1; 2; 4; 3; 5; 5; 6; 7) \rightarrow (1; 2; 3; 4; 5; 5; 6; 7).$