词典

跳转表：插入与删除

11-G3

如果一个人遇到不可解之事，把脑子想穿了，也找不到其中的原因，怎么办呢？
他或许会去庙里烧香，把自己的难题交给算命先生，听任他们的摆布。

邓俊辉
deng@tsinghua.edu.cn

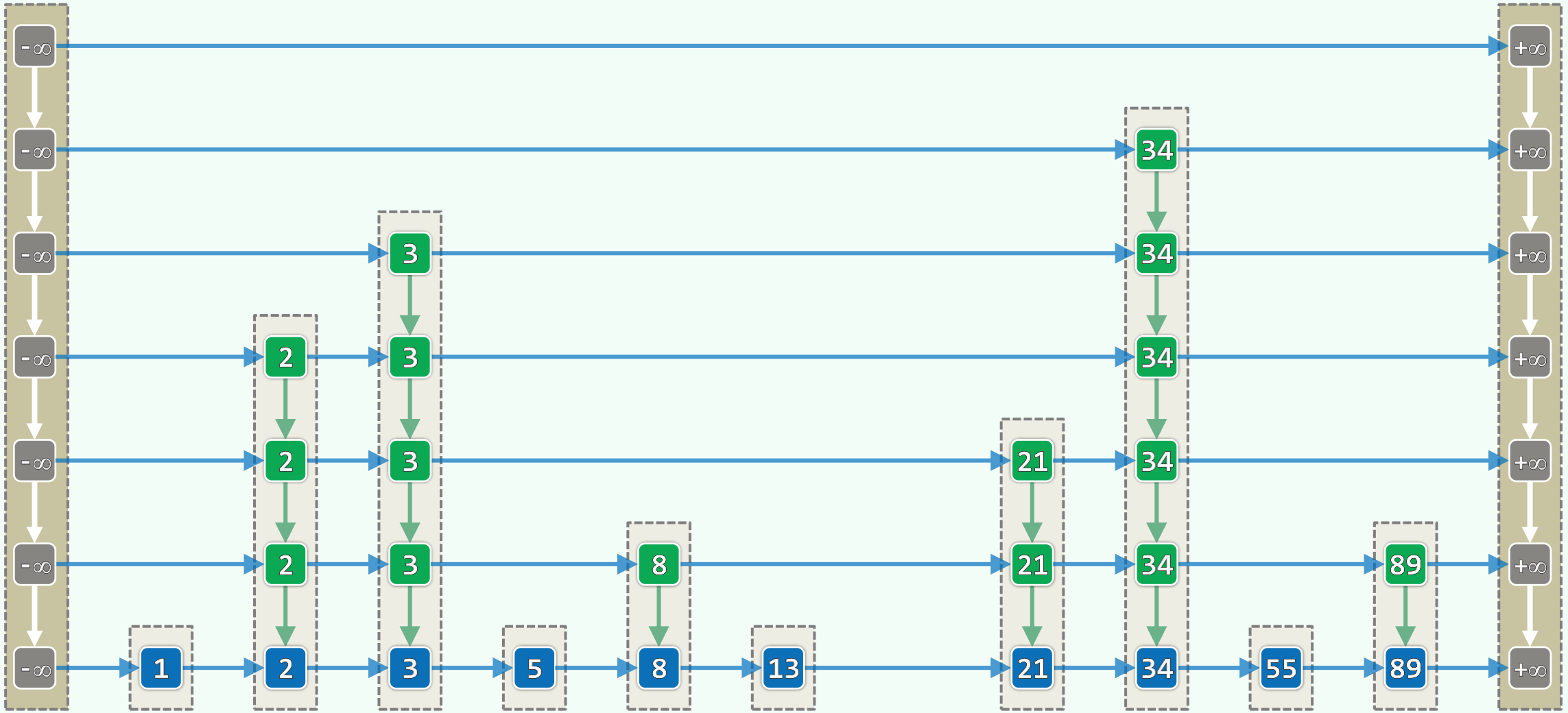
插入算法（1/2）

```
template <typename K, typename V> bool Skiplist< K, V >::put( K k, V v ) {
   Entry< K, V > e = Entry< K, V >( k, v ); //将被随机地插入多个副本的新词条
   if ( empty() ) insertAsFirst( new Quadlist< Entry<K,V> > ); //首个Entry
   ListNode< Quadlist< Entry<K,V> >* >* qlist = first(); //从顶层列表的
   QuadlistNode< Entry<K,V> >* p = qlist->data->first(); //首节点开始
   if ( skipSearch( qlist, p, k ) ) //查找适当的插入位置——若已有雷同词条，则
      while ( p->below ) p = p->below; //强制转到塔底
   qlist = last();
   QuadlistNode< Entry<K,V> >* b = qlist->data->insertAfterAbove( e, p );
   /* ... 以下，紧邻于p的右侧，以新节点b为基座，自底而上逐层长出一座新塔 ... */
```

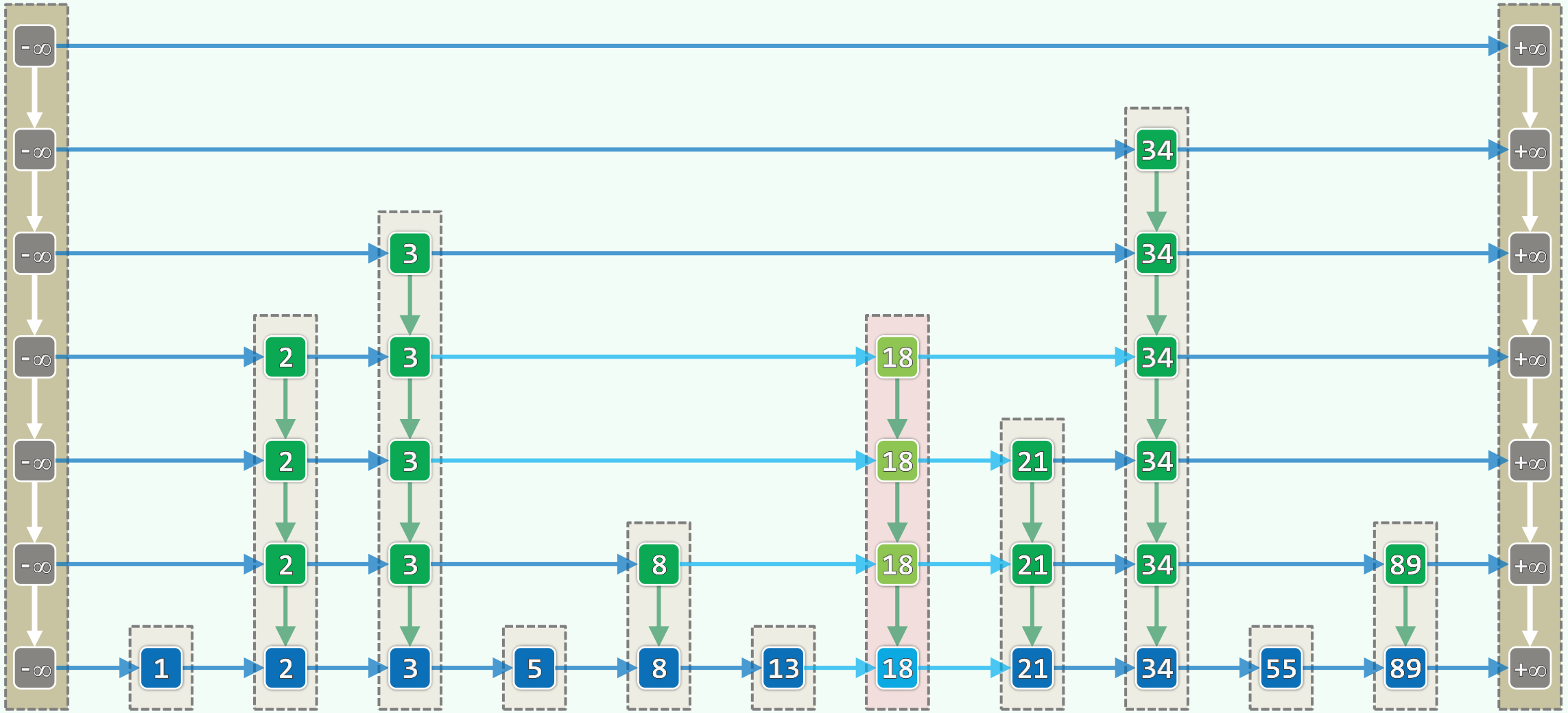
插入算法（2/2）

```
while ( rand() & 1 ) { //经投掷硬币，若新塔需再长高，则先找出不低于此高度的...
   while ( qlist->data->valid(p) && ! p->above ) p = p->pred; //最近前驱
   if ( ! qlist->data->valid(p) ) { //若该前驱是header
      if ( qlist == first() ) //且当前已是最顶层，则意味着必须
         insertAsFirst( new Quadlist< Entry< K, V > > ); //先创建新层，再
      p = qlist->pred->data->first()->pred; //将p转至上一层的header
   } else p = p->above; ////否则，可径自将p提升至该高度
   qlist = qlist->pred; //上升一层，并在该层将新节点
   b = qlist->data->insertAfterAbove( e, p, b ); //插至p之后、b之上
} //while ( rand() & 1 )
return true; //Skiplist允许重复元素，故插入必成功
}
```

实例：put(18)之前 = remove(18)之后



实例：put(18)之后 = remove(18)之前



删除算法（1/2）

```
//插入的逆过程
template <typename K, typename V> bool Skiplist< K, V >::remove( K k ) {
   if ( empty() ) return false; //空表
   ListNode< Quadlist< Entry< K, V > >* >* qlist = first(); //从顶层Quadlist
   QuadlistNode< Entry< K, V > >* p = qlist->data->first(); //的首节点开始
   if ( ! skipSearch( qlist, p, k ) ) //目标词条不存在，则
      return false; //直接返回
   /* ... TBC ... */
```

删除算法（2/2）

```
do { //若目标词条存在，则逐层拆除与之对应的塔
   QuadlistNode< Entry< K, V > >* lower = p->below; //记住下一层节点
   qlist->data->remove(p); //删除当前层的节点后，再
   p = lower; qlist = qlist->succ; //转入下一层
} while ( qlist->succ ); //如上不断重复，直到塔基
while ( ! empty() && first()->data->empty() ) //逐一地
   List::remove( first() ); //清除已可能不含词条的顶层Quadlist
return true; //删除操作成功完成
} //体会：得益于哨兵的设置，哪些环节被简化了？
```