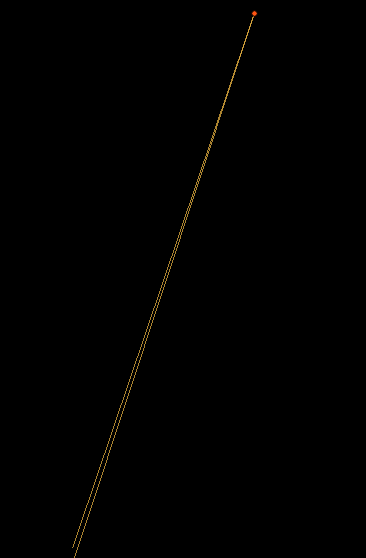
在进行空间运算（例如Union），可能遇到这个问题，主要错误可能是相交点数不够，如下图



原因有可能是Shp图层中节点坐标的精度问题，比如某个点的坐标是43.23，另外一个点是43.231499999999996这样的，从而造成了几条线段虽然看起来是重合的，但实际上有微小的差距，导致中间有条小缝

解决的办法之一：

对所有参与运算的空间对象，做一个外扩0.000001的Buffer操作，这样基本不会改变原有对象的空间位置和形状，但会消除之前提到的精度问题

但这样做的话，每操作一次，对象就会外扩一点，而且Buffer操作会增加多边形的节点数目，导致每做一次BUFFER，SHp文件就会变大一次

要解决这个问题，可以尝试在每做一次Buffer的同时做一次0.000001阈值的Simplify操作，如下所示

targetGeo.Buffer(0.000001,0).Simplify(0.000001)

其中Buffer的第二个参数是精细度，几何意义为用多少个点来表示一个90度的直角，直观理解为：这个值越大，新增节点数就越多，Shp文件变大得越快

如果没有特殊需求，在本文的功能需求中，取1或0都行

解决的办法之二：

对Shp文件涉及所有多边形的所有节点进行一次操作，把所有节点对齐到一个大小为0.0000001的网格上，如下所示：

private Geometry SetGeometryToGrid(Geometry inputGeo, int precition)

{

Geometry resGeo = null;

//只针对多边形对象

//if (resGeo.GetGeometryType() != wkbGeometryType.wkbPolygon && resGeo.GetGeometryType() != wkbGeometryType.wkbMultiPolygon)

// return null;

int tmpCount = inputGeo.GetGeometryCount();

if (tmpCount == 0)

{

//当前已经是最基础一层了，则可以直接获取点

int tmpPntsCount = inputGeo.GetPointCount();

double[] pntCoords = new double[2];

for (int i = 0; i < tmpPntsCount; i++)

{

inputGeo.GetPoint\_2D(i, pntCoords);

//这里还应该判断一下该点与重心点的位置关系，按位置关系（X大于还是小于，Y大于还是小于），取向外的方向变化，例如X小于中心点X，那就先减一个1E-6，再取Round

inputGeo.SetPoint\_2D(i, Math.Round(pntCoords[0], precition), Math.Round(pntCoords[1], precition));

}

return inputGeo;

}

else

{

//如果不是，迭代其子对象，并重新组合

//List<Geometry> subGeos = new List<Geometry>();

resGeo = new Geometry(inputGeo.GetGeometryType());

for (int i = 0; i < tmpCount; i++)

{

//subGeos.Add(SetGeometryToGrid(inputGeo.GetGeometryRef(i), precition));

resGeo.AddGeometry(SetGeometryToGrid(inputGeo.GetGeometryRef(i), precition));

}

return resGeo;

}

//return resGeo;

}

该方法还不是很完善，有可能虽然取过了，但线段还是不完全重合，有待后续完善