展示讲稿

**·Dijkstra's algorithm**

**算法步骤：**1.初始化：将起始节点设置为当前节点，并将起始节点到自己的距离设置为0，所有其他节点到起始节点的距离设置为无穷大。同时创建一个空的集合S用于存储已经找到了最短路径的节点，集合T用来存储剩余的点。

2.遍历：从起始节点开始遍历，找到与当前节点距离最短的相邻节点。在所有未被访问的节点中，选择一个到起始节点距离最短的节点，并将其标记为已访问。将当前节点设置为该节点，并更新该节点的相邻节点的距离。

3.更新距离：对于当前节点的所有相邻节点，如果从起始节点到该节点的距离通过当前节点更短，就更新该节点的距离。即如果当前节点到相邻节点的距离加上起始节点到当前节点的距离小于起始节点到相邻节点的距离，则更新起始节点到相邻节点的距离为该值。

4.重复：重复步骤2和3，直到所有节点都被标记为已访问或者不存在与起始节点相连的未访问节点。

1. ***Heap Optimization***

算法中每次寻找T集中dis值最小的点的复杂度为O(n)，我们可以用堆（heap）来优化这一过程，使其复杂度降到O(logn)，这样总体复杂度降到O(nlogn)

1. ***Four heaps***

在这个Project当中，我们小组实现了四个堆，分别是斐波那契堆、二叉堆、左式堆、二项堆、这些堆相关操作的时间复杂度如右表所示。

**·斐波那契堆**

**定义：**一个斐波那契堆是一系列具有最小堆序 (min-heap ordered) 的有根树的集合。也就是说，每棵树均遵循最小堆性质 (min-heap property): 每个结点的关键字大于或等于它的父结点的关键字。（右图就是斐波那契堆的一个例子）

我们需要注意到的是与二项堆中树都是有序的不同，斐波那契堆中的树都是有根而无序的。并且对于另一结点，其孩子结点都被连接成一个环形双链表。我们通常会用指针H.min来访问一个给定的斐波那契堆，该指针指向具有最小关键字的树的根结点。

**结点的结构：**key（结点的值） father（指向父结点的指针） child（指向孩子结点的指针） rbro、lbro（指向左右兄弟结点的指针） degree（结点的孩子的数量） mark（布尔值，用于指示结点自从上一次成为另一个结点的孩子后，是否失去过孩子）

1. ***斐波那契堆的操作***

***MAKE-HEAP：***该操作会创建并返回一个不包含任何元素的新堆。

***MINIMUM：***返回一个指向堆H中键最小的元素的指针，也就是H.min，所以斐波那契堆找最小值的时间复杂度就是O(1）。

***INSERT：***插入一个结点，它的步骤为：1.对将要插入的结点进行初始化2.将其作为新树插入根表中3.检查并更新H.min指针

***UNION：***创建一个新堆，将H1和H2的根链表合并，并确立新的最小结点（H.min）

***EXTRACT-MIN:***从堆中删除最小关键字的元素，并返回一个指向该元素的指针。它的步骤如下：1.将被删结点的所有孩子作为新树插入到根表中2.将从堆H中删除3.调整根表。

***调整根表的步骤如下：***1.遍历根链表，将度数相同的根合并，直至每个度数的根至多只有一个2.遍历调整后的根链表，找到最小的根结点，把它赋值给H.min指针

***DECREASE-KEY：***减小堆中特定结点的值。它的步骤如下：1.先判断减小后的值是否仍大于父节点的值，如果是，则不执行任何操作，否则切断这个结点和父节点的连接，将其作为根结点加到根链表中，然后向上级联删除所有之前失去过孩子的结点。

值的注意的是，为了方便在Dijkstra 算法中更新路径的长度，我们用了一个名为nodeLib的二级指针来存储每个堆里点的位置，这样可以在O（1）的时间内找到需要减值的点。

下面的三个堆之前学过，所以我就简单讲一下。

**·Binary Heap**

**定义：**二叉堆是一个完全二叉树。除最深层外，所有内部节点都被完全填满，最深层的节点也从左到右被填充。存储在每个节点中的值总是大于等于或小于等于其子结点的值。

为了方便起见，我们组用STL中的最小优先队列来代替实现二叉堆。

**·Leftist Heap**

**定义：**左式堆是满足其任一节点左孩子的空节点路径长度，不小于右孩子的空节点路径长度性质的二叉树，其中的空节点路径长度指的是，该节点到外部节点的最短距离。

**Merge：**左式堆的常见操作是合并两个堆，它的算法是：比较需要合并的两个堆根结点值的大小，将较大的那个堆与较小那个堆的右节点进行递归合并操作，并将结果作为较小堆的新的右节点，在每次递归合并之后，都需要检查左孩子结点的NPL值是否仍大于右孩子结点的NPL值，如若不符，则交换左右子节点。

**·Binomial Heap**

**定义：**二项堆是一组由二项树组成的结构，其中的二项数可以由如下性质递归定义：1.、0阶的二项式树是单个节点 2、一个k阶的二项树有一个根节点，其子节点是k−1，k−阶的二项树，2，……，1,0（右图就是一个二项堆的例子）

二项堆的常见操作如下：

**MINIMUM：**寻找最小值。由最小堆性质，我们遍历根列表即可得到最小值。

**EXTRACT-MIN：**取出最小值。它的步骤是：找到具有最小值的根结点，将其从根链表中删除，并将它的子结点和剩下的根结点重新组合成一个新的二项堆。

**UNION：**1.将两个二项堆所有包含的二项树拆分，然后按度数递增的方式将所有的根列表进行串联，并创建一棵新的二叉树，使得其head节点指向这串联的根列表的第一个节点。 2.串连后，遍历根链表，将其中有相同度数的二项树进行合并，并使之符合最小堆性质

斐波那契堆的理论优势：

