字符串

清华大学 茹逸中

目录

1. 单字符串匹配算法

问题：

给定母串 s 和模式串 p ，求 p 在 s 中可叠加地出现了多少次，并求出每次出现的位置。

例如 s = “bbababa” ， p=“baba” 。则 p 在 s 中出现 2 次，分别出现在 2 和 4 的位置。

暴力解法：

匹配 s[i] 与 p[j] ，若 s[i] = p[j] ，则 i++, j++ ，否则 i = i – j + 1, j = 0

- 效率低
- 复杂度为 O(nm)

1. 单字符串匹配算法

暴力算法的致命弱点在于在失配时 i 指针回退。

KMP 算法：

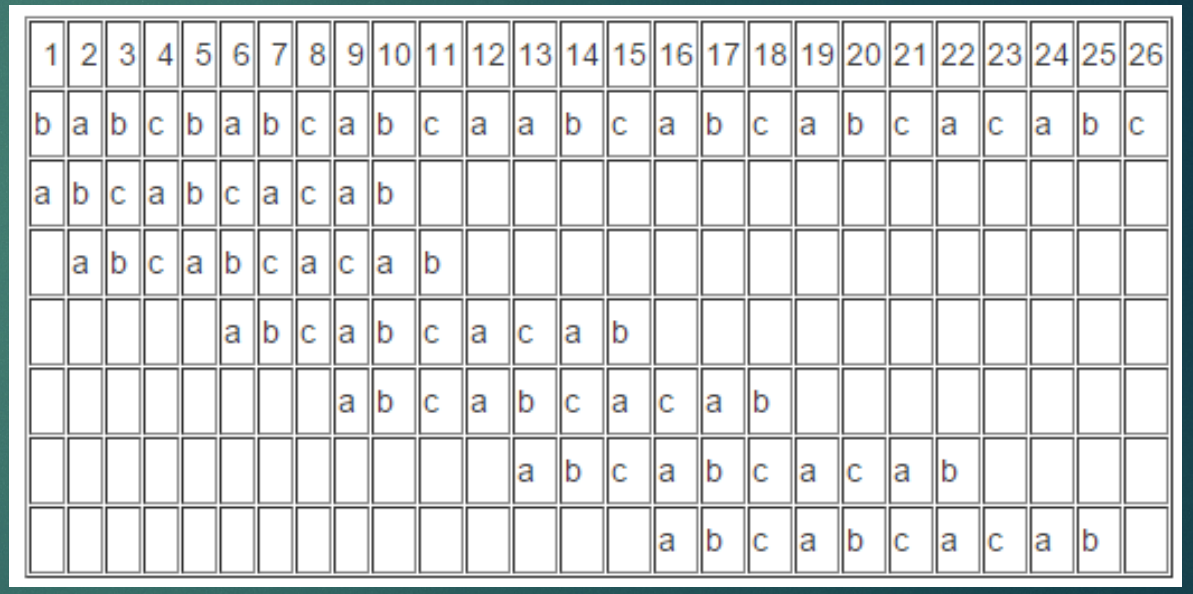
失配时 i 指针不回退，只改变 j 指针。

算法的重点是计算 next 数组。 Next 数组的用途是，当目标串 target 中的某个子部 target[m...m+(i-1)] 与 pattern 串的前 i 个字符 pattern[1...i] 相匹配时，如果 target[m+i] 与 pattern[i+1] 匹配失败，程序不会像朴素匹配算法那样，将 pattern[1] 与 target[m+1] 对其，然后由 target[m+1] 向后逐一进行匹配，而是会将模式串向后移动 i+1 - next[i+1] 个字符，使得 pattern[next[i+1]] 与 target[m+i] 对齐，然后再由 target[m+i] 向后与依次执行匹配。

1. 单字符串匹配算法

j	1	2	3	4	5	6	7	8	9	10
pattern[j]	a	b	c	a	b	c	a	c	a	b
next[j]	0	1	1	0	1	1	0	5	0	1

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
b	a	b	c	b	a	b	c	a	b	c	a	a	b	c	a	b	c	a	b	c	a	c	a	b	c
a	b	c	a	b	c	a	c	a	b																
	a	b	c	a	b	c	a	c	a	b															
					a	b	c	a	b	c	a	c	a	b											
								a	b	c	a	b	c	a	c	a	b								
												a	b	c	a	b	c	a	c	a	b				
															a	b	c	a	b	c	a	c	a	b	



1. 单字符串匹配算法

那如何计算 next 数组呢？

从左到右计算。

令 j = next[i-1]

若 p[j + 1] = p[i] 则 next[i] = j+1

否则 j = next[j]

1. 单字符串匹配算法

那如何计算 next 数组呢？

从左到右计算。

令 j = next[i-1]

若 p[j + 1] = p[i] 则 next[i] = j+1

否则 j = next[j]

1. 练习题

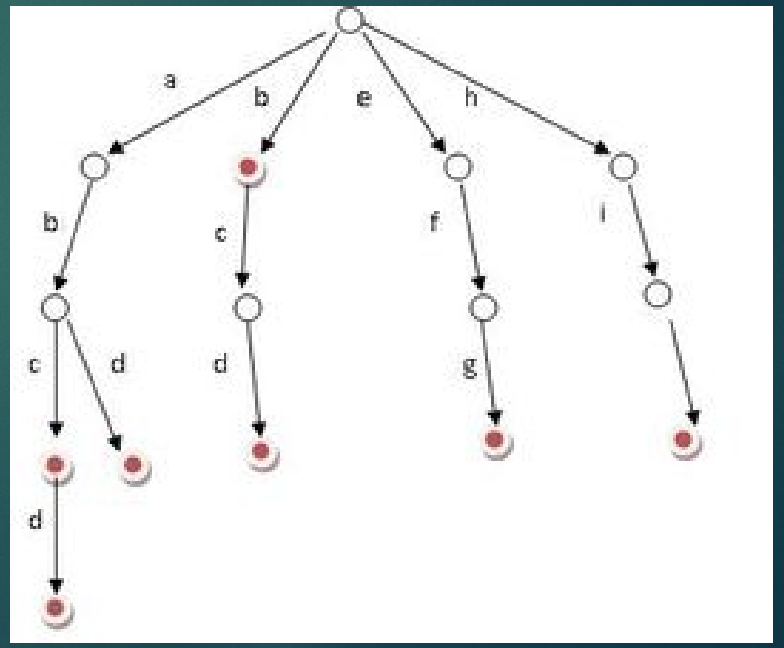
POJ3461

2.1 字母树 (trie)

Trie 树，是一种树形结构。典型应用是用于统计，排序和保存大量的字符串（但不仅限于字符串），所以经常被搜索引擎系统用于文本词频统计。它的优点是：利用字符串的公共前缀来减少查询时间，最大限度地减少无谓的字符串比较。

右图的 trie 树包含了以下几个单词：

abcd, abd, bcd, efg, hi,b,abc



2.1 字母树 (trie)

构建一棵 trie 树：

- 依次插入每一个单词。
- 从根节点开始，依次处理单词的每一个字母。
- 若当前节点的孩子中，没有标号为当前字母的孩子，则新建一个孩子，其标号为当前字母。
- 递归到标号为当前字母的孩子中，继续处理下一个字母。
- 整个单词结束后，在当前节点上打上标记，表示该单词存在。

在 trie 树上查询：

- 从根节点开始，依次处理每个字母。
- 若当前节点的孩子中，没有标号为当前字母的孩子，则返回不存在，否则递归到标号为当前字母的孩子中，继续处理下一个字母。
- 整个单词结束后，若当前节点有标记，则表示该单词存在，否则不存在。

2.1 例题 Shortest Prefixes(POJ2001)

给定一个单词集，对于每个单词，给出一个缩写，要求这个缩写是它本身的前缀但不是其它任何单词的前缀。若不存在，则输出它本身。

- 构建字母树
- 对于每个单词对应的结束节点，向上找若干层，直到找到有分叉的节点为止。

2.1 例题 最大异或值

给定 n 个数，任意选择两个数，使它们的异或值最大。

n <= 100000, 每个数 <2^31

- 将每个数转化成二进制，按位从高到低的顺序构建字母树
- 对于每个数，仍然按位从高到低查找，优先查找是否存在在某一位上与当前的数不同的数。

2.2 AC 自动机

问题：

给定一个母串 S ，和一个字典 D 。问字典 D 中的哪些单词在 S 中出现了，各出现多少次。

AC 自动机 =trie+KMP

AC 自动机的构造：

- 构造一棵 Trie ，作为 AC 自动机的搜索数据结构。
- 构造 fail 指针，使当前字符失配时跳转到具有最长公共前后缀的字符继续匹配。如同 KMP 算法一样， AC 自动机在匹配时如果当前字符匹配失败，那么利用 fail 指针进行跳转。由此可知如果跳转，跳转后的串的前缀，必为跳转前的模式串的后缀并且跳转的新位置的深度（匹配字符个数）一定小于跳之前的节点。所以我们可以利用 bfs 在 Trie 上面进行 fail 指针的求解。

2.2 AC 自动机

计算 fail 指针：

假设当前节点的字符标记为 a 。

```
v = u-f->fail
while (v->c[a] == NULL) v = v->fail
u->fail = v->c[a]
```

2.2 例题 DNA repair(POJ 3691)

给定一个母串和一些病毒串，要求修改母串上的最少的字符使其不出现任何病毒串。

- 动态规划时如何确定状态？
- 对病毒串建立 AC 自动机， AC 自动机上的状态作为动态规划的状态！
- 只要到达了 AC 自动机上有结束标记的节点就说明母串中出现了病毒串，是不可行的

3.1 后缀数组的概念

在字符串处理中，后缀数组是一种非常有用的工具。它是将一个字符串的所有后缀按字典序大小排序，然后将排序后的结果存在一个数组里，这个数组就是后缀数组。

例如，对于字符串 abcaaaab ，有 8 个后缀，分别是

abcaaaab,bcaaaab,caaaab,aaaab,aaab,aab,ab,b

将这些后缀按字典序排序，得到后缀数组

aaaab

aaab

aab

ab

abcaaaab

b

bcaaab

caaaab

3.1 后缀数组的概念

在字符串处理中，后缀数组是一种非常有用的工具。它是将一个字符串的所有后缀按字典序大小排序，然后将排序后的结果存在一个数组里，这个数组就是后缀数组。

例如，对于字符串 abcaaaab ，有 8 个后缀，分别是

abcaaaab,bcaaaab,caaaab,aaaab,aaab,aab,ab,b

将这些后缀按字典序排序，得到后缀数组

aaaab

aaab

aab

ab

abcaaaab

b

bcaaab

caaaab

3.2 倍增算法

如何求得后缀数组呢？

暴力排序复杂度 O(n^2 log n)。

一个效率较高的且容易理解并在考场实现的算法：倍增算法。

倍增算法的基础仍然是比较各后缀，但它的关键思想是分次比较。第一次比较各后缀的前 1 个字符并排序，第二次比较各后缀的前 2 个字符并排序，第三次比较各后缀的前 4 个字符并排序...

倍增算法将每次比较前 2^i 个字符并排序的过程的复杂度做到 O(n) ，因此倍增算法的总复杂度是 O(n log n)。

3.2 倍增算法

引入 rank 数组， rank[i] 表示以原串中第 i 个字符为起始的后缀在 sa 中的排名。

每次排序时都要生成 rank 数组。最后一次排序得到的 rank 数组应该是互不相同的，但是对第 k 次排序，前 2^k 个字符相同的后缀的 rank 值应相等。

对于第 k(k > 0) 次排序，已知第 k-1 次排序的结果，即已知每个后缀前 2^(k-1) 个字符排序后的结果。这时可以进行位数为 2 的基数排序，具体排序流程如下：

建立 m 个桶， m 为第 k-1 次 rank 数组中的最大值。对于每个后缀 i ，按照其后半部分在 i-1 次后缀数组中的顺序，加入 rank[i] 的桶中。然后从小到大扫描每个桶，将这个桶中的后缀依次拿出，即得到第 k 次的后缀数组。根据后缀数组容易算得 rank 数组。

3.3 height 数组

在后缀数组中，还有一个重要的概念，叫 height 数组。 height 数组中第 i 个数表示后缀数组中第 i 个后缀和第 i+1 个后缀的最长公共前缀。例如

aaaab 3

aaab 2

aab 1

ab 1

abcaaaab 0

b 1

bcaaab 0

caaaab

3.3 height 数组

如何求 height 数组？

暴力？

需要利用一个性质：在母串中开始位置为 i 的后缀相应的 height 值至少是开始位置为 i-1 的后缀相应的 height 值 -1 。

总复杂度 O(n)

3.4 典型例题

- 给定一个字符串，询问某两个后缀的最长公共前缀。
- height 数组 +rmq

3.4 典型例题

- 给定一个字符串，求可重叠的 k 次最长子串
- height 数组 + 二分 + 并查集

3.4 典型例题

- 求一个字符串不同子串的个数
- 求 height 数组
- 若某个后缀长度为 l ， height 值为 h ，则在这个后缀中与之前统计过的子串都不同的子串有 l-h 个
- 累加答案

3.4 典型例题

- 有一个字符串 S(初始为空)，你的程序需要支持以下操作：
  - 在 S 在后面加上字符串 a
  - 询问字符串 a 在 S 中出现的次数。
  - 将字符串 S 整个翻转
  - 可离线
- 先离线预处理出最终的字符串，问题就转化为计算询问串 a 在母串 S[l,r] 这一子串中出现的次数
- 令 f(a,l,r)= 询问串 a 在母串 S[l,r] 这一子串中出现的次数，显然有以下式子成立： f(a,l,r)=f(a,l,n)-f(a,r-len(a)+1,n) 。
- 求 S 的后缀数组，字符串 a 在后缀数组中一定匹配到连续的一段 [x,y] 。
- 要询问 f(a,l,n) ，只要知道在后 (n-l) 个后缀中有多少个在后缀数组中处于 [x,y] 位置就可以了。
- 在线可持久化线段树。
- 离线树状数组。

谢谢！

字符串

清华大学 茹逸中

目录

1. 单字符串匹配算法

问题：

给定母串 s 和模式串 p ，求 p 在 s 中可叠加地出现了多少次，并求出每次出现的位置。

例如 s = “bbababa” ， p=“baba” 。则 p 在 s 中出现 2 次，分别出现在 2 和 4 的位置。

暴力解法：

匹配 s[i] 与 p[j] ，若 s[i] = p[j] ，则 i++, j++ ，否则 i = i – j + 1, j = 0

- 效率低
- 复杂度为 O(nm)

1. 单字符串匹配算法

暴力算法的致命弱点在于在失配时 i 指针回退。

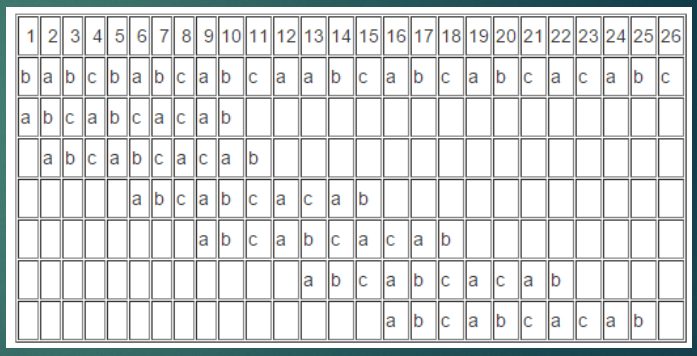
KMP 算法：

失配时 i 指针不回退，只改变 j 指针。

算法的重点是计算 next 数组。 Next 数组的用途是，当目标串 target 中的某个子部 target[m...m+(i-1)] 与 pattern 串的前 i 个字符 pattern[1...i] 相匹配时，如果 target[m+i] 与 pattern[i+1] 匹配失败，程序不会像朴素匹配算法那样，将 pattern[1] 与 target[m+1] 对其，然后由 target[m+1] 向后逐一进行匹配，而是会将模式串向后移动 i+1 - next[i+1] 个字符，使得 pattern[next[i+1]] 与 target[m+i] 对齐，然后再由 target[m+i] 向后与依次执行匹配。

1. 单字符串匹配算法

j	1	2	3	4	5	6	7	8	9	10
pattern[j]	a	b	c	a	b	c	a	c	a	b
next[j]	0	1	1	0	1	1	0	5	0	1



1. 单字符串匹配算法

那如何计算 next 数组呢？

从左到右计算。

令 j = next[i-1]

若 p[j + 1] = p[i] 则 next[i] = j+1

否则 j = next[j]

1. 单字符串匹配算法

那如何计算 next 数组呢？

从左到右计算。

令 j = next[i-1]

若 p[j + 1] = p[i] 则 next[i] = j+1

否则 j = next[j]

1. 练习题

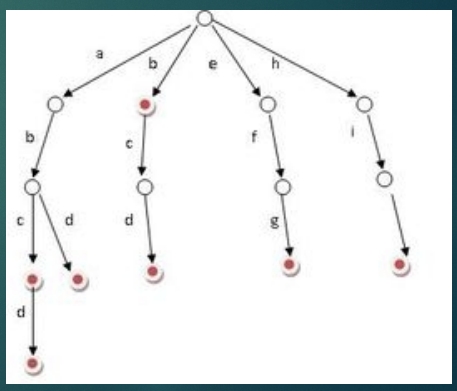
POJ3461

2.1 字母树 (trie)

Trie 树，是一种树形结构。典型应用是用于统计，排序和保存大量的字符串（但不仅限于字符串），所以经常被搜索引擎系统用于文本词频统计。它的优点是：利用字符串的公共前缀来减少查询时间，最大限度地减少无谓的字符串比较。

右图的 trie 树包含了以下几个单词：

abcd, abd, bcd, efg, hi,b,abc



2.1 字母树 (trie)

构建一棵 trie 树：

- 依次插入每一个单词。
- 从根节点开始，依次处理单词的每一个字母。
- 若当前节点的孩子中，没有标号为当前字母的孩子，则新建一个孩子，其标号为当前字母。
- 递归到标号为当前字母的孩子中，继续处理下一个字母。
- 整个单词结束后，在当前节点上打上标记，表示该单词存在。

在 trie 树上查询：

- 从根节点开始，依次处理每个字母。
- 若当前节点的孩子中，没有标号为当前字母的孩子，则返回不存在，否则递归到标号为当前字母的孩子中，继续处理下一个字母。
- 整个单词结束后，若当前节点有标记，则表示该单词存在，否则不存在。

2.1 例题 Shortest Prefixes(POJ2001)

给定一个单词集，对于每个单词，给出一个缩写，要求这个缩写是它本身的前缀但不是其它任何单词的前缀。若不存在，则输出它本身。

- 构建字母树
- 对于每个单词对应的结束节点，向上找若干层，直到找到有分叉的节点为止。

2.1 例题 最大异或值

给定 n 个数，任意选择两个数，使它们的异或值最大。

n <= 100000，每个数 <2^31

- 将每个数转化成二进制，按位从高到低的顺序构建字母树
- 对于每个数，仍然按位从高到低查找，优先查找是否存在在某一位上与当前的数不同的数。

2.2 AC 自动机

问题：

给定一个母串 S ，和一个字典 D 。问字典 D 中的哪些单词在 S 中出现了，各出现多少次。

AC 自动机 =trie+KMP

AC 自动机的构造：

- 构造一棵 Trie ，作为 AC 自动机的搜索数据结构。
- 构造 fail 指针，使当前字符失配时跳转到具有最长公共前后缀的字符继续匹配。如同 KMP 算法一样， AC 自动机在匹配时如果当前字符匹配失败，那么利用 fail 指针进行跳转。由此可知如果跳转，跳转后的串的前缀，必为跳转前的模式串的后缀并且跳转的新位置的深度（匹配字符个数）一定小于跳之前的节点。所以我们可以利用 bfs 在 Trie 上面进行 fail 指针的求解。

2.2 AC 自动机

计算 fail 指针：

假设当前节点的字符标记为 a 。

```
v = u-f->fail
while (v->c[a] == NULL) v = v->fail
u->fail = v->c[a]
```

2.2 例题 DNA repair(POJ 3691)

给定一个母串和一些病毒串，要求修改母串上的最少的字符使其不出现任何病毒串。

- 动态规划时如何确定状态？
- 对病毒串建立 AC 自动机， AC 自动机上的状态作为动态规划的状态！
- 只要到达了 AC 自动机上有结束标记的节点就说明母串中出现了病毒串，是不可行的

3.1 后缀数组的概念

在字符串处理中，后缀数组是一种非常有用的工具。它是将一个字符串的所有后缀按字典序大小排序，然后将排序后的结果存在一个数组里，这个数组就是后缀数组。

例如，对于字符串 abcaaaab ，有 8 个后缀，分别是

abcaaaab,bcaaaab,caaaab,aaaab,aaab,aab,ab,b

将这些后缀按字典序排序，得到后缀数组

aaaab

aaab

aab

ab

abcaaaab

b

bcaaab

caaaab

3.1 后缀数组的概念

在字符串处理中，后缀数组是一种非常有用的工具。它是将一个字符串的所有后缀按字典序大小排序，然后将排序后的结果存在一个数组里，这个数组就是后缀数组。

例如，对于字符串 abcaaaab ，有 8 个后缀，分别是

abcaaaab,bcaaaab,caaaab,aaaab,aaab,aab,ab,b

将这些后缀按字典序排序，得到后缀数组

aaaab

aaab

aab

ab

abcaaaab

b

bcaaab

caaaab

3.2 倍增算法

如何求得后缀数组呢？

暴力排序复杂度 O(n^2 log n)。

一个效率较高的且容易理解并在考场实现的算法：倍增算法。

倍增算法的基础仍然是比较各后缀，但它的关键思想是分次比较。第一次比较各后缀的前 1 个字符并排序，第二次比较各后缀的前 2 个字符并排序，第三次比较各后缀的前 4 个字符并排序...

倍增算法将每次比较前 2^i 个字符并排序的过程的复杂度做到 O(n) ，因此倍增算法的总复杂度是 O(n log n)。

3.2 倍增算法

引入 rank 数组， rank[i] 表示以原串中第 i 个字符为起始的后缀在 sa 中的排名。

每次排序时都要生成 rank 数组。最后一次排序得到的 rank 数组应该是互不相同的，但是对第 k 次排序，前 2^k 个字符相同的后缀的 rank 值应相等。

对于第 k(k > 0) 次排序，已知第 k-1 次排序的结果，即已知每个后缀前 2^(k-1) 个字符排序后的结果。这时可以进行位数为 2 的基数排序，具体排序流程如下：

建立 m 个桶， m 为第 k-1 次 rank 数组中的最大值。对于每个后缀 i ，按照其后半部分在 i-1 次后缀数组中的顺序，加入 rank[i] 的桶中。然后从小到大扫描每个桶，将这个桶中的后缀依次拿出，即得到第 k 次的后缀数组。根据后缀数组容易算得 rank 数组。

3.3 height 数组

在后缀数组中，还有一个重要的概念，叫 height 数组。 height 数组中第 i 个数表示后缀数组中第 i 个后缀和第 i+1 个后缀的最长公共前缀。例如

aaaab 3

aaab 2

aab 1

ab 1

abcaaaab 0

b 1

bcaaab 0

caaaab

3.3 height 数组

如何求 height 数组？

暴力？

需要利用一个性质：在母串中开始位置为 i 的后缀相应的 height 值至少是开始位置为 i-1 的后缀相应的 height 值 -1 。

总复杂度 O(n)

3.4 典型例题

- 给定一个字符串，询问某两个后缀的最长公共前缀。
- height 数组 +rmq

3.4 典型例题

- 给定一个字符串，求可重叠的 k 次最长子串
- height 数组 + 二分 + 并查集

3.4 典型例题

- 求一个字符串不同子串的个数
- 求 height 数组
- 若某个后缀长度为 l ， height 值为 h ，则在这个后缀中与之前统计过的子串都不同的子串有 l-h 个
- 累加答案

3.4 典型例题

- 有一个字符串 S(初始为空)，你的程序需要支持以下操作：
  - 在 S 在后面加上字符串 a
  - 询问字符串 a 在 S 中出现的次数。
  - 将字符串 S 整个翻转
  - 可离线
- 先离线预处理出最终的字符串，问题就转化为计算询问串 a 在母串 S[l,r] 这一子串中出现的次数
- 令 f(a,l,r)= 询问串 a 在母串 S[l,r] 这一子串中出现的次数，显然有以下式子成立： f(a,l,r)=f(a,l,n)-f(a,r-len(a)+1,n) 。
- 求 S 的后缀数组，字符串 a 在后缀数组中一定匹配到连续的一段 [x,y] 。
- 要询问 f(a,l,n) ，只要知道在后 (n-l) 个后缀中有多少个在后缀数组中处于 [x,y] 位置就可以了。
- 在线可持久化线段树。
- 离线树状数组。

谢谢！