算法基础第三次作业

PB20111686 黄瑞轩

1. 这之中稳定的排序：插入排序、归并排序、计数排序

   把原来的元素（假定 `int` 类型）看成一个 `pair<int, unsigned>`，第二个域表示其在原始数组中的位置。

   排序使用的 `compare` 函数变为：

   ```
   bool compare(pair<int, unsigned>& a, pair<int, unsigned>& b) {
       if (a.first == b.first) {
           return a.second < b.second;
       }
       return a.first < b.first;
   }
   ```

   用于比较的时间开销翻倍，空间开销因为多存放了一个域也翻倍。

2. 思路：开辟一个大小为 n 的 `std::vector<int>` 容器（有 n 个），每个容器 j 存放的是位数为 j 的数字。在每个容器中执行基数排序，然后按照容器顺序输出各容器中的排序后元素。

   时间复杂度分析：设长度为 j 的元素有 m_j 个，则 $\sum_{j=1}^{n} m_j = n$，在每个容器中执行基数排序，总的时间复杂度是

   $$\sum_{j=1}^{n} O(m_j) * j = \sum_{j=1}^{n} O(m_j) = O(\sum_{j=1}^{n} m_j) = O(n)$$

3. 优化算法如下：

   当 $i > n/2$ 时，就使用原来的 SELECT 算法；

   当 $i < n/2$ 时，

   做分组：$A[0...\lfloor \frac{n}{2} \rfloor], A[\lfloor \frac{n}{2} \rfloor ... 2\lfloor \frac{n}{2} \rfloor]$。

   在两个数组的相同位置进行比较，如果 $A[j]$ 比 $A[\lfloor \frac{n}{2} \rfloor + j]$ 大，则交换二者。

   对左侧分组递归执行上面两条操作，当不满足 $i < n'/2, n'$ 是当前分组元素个数时，停止递归，对当前分组使用原来的 SELECT 算法。

   > 注：
   >
   > 1. 如果 n 是奇数，则取出最后一个元素不参与分组，在下一次递归时加入。
   > 2. 在交换时，其上所有层的位置都要相应调换，比如 $A[0]$ 与 $A[j]$（假设在同一组）交换时，$A[\lfloor \frac{n}{2} \rfloor]$ 与 $A[\lfloor \frac{n}{2} \rfloor + j]$ 也要交换。

   时间复杂度分析如下：

   当 $i > n/2$ 时，$T_i(n) = T(n)$

   当 $i < n/2$ 时，$T_i(n) = \lfloor n/2 \rfloor + T_i(\lceil n/2 \rceil) + T(2i)$

   假设对较小的 n 有 $T_i(n) = n + cT(2i)\lg(n/i)$。

   当 $i < n/4$ 时，$T_i(n) \le n/2 + n/2 + cT(2i)\lg(n/2i) + T(2i)$（$c \ge 1$）

   否则，$T_i(n) \le n/2 + 2T(2i)$（$c \ge 2$）

   所以，选择 $c = 2$，有 $T_i(n) = n + O(T(2i)\lg(n/i))$

   当 i 是常数时，$T_i(n) = n + O(T(2i)\lg(n/i)) = n + O(\lg n)$