**14 带权无向图与旅游区景点路线规划系统（难度：7 级）**

**任务一、构造带权无向图**

为了提高速度，图的存储使用两个数据结构来实现：双向邻接表和哈希表。请按照要求的时间复杂度来实现下列对图的具体操作，其中|v|表示图中的顶点数，d 是顶点的度数。

| 函数 | 函数功能 | 时间复杂度 |
|---|---|---|
| Initiate() | 新建一个无边无顶点的图 | O(1) |
| int vertexCount() | 返回图中的顶点数 | O(1) |
| int edgeCount（） | 返回图中的边数 | O(1) |
| getVertices() | 返回包含所有顶点的数组 | O（|v|） |
| void addVertex(Object) | 添加顶点 | O(1) |
| void removeVertex(Object) | 删除顶点 | O(d) |
| int isVertex(Object) | 判断该顶点是否在图中 | O(1) |
| int degree(int v) | 顶点的度 | O(1) |
| int getFirstNeighbor(int v) | 返回第一个邻接顶点 | O(1) |
| int getNextNeighbor(int v1,int v2) | 返回下一个邻接顶点 | O(d) |
| void addEdge(int v1,int v2) | 添加边 | O(1) |
| void removeEdge(int v1,int v2) | 删除边 | O(1) |
| int isEdge(Object,Object) | 判断是否为边 | O(1) |
| int weight(Object,Object) | 求某边的权值 | O(1) |

以下是一些具体的思想来实现特定操作来达到上述目的（可参考图 1）：

（1）对于图中顶点的存储，最好的方式是采用哈希结构来存储。哈希存储可以使得 isVertex()的时间复杂度为 O(1)。

（2）为使 getVertices()时间复杂度为 O(|V|)，则应设计链表来存储顶点序列；为使 removeVertex()时间复杂度为 O(d)，则该链表应该是双向的；

（3）为使得 getNeighbors()时间复杂度为 O(d)，需要对每个顶点创建邻接表来存放边；为使 removeEdge()是 O(1)，则此表应该是双向的。

（4）在无向图中，边（u，v）必须出现在两个链表中（u 顶点和 v 顶点的边邻接表）。若删除 u，则必须删除 u 的所有相关联的边，即使包含在 v 的链表中。为使 removeVertex()时间复杂度为 O(d)，不能遍历所有的邻接表，可以尝试用下列方法来得到 O(d）的时间。

既然(u,v)在图中出现于两个表中，则可以用两个结点来表示（u，v），分别在 u 表和 v 表中。这些结点每一个可称为“半条边”，其中一条边为另一条的“伙伴”。每一个半条边有向前和向后的指针链入邻接链表中，此外该半条边和其伙伴由“伙伴指针”相关联。采用这种方式，我们可以从图中删除 u 时，可遍历 u 的邻接链表，并使用伙伴指针来删除另外半条边只使用 O(1)的时间。

（5）为使 removeEdge()、isEdge()、weight()时间复杂度为 O(1)，你需要第二个哈希表来存储边，这个哈希表是成对出现的顶点在图中的表现形式。为使 removeVertex()时间复杂度为 O(d)，你需要从哈希表和顶点的邻接表上删除边

（6）为使 vertexCount(),edgeCount(),degree()时间复杂度为 O(1)，你需要添加边和顶点的计数器、每个顶点的度数，并且在每个操作注意其动态更新。

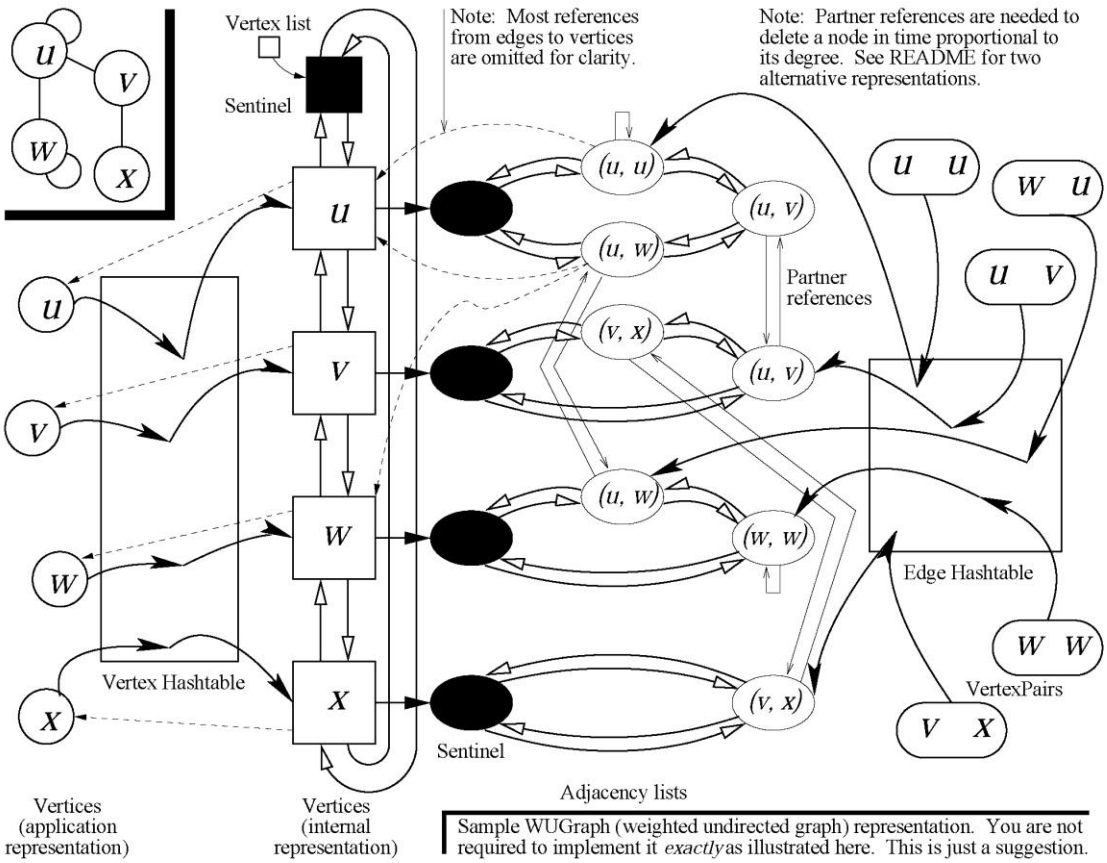


图 2 带权无向图数据结构设计参考

**任务二、基于该数据结构实现克努斯卡算法得到最小生成树。**

**任务三、基于该数据机构实现一个简易的旅游区景点路线规划系统。**

设某个旅游区共有 n 个旅游景点，每个旅游景点都和相邻的 m 个旅游景点（m≥2，m<n）有直接的道路（有对应的距离）相通，假设该旅游区的入口也是出口，请设计一个简易的旅游区导游系统。

（1）请建立景点库：景点、景点间的道路（以及距离）的增加/删除/修改。

（2）输入该旅游区的任意两个景点，请找出它们之间的最短简单路径和距离。

（3）用户一键知晓所有景点：输入景点库的任意一个景点，系统就会显示周围 R 公里范围内的所有景点。

（4）用户输入感兴趣的一组景点后，请选出一系列道路，能够连接所有输入景点以及出入口，并且总距离最短。

（5）最佳旅游路线确定：用户输入感兴趣的一组景点后，请确定一条最佳的旅游线路，该线路必须经过所有的输入景点（一次且仅仅一次），并且走的路尽可能最短。旅程中的最后一程必须从最后一个景点回到出口。

【选做内容】

（1）可以在显示器上简单显示地图、路线规划结果图。

（2）简单的可视化的界面设计。