УВАЖАЕМЫЕ СТУДЕНТЫ!

ВАМ НЕОБХОДИМО ВЫПОЛНИТЬ СЛЕДУЮЩЕЕ:

1. Ознакомиться с лекцией и законспектируйте
2. Предоставит отчет конспекта лекции прислать в виде скриншото в течении трех дней.
3. Отправить преподавателю на почту v.vika2014@mail.ru и указать свою Ф.И.О, группу, и название дисциплины тел 072-17-44-9-22

Тема: Расчет сложности алгоритмов сортировки.

Анализ различных методов сортировки

Анализ временной сложности – Мы обсудили наилучшую, среднюю и наихудшую сложность различных методов сортировки с возможными сценариями.

Сортировка на основе сравнения – При сортировке на основе сравнения элементы массива сравниваются друг с другом для нахождения отсортированного массива.

Сортировка пузырьками и сортировка вставками – Средняя и наихудшая временная сложность: n ^ 2В лучшем случае временная сложность: n, когда массив уже отсортирован. Наихудший случай: когда массив сортируется в обратном порядке.

Сортировка по выбору

Временная сложность наилучшего, среднего и наихудшего случаев: n ^ 2, которая не зависит от распределения данных.

Сортировка слиянием –

Наилучшая, средняя и наихудшая временная сложность: nlogn, который не зависит от распределения данных.

Сортировка по куче

Наилучшая, средняя и наихудшая временная сложность: nlogn, который не зависит от распределения данных

Быстрая сортировка

Это подход "разделяй и властвуй" с рекуррентным соотношением:

T(n) = T(k) + T(n-k-1) + cn

- Наихудший случай: когда массив сортируется в обратном порядке, алгоритм разделения делит массив на два подмассива с 0 и n-1 элементами. Поэтому,

T(n) = T(0) + T(n-1) + cn

Solving this we get, T(n) = O(n^2)

- Наилучший вариант и средний вариант: в среднем алгоритм разбиения массива на две подмассивы одинакового размера. Поэтому, T(n) = 2T(n/2) + cn

Solving this we get, T(n) = O(nlogn)

Сортировка, не основанная на сравнении – При сортировке без сравнения элементы массива не сравниваются друг с другом для нахождения отсортированного массива.

Сортировка по принципу Radix – Наилучшая, средняя и наихудшая временная сложность: nk, где k - максимальное количество цифр в элементах массива.

- **Сортировка по количеству** – Наилучшая, средняя и наихудшая временная сложность: n + k, где k - размер массива count.
- **Ковшовая сортировка** – Наилучшая и средняя трудоемкость по времени: n + k, где k - количество корзин.

Временная сложность в наихудшем случае: n ^ 2, если все элементы принадлежат одному и тому же сегменту.

Техника на месте / Outplace – Метод сортировки применяется, если он не использует дополнительную память для сортировки массива.

Среди рассмотренных методов, основанных на сравнении, только сортировка слиянием является устаревшим методом, поскольку для объединения отсортированных под массивами требуется дополнительный массив.

Среди рассмотренных методов, не основанных на сравнении, все они являются устаревшими методами. При под счётной сортировкой используется подсчитывающий массив, а при корзинной сортировке для сортировки массива используется хэш-таблица.

Онлайн / офлайн техника – Метод сортировки считается онлайновым, если он может принимать новые данные во время выполнения процедуры, т.е. для запуска операции сортировки не требуются полные данные.

Среди рассмотренных методов, основанных на сравнении, только сортировка по вставке подходит для этого из-за используемого базового алгоритма, то есть он обрабатывает массив (а не только элементы) слева направо, и если новые элементы добавляются справа, это не влияет на текущую операцию.

Стабильный / нестабильный метод – Метод сортировки стабилен, если он не изменяет порядок элементов с одинаковым значением.

Из методов, основанных на сравнении, пузырьковая сортировка, сортировка вставкой и сортировка слиянием являются стабильными методами. Сортировка по выбору нестабильна, поскольку может изменить порядок элементов с одинаковым значением. Для примера рассмотрим массив 4, 4, 1, 3.

На первой итерации минимальный найденный элемент равен 1, и он заменяется на 4 в 0-й позиции. Следовательно, порядок 4 по отношению к 4 на 1-й позиции изменится. Аналогичным образом, быстрая сортировка и сортировка по куче также нестабильны.

Из методов, не основанных на сравнении, под счётная сортировка и сортировка по сегментам являются стабильными методами сортировки, тогда как стабильность сортировки по основанию зависит от базового алгоритма, используемого для сортировки.

Анализ методов сортировки:

Когда массив почти отсортирован, сортировка по вставке может быть предпочтительной.

- Когда порядок ввода неизвестен, предпочтительна сортировка слиянием, поскольку она имеет наихудшую временную сложность по сравнению с nlogn, а также стабильна.
- Когда массив отсортирован, вставка и пузырьковая сортировка дают сложность n, но быстрая сортировка дает сложность n ^ 2.

Вопрос – 1. Какой алгоритм сортировки займет меньше всего времени, когда все элементы входного массива идентичны? Рассмотрим типичные реализации алгоритмов сортировки.

(A) Сортировка по вставке

(B) Сортировка по куче

(C) Сортировка слиянием

(D) Сортировка по выбору

Решение: как обсуждалось, сортировка по вставке будет иметь сложность n, когда входной массив уже отсортирован.

Вопрос – 2. Рассмотрим алгоритм быстрой сортировки. Предположим, что существует процедура поиска сводного элемента, которая разбивает список на два подсписка, каждый из которых содержит по меньшей мере одну пятую часть элементов. Пусть T (n) - количество сравнений, необходимых для сортировки n элементов. Затем (GATE-CS-2012)

(A) T (n) <= 2T (n/5) + n

(B) T (n) <= T(n/5) + T (4n/5) + n

(C) T (n) <= 2T (4n/5) + n

(D) T(n) <= 2T (n/2) + n

Решение: Сложность быстрой сортировки может быть записана как:

T(n) = T(k) + T(n-k-1) + cn

Как указано в вопросе, один список содержит 1/5 от общего количества элементов. Таким образом, в другом списке будет 4/5 от общего количества элементов. Складывая значения, мы получаем:

T(n) = T(n/5) + T (4n/5) + cn, что соответствует варианту (B).

Таблица сравнения сложности во времени и пространстве:

Алгоритм сортировки	**Временная сложность**			**Сложность пространства**
	Наилучший вариант	**Средний случай**	**Наихудший вариант**	**Наихудший вариант**
Пузырьковая сортировка	**$\Omega(N)$**	**$\Theta(N^2)$**	**$O(N^2)$**	**$O(1)$**
Сортировка по выбору	**$\Omega(N^2)$**	**$\Theta(N^2)$**	**$O(N^2)$**	**$O(1)$**
Сортировка по вставке	**$\Omega(N)$**	**$\Theta(N^2)$**	**$O(N^2)$**	**$O(1)$**
Сортировка слиянием	**$\Omega(N \log N)$**	**$\Theta(N \log N)$**	**$O(N \log N)$**	**$O(N)$**
Сортировка по куче	**$\Omega(N \log N)$**	**$\Theta(N \log N)$**	**$O(N \log N)$**	**$O(1)$**
Быстрая сортировка	**$\Omega(N \log N)$**	**$\Theta(N \log N)$**	**$O(N^2)$**	**O(логарифмическое число)**
Сортировка по принципу Radix	**$\Omega(N k)$**	**$\Theta(N k)$**	**$O(N k)$**	**$O(N + k)$**
Подсчитывать сортировку	**$\Omega(N + k)$**	**$\Theta(N + k)$**	**$O(N + k)$**	**$O(k)$**
Ковшовая сортировка	**$\Omega(N + k)$**	**$\Theta(N + k)$**	**$O(N^2)$**	**$O(N)$**