有序表的合并

```
Status List_Merge (SqListPtr La, SqListPtr Lb, SqListPtr Lc){
    ElemType elem1, elem2;  status = List_Init(Lc);
    int i=1, j=1, k=1;    /*i, j, k分别用于指示La, Lb, Lc中当前元素*/
    int n = List_Size(La),  m = List_Size(Lb);
    while(i<=n && j<=m){              /*两个表都还未处理完*/
        List_Retrieve(La,i,&elem1); List_Retrieve(Lb,j,&elem2);
        if(elem1<elem2){ status = List_Insert(Lc,k,elem1);  i=i+1;  }
        else{ status = List_Insert(Lc,k,elem2); j=j+1;  }
         k=k+1;
   }
```

```
Status List_Merge (SqListPtr La, SqListPtr Lb, SqListPtr Lc){
    .......
    while(i<=n){                /*表La都还未处理完*/
        List_Retrieve(La, i, &elem1);
        status = List_Insert(Lc, k, elem1);
        i=i+1; k=k+1;    }
    while(j<=m){                /*表Lb都还未处理完*/
        List_Retrieve(Lb, j, &elem2);
        status = List_Insert(Lc, k, elem2);
        j=j+1; k=k+1;    }
    return status;
}
```

性能分析

- 顺序存储结构：
  - List_Size: O(1)
  - List_Retrieve: O(1)
  - List_Insert: O(1)---插入到线性表C的尾部

顺序存储结构效率 最优！

循环n次

- 链式存储结构：
  - List_Size: O(n)
  - List_Retrieve: O(n)
  - List_Insert： O(1)---插入到线性表C的头部

循环n次

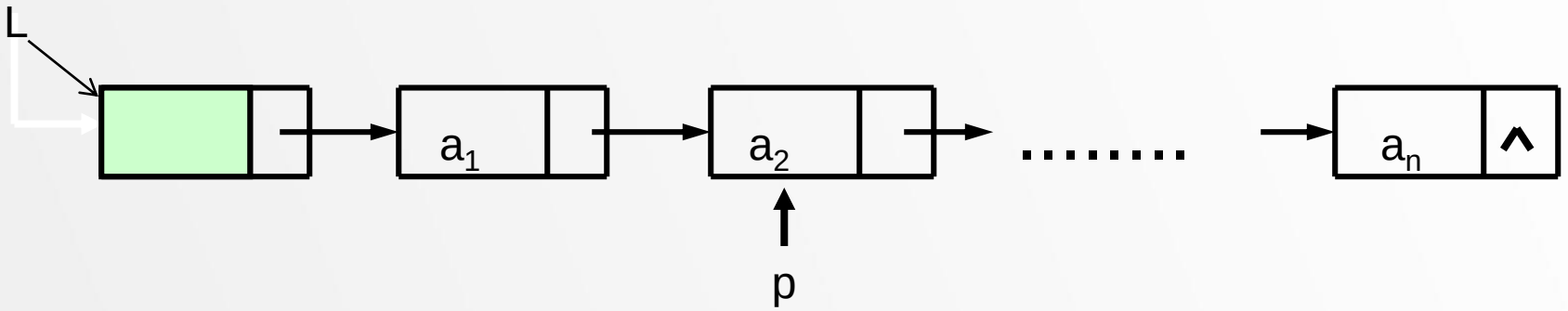
性能分析

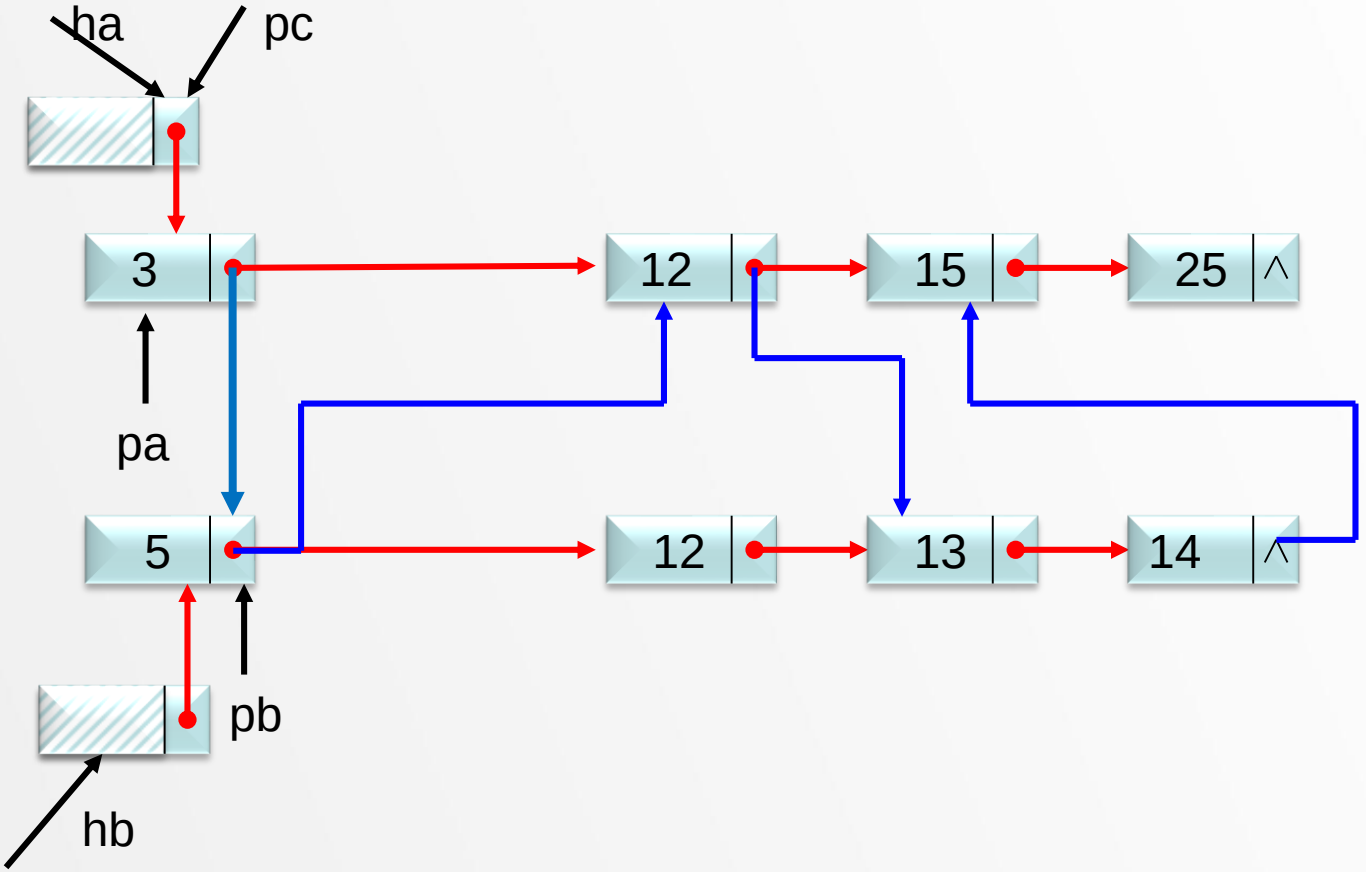
- 顺序存储结构：
  - List_Size: O(1)
  - List_Retrieve: O(1)
  - List_Insert: O(1)---插入到线性表C的尾部

循环n次

- 链式存储结构：
  - List_Size: O(n)
  - List_Retrieve: O(1)
  - List_Insert： O(1)---插入到线性表C的头部

循环n次


L
a1
a2
an
^
p


ha
pc
3
12
15
25
pa
5
12
13
14
pb
hb

性能分析

链式存储结构效率更高

- 顺序存储结构：
  - List_Size: O(1)
  - List_Retrieve: O(1)
  - List_Insert: O(1)---插入到线性表C的尾部

循环n次

- 链式存储结构：
  - List_Size: O(n)
  - List_Retrieve: O(1)
  - List_Insert： O(1)---插入到线性表C的头部

循环n次

归并2个有序表 | 链式储存：

```
void  merge_linklist(ListPtr *la, ListPtr *lb, ListPtr *lc)  {    //la和lb为参与归并的两个有序表,lc为结果有序表
    ListPtr pa,pb,pc,tp;
     pa=(*la)->next;  pb=(*lb)->next;  pc=*la;
     while( pa  && pb){
            if (pa->data < pb->data){
                    pc->next=pa;  pc=pa;  pa=pa->next;
            } else if(pa->data>pb->data){
                    pc->next=pb;  pc=pb;  pb=pb->next; }
                        else{pc->next=pa;pc=pa;pa=pa->next;
                      tp=pb;pb=pb->next;free(tp);tp=NULL;}
       }
       pc->next = (pa?pa:pb) ;     //插入剩余段
       free(lb);
       lc=la;
   }//mer-linklist
```

问题：如果不破坏la、lb，怎样改写该算法?

顺序表实现有序表的合并