字母异位词分组

力扣官方题解 | + 关注
767728 2020.12.13 发布于 未知归属地

官方题解 哈希表 字符串 C++ Go 3+

前言

两个字符串互为字母异位词，当且仅当两个字符串包含的字母相同。同一组字母异位词中的字符串具备相同点，可以使用相同点作为一组字母异位词的标志，使用哈希表存储每一组字母异位词，哈希表的键为一组字母异位词的标志，哈希表的值为一组字母异位词列表。

遍历每个字符串，对于每个字符串，得到该字符串所在的一组字母异位词的标志，将当前字符串加入该组字母异位词的列表中。遍历全部字符串之后，哈希表中的每个键值对即为一组字母异位词。

以下的两种方法分别使用排序和计数作为哈希表的键。

方法一：排序

由于互为字母异位词的两个字符串包含的字母相同，因此对两个字符串分别进行排序之后得到的字符串一定是相同的，故可以将排序之后的字符串作为哈希表的键。

```
class Solution {
public:
    vector<vector<string>> groupAnagrams(vector<string>& strs) {
        unordered_map<string, vector<string>> mp;
        for (string& str: strs) {
            string key = str;
            sort(key.begin(), key.end());
            mp[key].emplace_back(str);
        }
        vector<vector<string>> ans;
        for (auto it = mp.begin(); it != mp.end(); ++it) {
            ans.emplace_back(it->second);
        }
        return ans;
    }
};
```

复杂度分析

- 时间复杂度：$O(nk \log k)$，其中 n 是 $strs$ 中的字符串的数量，k 是 $strs$ 中的字符串的的最大长度。需要遍历 n 个字符串，对于每个字符串，需要 $O(k \log k)$ 的时间进行排序以及 $O(1)$ 的时间更新哈希表，因此总时间复杂度是 $O(nk \log k)$。

- 空间复杂度：$O(nk)$，其中 n 是 $strs$ 中的字符串的数量，k 是 $strs$ 中的字符串的的最大长度。需要用哈希表存储全部字符串。

方法二：计数

由于互为字母异位词的两个字符串包含的字母相同，因此两个字符串中的相同字母出现的次数一定是相同的，故可以将每个字母出现的次数使用字符串表示，作为哈希表的键。

由于字符串只包含小写字母，因此对于每个字符串，可以使用长度为 26 的数组记录每个字母出现的次数。需要注意的是，在使用数组作为哈希表的键时，不同语言的支持程度不同，因此不同语言的实现方式也不同。

```
class Solution {
public:
    vector<vector<string>> groupAnagrams(vector<string>& strs) {
        // 自定义对 array<int, 26> 类型的哈希函数
        auto arrayHash = [fn = hash<int>{}] (const array<int, 26>& arr) -> size_t {
            return accumulate(arr.begin(), arr.end(), 0u, [&](size_t acc, int num) {
                return (acc << 1) ^ fn(num);
            });
        };

        unordered_map<array<int, 26>, vector<string>, decltype(arrayHash)> mp(0, arrayHash);
        for (string& str: strs) {
            array<int, 26> counts{};
            int length = str.length();
            for (int i = 0; i < length; ++i) {
                counts[str[i] - 'a'] ++;
            }
            mp[counts].emplace_back(str);
        }
        vector<vector<string>> ans;
        for (auto it = mp.begin(); it != mp.end(); ++it) {
            ans.emplace_back(it->second);
        }
        return ans;
```

```
    }
};
```

复杂度分析

- 时间复杂度：$O(n(k+|\Sigma|))$，其中 n 是 $strs$ 中的字符串的数量，k 是 $strs$ 中的字符串的的最大长度，Σ 是字符集，在本题中字符集为所有小写字母，$|\Sigma|=26$。需要遍历 n 个字符串，对于每个字符串，需要 $O(k)$ 的时间计算每个字母出现的次数，$O(|\Sigma|)$ 的时间生成哈希表的键，以及 $O(1)$ 的时间更新哈希表，因此总时间复杂度是 $O(n(k+|\Sigma|))$。

- 空间复杂度：$O(n(k+|\Sigma|))$，其中 n 是 $strs$ 中的字符串的数量，k 是 $strs$ 中的字符串的最大长度，Σ 是字符集，在本题中字符集为所有小写字母，$|\Sigma|=26$。需要用哈希表存储全部字符串，而记录每个字符串中每个字母出现次数的数组需要的空间为 $O(|\Sigma|)$，在渐进意义下小于 $O(n(k+|\Sigma|))$，可以忽略不计。