

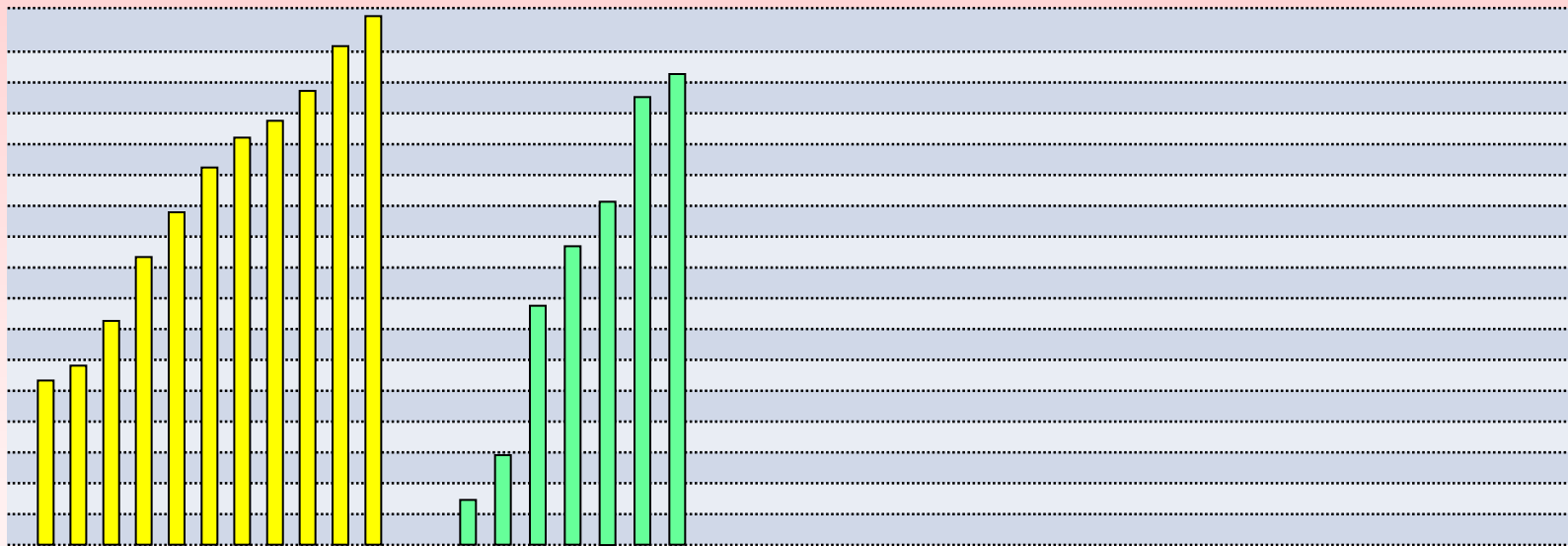


本节主题：

归并排序

归并排序思想

- ☐ 多次将两个或两个以上的有序表合并成一个新的有序表。
- ☐ 最简单的归并：是直接将两个有序的子表合并成一个有序的表。



2-路归并的实现

有序子序列 $R[low..mid]$

有序子序列 $R[mim+1..high]$

有序序列 $R1[0..high-low]$

有序序列 $R[low..high]$

```
void Merge(RecType R[],int low,int mid,int high)
```

```
{
```

```
    RecType *R1;
```

```
    int i=low,j=mid+1,k=0;
```

```
    //动态分配空间R1，用于保存合并结果
```

```
    R1=(RecType *)malloc((high-low+1)*sizeof(RecType));
```

```
    //两段均未扫描完时合并
```

```
    //将第1段余下的部分复制到R1
```

```
    //将第2段余下的部分复制到R1
```

```
    //将合并后的结果复制回R
```

```
    for (k=0,i=low; i<=high; k++,i++)  
        R[i]=R1[k];
```

```
    while (j<=high)  
    {  
        R1[k]=R[j];  
        j++;  
        k++;  
    }
```

```
    while (i<=mid)  
    {  
        R1[k]=R[i];  
        i++;  
        k++;  
    }
```

空间复杂度为
 $O(high-low+1)$

```
while (i<=mid && j<=high)  
    if (R[i].key<=R[j].key)  
    {  
        R1[k]=R[i];  
        i++;  
        k++;  
    }  
    else  
    {  
        R1[k]=R[j];  
        j++;  
        k++;  
    }
```

归并排序的过程

初始: 18 2 20 34 12 32 6 16 1 5

第1趟 2 18 20 34 12 32 6 16 1 5

第2趟 2 18 20 34 6 12 16 32 1 5

第3趟 2 6 12 16 18 20 32 34 1 5

第4趟 1 2 5 6 12 16 18 20 32 34

共4趟, $\log_2 10 = 4$

实现一趟归并MergePass()及合并排序MergeSort()的实现

```
void MergePass(RecType R[],int length,int n)
```

```
{  
    int i;  
    for (i=0; i+2*length-1<n; i=i+2*length)  
        Merge(R,i,i+length-1,i+2*length-1);  
    if (i+length-1<n)  
        Merge(R,i,i+length-1,n-1);  
}
```

```
void MergeSort(RecType R[],int n)
```

```
{  
    int length;  
    for (length=1; length<n; length=2*length)  
        MergePass(R,length,n);  
}
```

初始:

18 2 20 34 12 32 6 16 1 5

第1趟

2 18 20 34 12 32 6 16 1 5

第2趟

2 18 20 34 6 12 16 32 1 5

第3趟

2 6 12 16 18 20 32 34 1 5

第4趟

1 2 5 6 12 16 18 20 32 34

进行 $\log_2 n$
趟归并

有序子序列

有序子序列

length

length

性能及应用

时间复杂度

- 每一趟归并的时间复杂度为 $O(n)$
- 总共需进行 $\lceil \log_2 n \rceil$ 趟。
- 归并排序的时间复杂度为 $O(n \log_2 n)$

空间复杂度

- $O(n)$

实际应用

- 当要合并的段很短时，直接用插入排序等简单算法

初始:	<u>18</u>	<u>2</u>	<u>20</u>	<u>34</u>	<u>12</u>	<u>32</u>	<u>6</u>	<u>16</u>	<u>1</u>	<u>5</u>
第1趟	2	18	20	34	12	32	6	16	1	5
第2趟	2	18	20	34	6	12	16	32	1	5
第3趟	2	6	12	16	18	20	32	34	1	5
第4趟	1	2	5	6	12	16	18	20	32	34