Отчёт по Bucket sort.

Санкт-Петербургский государственный университет
Академическая гимназия имени Д. К. Фаддеева

Шилинскас П.А. 10 м класс.

Общее понятие

Блочная сортировка (Карманная сортировка, корзинная сортировка, англ. Bucket sort) — алгоритм сортировки, в котором сортируемые элементы распределяются между конечным числом отдельных блоков (карманов, корзин) так, чтобы все элементы в каждом следующем по порядку блоке были всегда больше (или меньше), чем в предыдущем. Каждый блок затем сортируется отдельно, либо рекурсивно тем же методом, либо другим. Затем элементы помещаются обратно в массив. Этот тип сортировки может обладать линейным временем исполнения.

Алгоритм

Если входные элементы подчиняются равномерному закону распределения, то математическое ожидание времени работы алгоритма карманной сортировки является линейным. Это возможно благодаря определенным предположениям о входных данных. При карманной сортировке предполагается, что входные данные равномерно распределены на отрезке [0, 1).
Идея алгоритма заключается в том, чтобы разбить отрезок [0, 1) на n одинаковых отрезков (карманов), и разделить по этим карманам n входных величин. Поскольку входные числа равномерно распределены, предполагается, что в каждый карман попадет небольшое количество чисел. Затем последовательно сортируются числа в карманах. Отсортированный массив получается путём последовательного перечисления элементов каждого кармана.

Реализация на C++

```
void BucketSort(int* Array, int n){
  int k=0;
  int b=0;

  for(int i=0;i<n;i++){
    if(k<Array[i]){
      k=Array[i];
    };
  };

  k=(k/10)+1;
  int Joker[k][10];

  for(int i=0;i<k;i++){
    for(int j=0;j<10;j++){
      int s=0;
      for(int c=0;c<n;c++){
        if((Array[c]>i*10)&&(Array[c]<(i+1)*10)){
          Joker[i][j]=Array[c];
          Array[c]=-1;
          s++;
        };
      };
      if(s<11){
        Joker[i][s]=-1;
      };
    };
  };
  for(int i=0;i<k;i++){
    for(int j=0;(j<10)||(Joker[i][j]!=-1);j++){
      Array[b]=Joker[i][j];
      b++;
    };
  };
}
```

Оценка сложности

Оценим сложность алгоритма блочной сортировки для случая, при котором в качестве алгоритма сортировки блоков (*next-sort* из псевдокода) используется сортировка вставками.

Для оценки сложности алгоритма введём случайную величину n_i обозначающую количество элементов, которые попадут в карман `B[i]`. Время работы сортировки вставками равно $O\left(n^2\right)$.

Время работы алгоритма карманной сортировки равно

$$T(n) = \Theta(n) + \sum_{i=0}^{n-1} O(n_i^2)$$

Вычислим математическое ожидание обеих частей равенства:

$$M\left(T(n)\right) = M\left(\Theta(n) + \sum_{i=0}^{n-1} O(n_i^2)\right) = \Theta(n) + \sum_{i=0}^{n-1} O\left(M(n_i^2)\right)$$

Найдем величину $M(n_i^2)$.

Введем случайную величину X_{ij}, которая равна 1, если `A[j]` попадает в i-й карман, и 0 в противном случае:

$$n_i = \sum_{j=1}^{n} X_{ij}$$

$$M\left(n_i^2\right) = M\left[\left(\sum_{j=1}^{n} X_{ij}\right)^2\right] = M\left[\sum_{j=1}^{n}\sum_{k=1}^{n} X_{ij}X_{ik}\right] =$$

$$\sum_{j=1}^{n} M\left[X_{ij}^2\right] + \sum_{1\le j\le n}\ \sum_{1\le k\le n, k\ne j} M\left[X_{ij}X_{ik}\right]$$

$$M\left[X_{ij}^2\right] = 1\cdot\frac{1}{n} + 0\cdot\left(1-\frac{1}{n}\right) = \frac{1}{n}$$

Если $k \neq j$, величины X_{ij} и X_{ik} независимы, поэтому:

$$M\left[X_{ij}X_{ik}\right] = M\left[X_{ij}\right]M\left[X_{ik}\right] = \frac{1}{n^2}$$

Итак, ожидаемое время работы алгоритма карманной сортировки равно

$$\Theta(n) + n\cdot O(2 - 1/n) = \Theta(n)$$

Преимущества и недостатки

Преимущества: относится к классу быстрых алгоритмов с линейным временем исполнения O(N) (на удачных входных данных).

Недостатки: сильно деградирует при большом количестве мало отличных элементов, или же на неудачной функции получения номера корзины по содержимому элемента. В некоторых таких случаях для строк, возникающих в реализациях основанного на сортировке строк алгоритма сжатия BWT, оказывается, что быстрая сортировка строк в версии Седжвика значительно превосходит блочную сортировку скоростью.

Содержание:

Стр. 2 ---------- Общее понятие..
Стр. 2 ---------- Алгоритм.
Стр. 3 ---------- Реализация на C++.
Стр. 4 ---------- Оценка сложности.
Стр. 4 ---------- Преимущества и недостатки.
Стр. 5 ---------- Содержание.