delete 可以通过结合 decreaseKey 和 deleteMin 而以时间 $O(\log N)$完成。

```java
    /**
     * Merge rhs into the priority queue.
     * rhs becomes empty. rhs must be different from this.
     * @param rhs the other binomial queue.
     */
    public void merge( BinomialQueue<AnyType> rhs )
    {
        if( this == rhs )    // Avoid aliasing problems
            return;

        currentSize += rhs.currentSize;

        if( currentSize > capacity( ) )
        {
            int maxLength = Math.max( theTrees.length, rhs.theTrees.length );
            expandTheTrees( maxLength + 1 );
        }

        Node<AnyType> carry = null;
        for( int i = 0, j = 1; j <= currentSize; i++, j *= 2 )
        {
            Node<AnyType> t1 = theTrees[ i ];
            Node<AnyType> t2 = i < rhs.theTrees.length ? rhs.theTrees[ i ] : null;

            int whichCase = t1 == null ? 0 : 1;
            whichCase += t2 == null ? 0 : 2;
            whichCase += carry == null ? 0 : 4;

            switch( whichCase )
            {
              case 0: /* No trees */
              case 1: /* Only this */
                break;
              case 2: /* Only rhs */
                theTrees[ i ] = t2;
                rhs.theTrees[ i ] = null;
                break;
              case 4: /* Only carry */
                theTrees[ i ] = carry;
                carry = null;
                break;
              case 3: /* this and rhs */
                carry = combineTrees( t1, t2 );
                theTrees[ i ] = rhs.theTrees[ i ] = null;
                break;
              case 5: /* this and carry */
                carry = combineTrees( t1, carry );
```

图 6-55　合并两个优先队列的例程