養天地正氣 法古今完人

队列的链式实现

链式队列

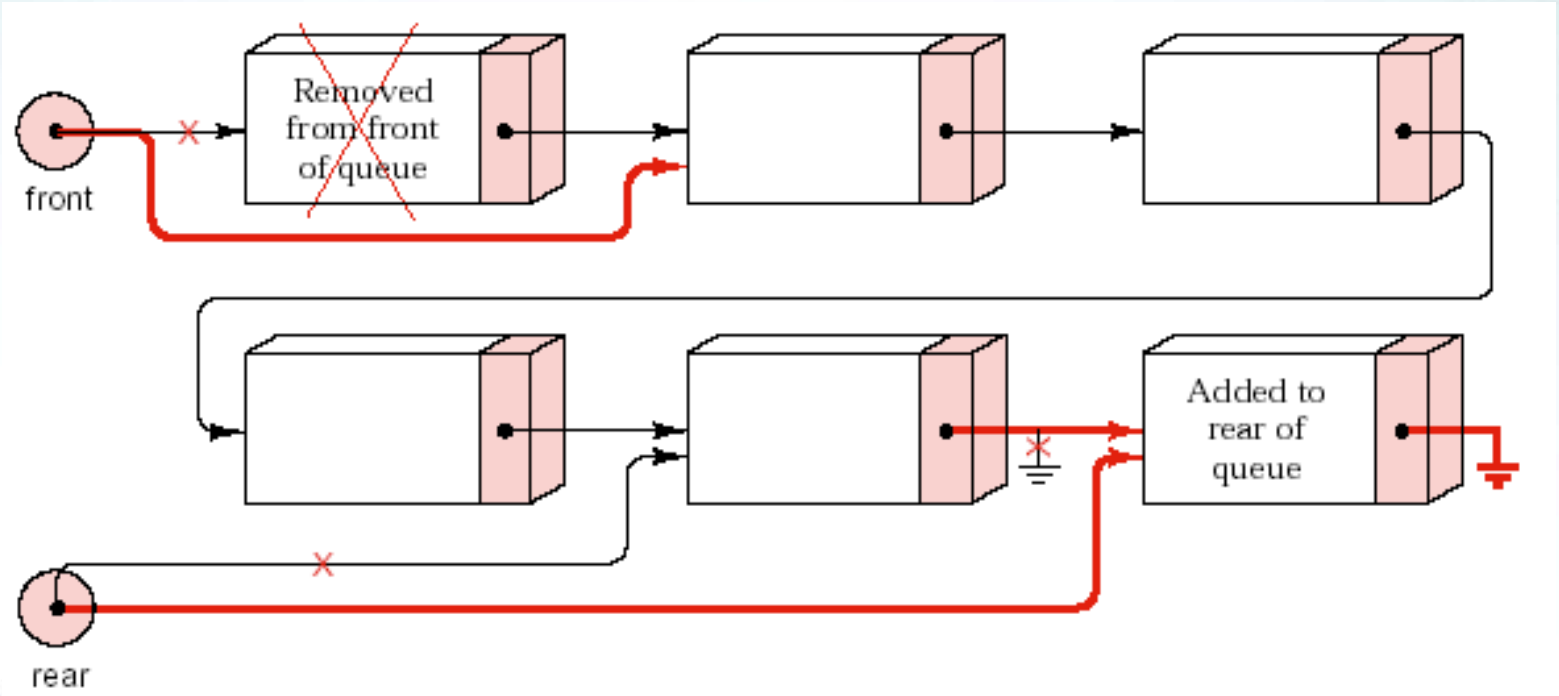

Figure 4.13. Operations on a linked queue

链式队列

```
class Queue {
    public:
    // standard Queue methods
        Queue( );
        bool empty( ) const;
        Error_code append(const Queue_entry &item);
        Error_code serve( );
        Error_code retrieve(Queue_entry &item) const;
    // safety features for linked structures
        ~Queue( );
        Queue(const Queue &original);
        void operator = (const Queue &original);
    protected:
        Node *front, *rear;
};
```

链式队列

- 初始化：
  - 空队，无任何数据元素，front和rear均为NULL

```
Queue :: Queue( )
/* Post: The Queue is initialized to be empty. */
{
    front = rear = NULL;
}
```

链式队列

- 入队

```
Error_code Queue :: append(const Queue_entry &item)
/* Post: Add item to the rear of the Queue and return a code of success
or return a code of overflow if dynamic memory is exhausted. */ {
    Node *new_rear = new Node(item);
    if (new_rear == NULL) return overflow;
    //原为空队，新入队元素即是队头，也是队尾
    if (rear == NULL) front = rear = new_rear;
    else {//原为非空队，只需修改rear，与front无关
        rear->next = new_rear;
        rear = new_rear;
    }
    return success;
}
```

链式队列

- 出队

```
Error_code Queue :: serve( )
/* Post: The front of the Queue is removed. If the Queue
is empty, return an Error_code of underflow. */ {
    if (front == NULL) return underflow;
    Node *old_front = front; //保存好原来的队头结点
    front = old_front->next;
    //原队列只有一个元素，出队后，rear也需要修改
    if (front == NULL) rear = NULL;
    delete old_front;          //归还原队头结点
    return success;
}
```

链式队列

- 扩展的链式队列

```
class Extended_queue: public Queue {
public:
    bool full( ) const;
    int size( ) const;
    void clear( );
    Error_code serve_and_retrieve(Queue_entry &item);
};
```

注：安全性的修正（析构函数、赋值运算符重载、拷贝构造函数）以继承的方式获得，无需再显式定义。

链式队列

```
void  Extended_queue :: clear( ) {
        Node *window = front;
        while (window != NULL) {
                front=front->next;
                delete window;
                window =front;
                }
}
```

或者

```
void Extended_Queue :: clear(){
        while (!empty())
                serve();
}
```