四叉树的邻居关系

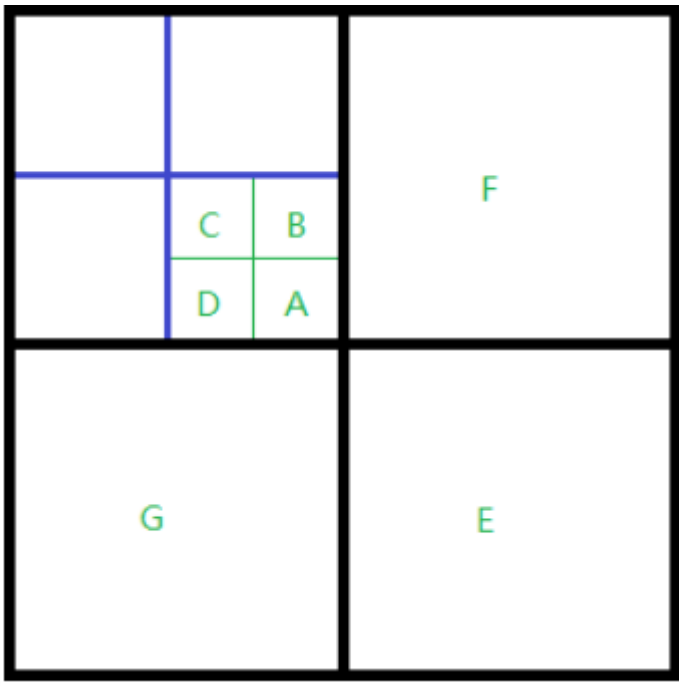
四叉树的邻居结点

常见的AOI使用Tile为基础，来实现。

每个Tile周围有8个邻居。因此在游戏对象移动或AOI时，可以O(1)的时间复杂度，定位8个邻居Tile。

而经典的四叉树代码实现，是没有邻居节点概念的。

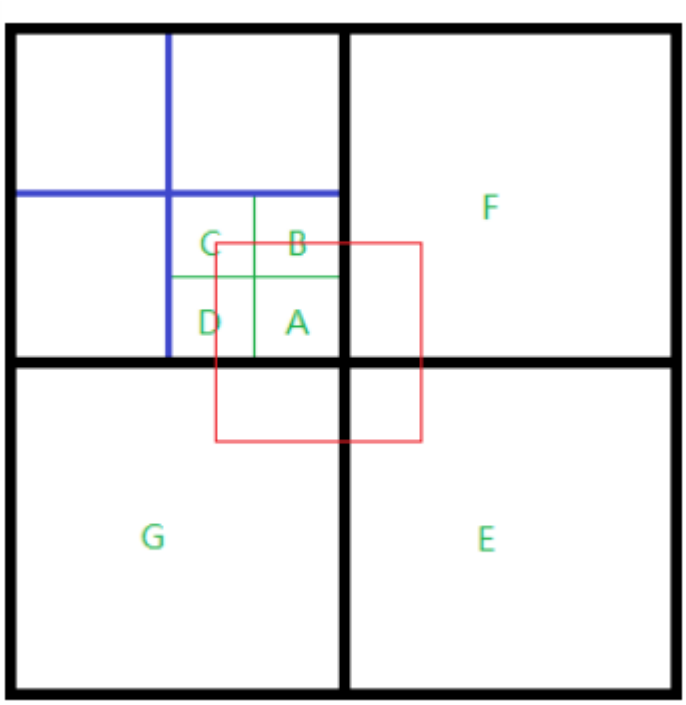
图1，A节点的邻居节点：



A节点有B、C、D、E、F、G邻居节点。

经典的四叉树代码实现，是需要从根节点开始遍历，才能够访问到邻居节点E、F、G。

图2，某AOI操作：

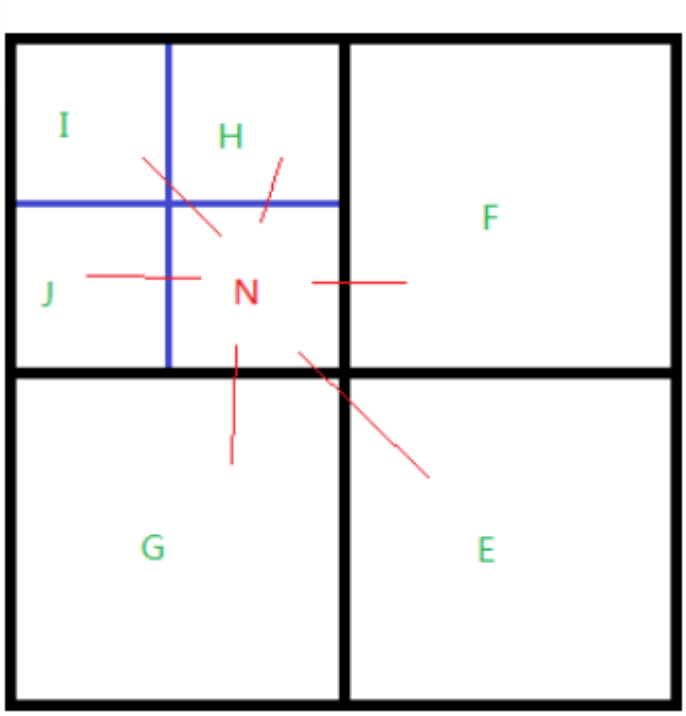


红色区域的AOI，经典四叉树实现上，从根节点宽度优先遍历，按红色区域是否与节点区域有相交或包含作为条件，遍历之。时间复杂度为`O(logN)`。

如果A节点知道自己的邻居节点，那么可以`O(1)`的时间复杂度，完成需要处理的节点定位

有邻居关系的四叉树的AOI操作

图3，A上的红色区域的AOI：



只要遍历A节点及其所有邻居节点，按红色区域是否与节点区域有相交或包含作为条件，遍历之；A的每个邻居节点递归重复操作。

以上操作与Tile上的AOI操作是同时间复杂度的。且比经典四叉树实现高效。

如何创建四叉树的邻居关系

图4，若N节点已经知道自己的邻居关系：

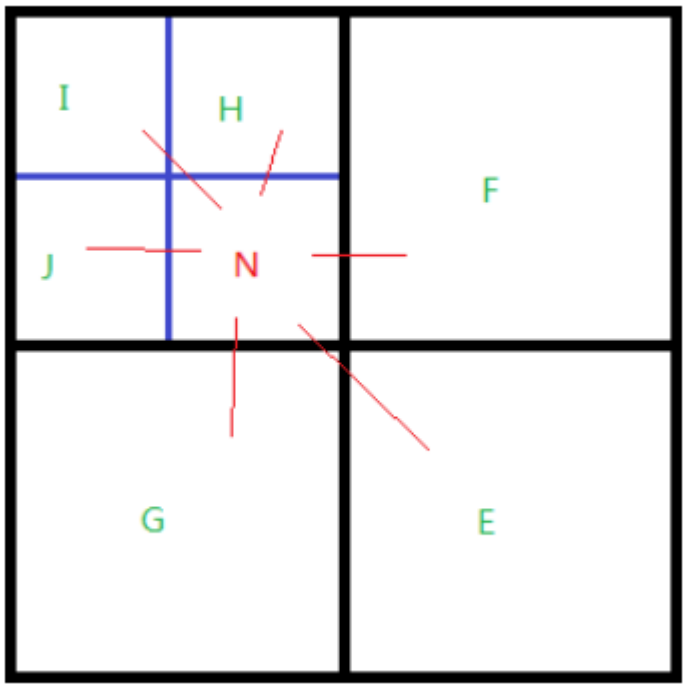
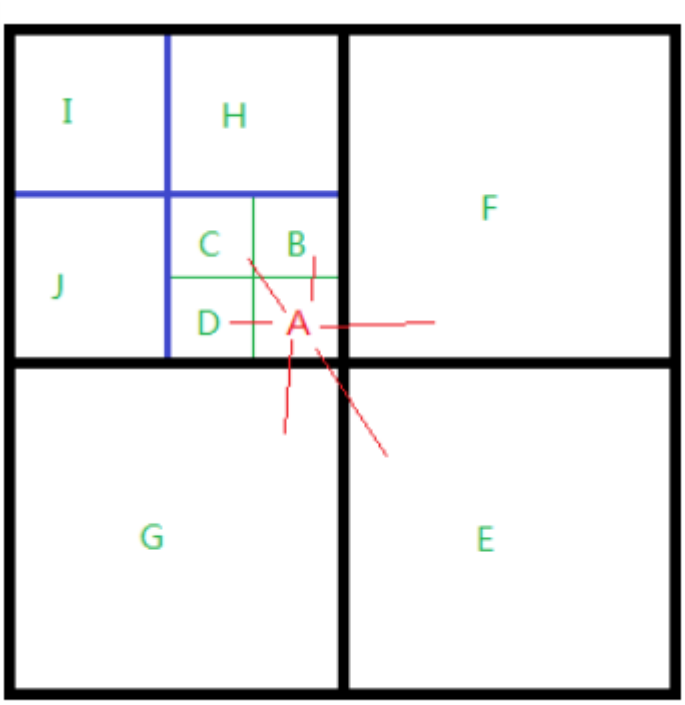


图5，那么N节点分裂时，只要维护下孩子节点与邻居节点的关系即可：



1. L为N节点的邻居节点列表
2. 遍历L，删除邻居节点对N的邻居信息
3. N节点变成非叶节点，不再需要邻居信息，删除这些信息之
4. N节点分裂为A、B、C、D4个孩子节点
5. 对每个孩子节点，遍历L，根据节点区域是否相邻，构建孩子节点与L列表中节点的邻居信息