地理信息系统应用程序设计与开发

第八章 空间计算与分析

主讲：张宝一
Email：zhangbaoyi.csu@qq.com

教学目标

- 熟悉空间计算与空间分析的概念
- 掌握空间拓扑运算和空间关系运算的方法
- 了解栅格图层分析方法
- 熟悉空间统计分析方法
- 熟悉地理处理工具的调用方法

教学重点和难点

- 空间拓扑运算和空间关系运算

教学内容

- 8.1 空间计算与空间分析概念
- 8.2 空间查询
- 8.3 空间拓扑运算
- 8.4 空间关系运算
- 8.5 空间统计分析
- 8.6 地理处理工具
- 8.7 栅格数据分析
- 8.8 开发实例

8.1 空间计算与空间分析概念

- 空间计算是利用空间拓扑算子、关系算子对空间几何图形数据进行计算，从而得到新的空间数据的计算过程。
  - 空间计算强调数据的处理，产生新的空间数据。
- 空间分析是对空间对象的位置和属性数据进行分析的一系列技术，通过“产生式分析”得到新的空间信息的过程(**Goodchild**)
- 空间分析是基于地理对象的位置和形态特征的空间数据分析技术，目的在于提取和传输空间信息(**郭仁忠**)
- 空间计算是空间分析的基础，空间分析通过空间计算获取新的信息

8.1 空间计算与空间分析概念

- 广义上讲，空间分析包括空间计算、空间统计、定量地理计算等内容。
- 空间分析可以分为：
  - 基于空间图形(几何)数据的分析运算
  - 基于空间变量(属性)的数据运算
  - 基于图形和变量数据的联合运算
- **讨论：** 请举出几个空间分析的示例

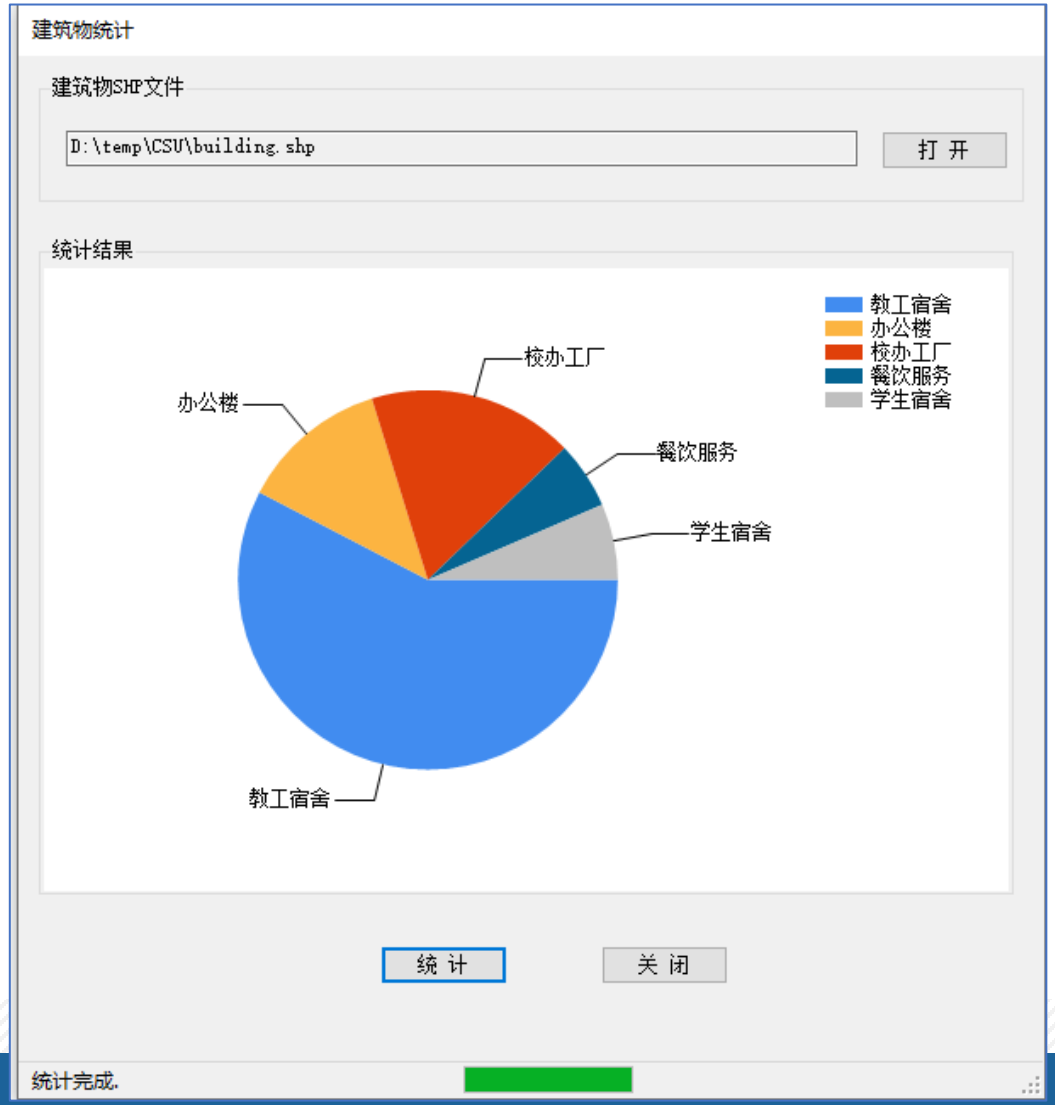
8.2 空间查询

- 空间查询是通过条件筛选空间要素的过程，包括属性筛选和空间关系的筛选。
- 属性查询(筛选)
  - 利用IQueryFilter接口设置WhereClause语句，实现要素的查询
- 空间查询(筛选)
  - 利用ISpatialFilter接口，可同时包含空间约束和属性约束的过滤条件，须设置Geometry（几何）、GeometryField（几何字段）和SpactialRel（空间关系）
  - **ISpatailFilter继承了IQueryFilter**

- (1)IQueryFilter接口

Name	Description
AddField	Appends a single field name to the list of sub-fields.
OutputSpatialReference	The spatial reference in which to output geometry for a given field.
SubFields	The comma delimited list of field names for the filter.
WhereClause	The where clause for the filter.

- (2)[示例] 统计校本部各类建筑物的数量，并绘制统计饼图。



8.2 空间查询

- (2)[示例1] 方法1

```
①  //定义数据系列
②  List<String> cats = new List<string>();          List<int> nums = new List<int>();
③  string cat;
④  IFeatureCursor cursor = this.featureClass.Search(null, true);          IFeature feat;
⑤  //当前类别在分类集合的序次
⑥  int seq;
⑦  while((feat=cursor.NextFeature())!=null)     {   //获取分类
⑧     cat = feat.Value[catIndex].ToString();
⑨     //获取类别在分类集合中的序次
⑩     seq = cats.IndexOf(cat);
⑪     //如果小于0，表示未找到，则需要加入分类集
⑫     if (seq < 0)     {
⑬         cats.Add(cat);
⑭         // 最后一个就是当前分类的序次
⑮         seq = cats.Count - 1;
⑯         //数量集也要加入一行
⑰         nums.Add(0);
⑱     }
⑲     nums[seq]++;  //统计数量+1
⑳  }
```

```
21  //chart是C#统计图控件
22  chart.Series[0]["PieLabelStyle"] = "Outside";//将文字移到外侧
23  chart.Series[0]["PieLineColor"] = "Black";//绘制黑色的连线。
24  chart.Series[0].Points.DataBindXY(cats, nums);
```

8.2 空间查询

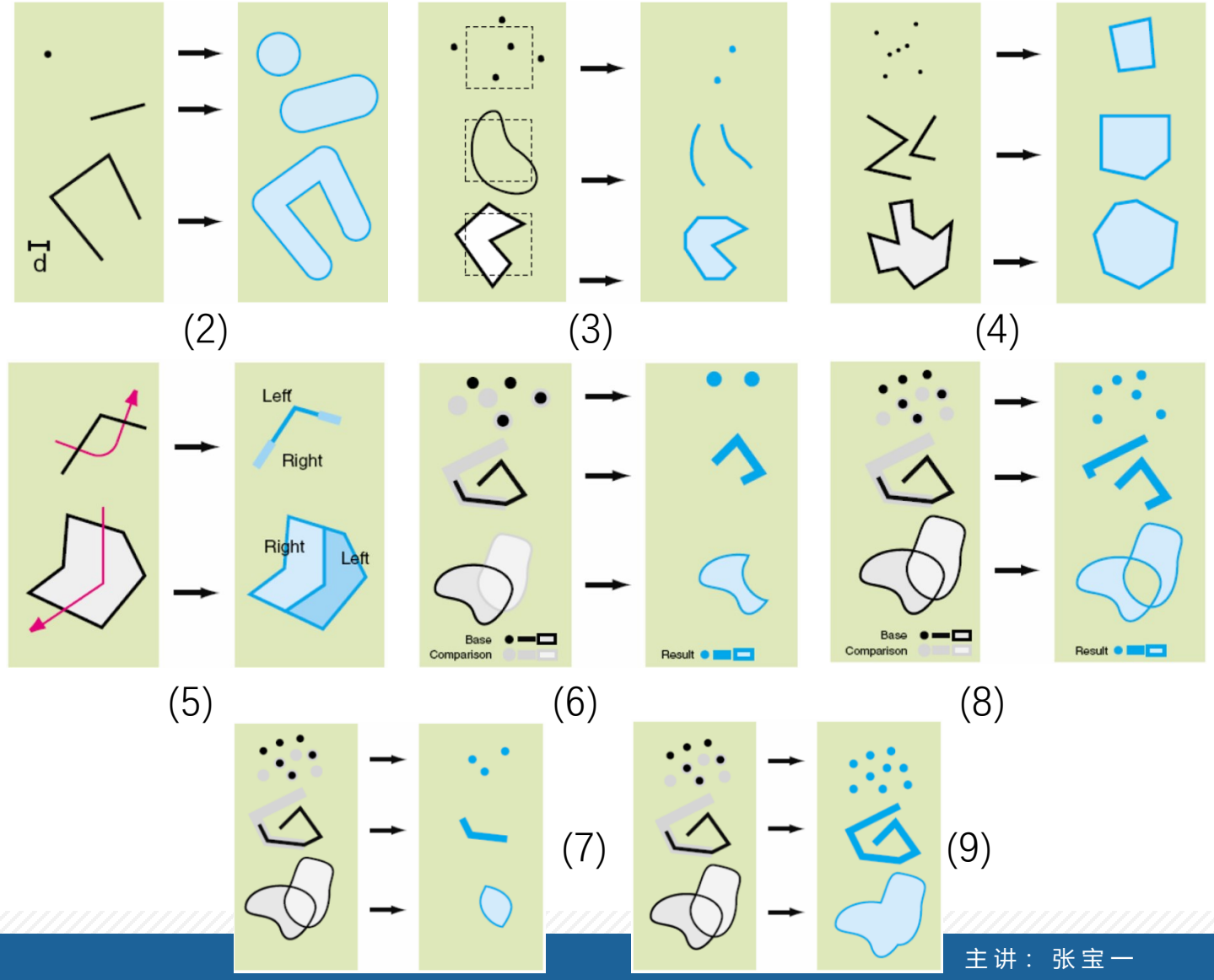
- (2)[示例1] 方法2

```
①  //定义数据系列
②  List<String> cats = new List<string>();          List<int> nums = new List<int>();
③  string cat;
④  IFeatureCursor cursor = this.featureClass.Search(null, true);          IFeature feat;
⑤  while ((feat = cursor.NextFeature()) != null)    {
⑥      cat = feat.Value[catIndex].ToString();
⑦      if (cats.IndexOf(cat)< 0) //不在分类集中，就加入
⑧          cats.Add(cat);
⑨  }
⑩  int n = 0;  //每类数据的计数器
⑪  IQueryFilter qf = new QueryFilterClass();
⑫  foreach(String str in cats)   {
⑬      qf.WhereClause = "Catagory = '" + str + "'";
⑭      cursor = this.featureClass.Search(qf, false);
⑮      n = 0;
⑯      while ((feat = cursor.NextFeature()) != null)
⑰          n++;
⑱     nums.Add(n);
⑲  }
```

```
21  //chart是C#统计图控件
22  chart.Series[0]["PieLabelStyle"] = "Outside";//将文字移到外侧
23  chart.Series[0]["PieLineColor"] = "Black";//绘制黑色的连线。
24  chart.Series[0].Points.DataBindXY(cats, nums);
```

8.3 空间拓扑运算

- 拓扑（topology）几何形状的空间关联
- 拓扑算子
  - (1)边界（Doundary）
  - (2)缓冲（Buffer）
  - (3)裁剪（Clip）
  - (4)凸包（ConvexHull）
  - (5)切割（Cut）
  - (6)差集（Difference）
  - (7)交集（Intersect）
  - (8)异集（SymmeticDifference）
  - (9)并集（Union）



8.3 空间拓扑运算

- [示例1] 图上点击绘制点、线，并计算其缓冲区
  - //点缓冲

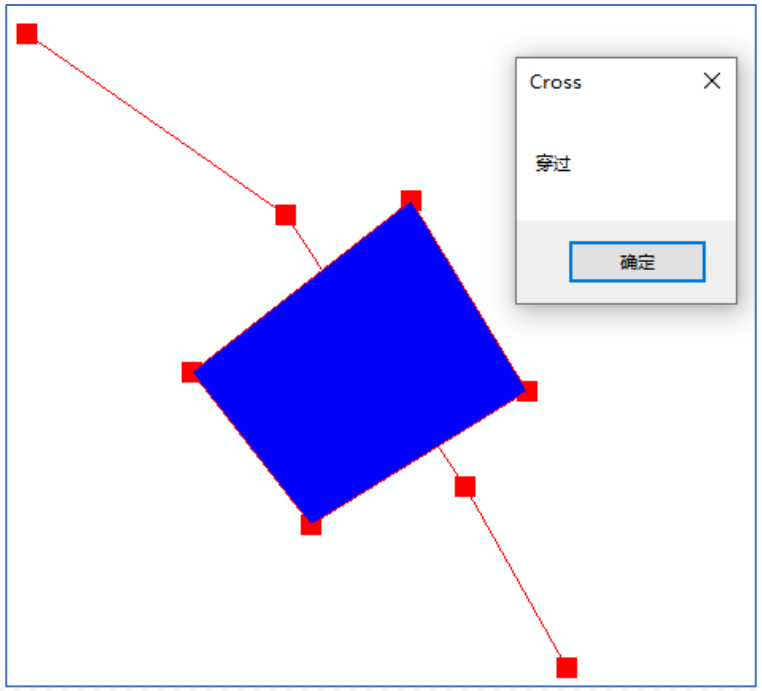
```
① IPoint pt = new PointClass();
② pt.PutCoords(e.mapX, e.mapY);
③ IElement marker = new MarkerElementClass();
④ marker.Geometry = pt;
⑤ this.axMap.ActiveView.GraphicsContainer.AddElement(marker, 0);

⑥ topo = (ITopologicalOperator)pt;
⑦ buffer = topo.Buffer(radius) as IPolygon;

⑧ IElement el = new PolygonElement();
⑨ el.Geometry = buffer;
⑩ ((IFillShapeElement)el).Symbol = fillSymbol;
⑪ this.axMap.ActiveView.GraphicsContainer.AddElement(el, 0);
```

- [示例2] 图上点击绘制线和多边形，计算线与多边形交集。
  - IPolyline line = this.pc as IPolyline;
  - IPolygon pg = this.pc2 as IPolygon;
  - ITopologicalOperator topo = (ITopologicalOperator)pg;
  - IGeometry geom = topo.Intersect(line, esriGeometryDimension.esriGeometry1Dimension);

8.4 空间关系运算

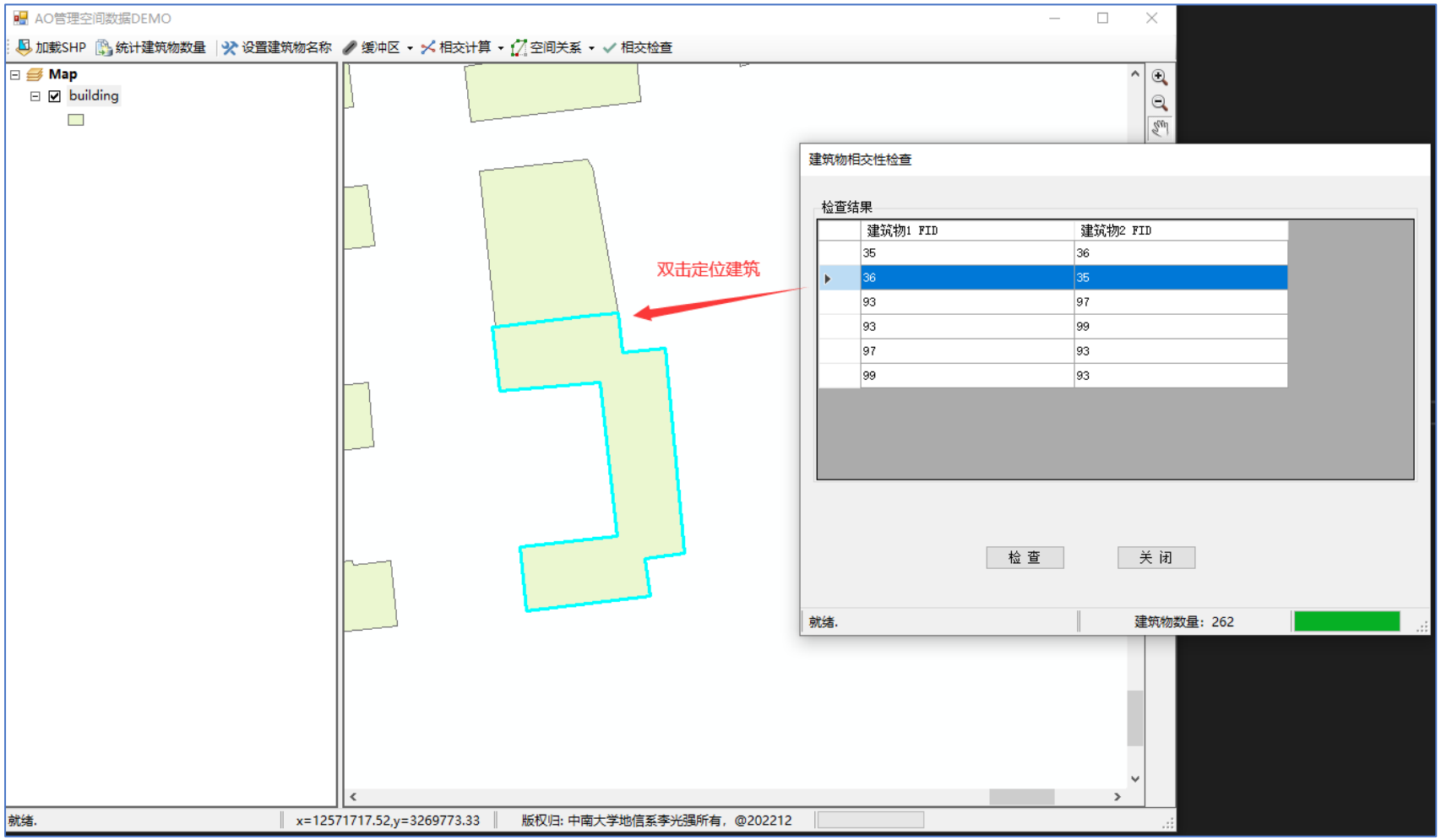
- 空间关系接口IRelationaloperator
  - 相离（disjoint）没有公共点
  - 相接（touch）两个几何图形在边界相交
  - 重叠（overlap）相交部分与两者有着相同维度的几何图形
  - 相交（cross）在比最高维度更低的维度中相交
  - 内含（within）一个几何图形不能内含于另一个更低维的几何图形
- [示例] 绘线和多边形，判断线是否穿过多边形

```
IPolyline line = this.pc as IPolyline;
IPolygon pg = this.pc2 as IPolygon;
IRelationalOperator ro = (IRelationalOperator)line;
if (ro.Crosses(pg))
      MessageBox.Show("穿过", "Cross");
else
    MessageBox.Show("不穿过", "Cross");
```



8.4 空间关系运算

- [例] 检查建筑物是否相交，若有则列出相交建筑物的FID，双击相交建筑物FID，在地图上定位该建筑物，即将双击FID对应建筑物移到地图中间放大显示。



8.4 空间关系运算

[例] 相交性检查程序：

```
①  IFeatureCursor cur1 = this.featureClass.Search(null, true) , cur2;
②  IFeature feat1, feat2;
③  IRelationalOperator ro;
④  while ((feat1 = cur1.NextFeature()) != null)
⑤  {
⑥      cur2 = this.featureClass.Search(null, true);
⑦      while ((feat2 = cur2.NextFeature()) != null)
⑧      {
⑨          if (feat1.OID == feat2.OID)   continue;
⑩          ro = feat1.Shape as IRelationalOperator;
⑪          if (ro.Overlaps(feat2.Shape))
⑫              this.dataGridView1.Rows.Add(feat1.OID, feat2.OID);
⑬      }
⑭  }
```

8.4 空间关系运算

[例] 双击表格定位地图程序：

```
① DataGridViewRow row = this.dataGridView1.Rows[e.RowIndex];
② int fid1= int.Parse(row.Cells[0].Value.ToString()), fid2= int.Parse(row.Cells[0].Value.ToString());
③ IQueryFilter qf = new QueryFilterClass();
④ qf.WhereClause = "FID = " + fid1;
⑤ IFeatureCursor cur = this.featureClass.Search(qf, true);
⑥ IFeature feat1 = cur.NextFeature();
⑦ qf.WhereClause = "FID = " + fid2;
⑧ cur = this.featureClass.Search(qf, true);
⑨ IFeature feat2 = cur.NextFeature();
⑩ this.axMap.Map.SelectFeature(layer,feat1);      this.axMap.Map.SelectFeature(layer, feat2);

⑪ ITopologicalOperator topo = (ITopologicalOperator)feat1.Shape;
⑫ IGeometry geom = topo.Union(feat2.Shape);
⑬ IEnvelope env = geom.Envelope;        env.Expand(1.3, 1.3, true);
⑭ this.axMap.Extent = env;
⑮ this.axMap.ActiveView.PartialRefresh(esriViewDrawPhase.esriViewGraphics, null, null);
```

8.5 空间统计分析

- 统计分析用于对表格或者属性表进行统计计算，如频率、平均值、最小值、最大值和标准差等。在实际使用中，统计分析常用来数据探索和数据汇总。
  - 频数：在表格或者图层的属性表中，属性值或者属性组合出现的次数。频数工具的主要作用是读取表格中的一组字段，计算字段的每个唯一值出现的频数，并创建一个包含唯一字段及其频数的新表。频数工具可以进行分类统计以及分类求和。
  - 汇总统计数据：对输入表格中的字段进行汇总计算，输出结果为表格，表格由包含统计运算结果的字段组成。
- 空间统计分析使用统计方法分析地理空间变量，旨在发现地理空间规律和模式，解释空间要素存在/分布特征，不是预测（?）。
- 空间统计和传统统计方法在概念和目标方面存在相似性，但空间统计具有其固有的独特性，因为空间统计方法是专门为处理地理数据而开发的。空间统计方法是将地理空间（邻域、区域、连通性和/或其他空间关系）直接融入到数学逻辑中[ESRI]。

8.5 空间统计分析

- ArcGIS空间统计工具箱包含用于分析空间分布、模式、过程和关系的统计工具

工具集	说明	工具
分析模式	聚类空间模式、离散空间模式、随机空间模式	平均最近邻、高/低聚类、空间自相关、多距离空间聚类
聚类分布制图	识别具有统计显著性的热点、冷点或空间异常值，以及多种工具用于识别或分组具有类似特征的要素。	聚类和异常值分析、分组分析、热点分析、优化的热点分析、优化的异常值分析、相似搜索
度量地理分布	度量中心位置、几何形状和方向等	中心要素、方向分布、线性方向平均值、平均中心、中位数中心、标准距离
空间关系建模	回归分析建立数据关系模型，构建空间权重矩阵	探索性回归、生成网络空间权重、生成空间权重矩阵、地理加权回归、普通最小二乘法
实用工具	计算面积、评估最小距离、导出变量和几何、转换空间权重文件和采集重合点	计算近邻点距离、收集事件、将空间权重矩阵转换为表、将要素属性导出到ASCII

8.6 地理处理工具

- 地理处理框架是指ArcGIS用于建立自动化处理流程的应用环境和开发框架，其主要内容包括：地理处理工具、模型构造器(Model Builder)、脚本、ArcObjects中的地理处理。
- Geoprocessor是ArcEngine的一个基于.net Framework的托管类，所有的Geoprocessing工具，包括扩展工具，都是由Geoprocessor对象调用运行.
- Geoprocessor通过设置不同的环境参数，简化执行Geoprocessing工具的过程，并返回相应的处理结果。
- 地理处理(Geoprocessing)提供一组丰富的工具和机制来实现GIS工作流的自动化操作，这些工具和机制能够使用模型、脚本、高级开发语言将一系列的工具按照一定操作顺序结合在一起，完成更复杂的GIS工作流。

8.6 地理处理工具

- 在ArcGIS通过ArcToolbox提供了大量的地理处理工具，AE为每类工具箱设定了对应类库的命名空间，如ESRI.ArcGIS.*Tools

工具名称	命名空间
3D Analyst tools	ESRI.ArcGIS.Analyst3DTools
Analysis tools	ESRI.ArcGIS.AnalysisTools
Conversion tools	ESRI.ArcGIS.ConversionTools
Data Management tools	ESRI.ArcGIS.DataManagementTools
Cartography tools	ESRI.ArcGIS.CartographyTools
Coverage tools	ESRI.ArcGIS.CoverageTools
Geocoding tools	ESRI.ArcGIS.GeocodingTools
Geostatistical Analyst tools	ESRI.ArcGIS.GeostatisticalAnalystTools
Linear Referencing tools	ESRI.ArcGIS.LinearReferencingAnalystTools
Multidimension tools	ESRI.ArcGIS.MultidimensionTools
Network Analyst tools	ESRI.ArcGIS.NetworkAnalystTools
Samples	ESRI.ArcGIS.SamplesTools
Spatial Analyst tools	ESRI.ArcGIS.SpatialAnalystTools
Spatial Statistics tools	ESRI.ArcGIS.SpatialStatisticsTools

8.6 地理处理工具

- AE调用GP工具有两种接口方式：
  - 方式1 直接使用Geoprocessor类，执行工具类

```
// using ESRI.ArcGIS.Geoprocessor;
Geoprocessor GP = new Geoprocessor();
```

  - 方式2 使用IGeoProcessor2接口，执行工具名称

```
// using ESRI.ArcGIS.Geoprocessing;
IGeoProcessor2 GP = new GeoProcessor() as IGeoProcessor2;
```

GP工具调用方式1 示例：

```
①  //初始化GP
②  Geoprocessor GP = new Geoprocessor();
③  //初始化Buffer
④  ESRI.ArcGIS.AnalysisTools.Buffer buffer = new Buffer();
⑤  //输入要素
⑥  buffer.in_features = @"D\data\temp.gdb\road";
⑦  //输出要素类
⑧  buffer.out_feature_class = @"D\data\temp.gdb\road_bf30";
⑨  //缓冲距离
⑩  buffer.buffer_distance_or_field = 30; //默认单位
⑪  //执行工具
⑫  GP.Execute(buffer, null);
```

GP工具调用方法2 示例：

```
①  //初始化GP
②  IGeoProcessor2 gp2 = new GeoProcessor() as IGeoProcessor2;
③  gp2.OverwriteOutput = true;
④  IVariantArray parameters = new VarArrayClass();
⑤  parameters.Add(shpFileName);
⑥  parameters.Add(newShpFileName);
⑦  parameters.Add("10 meters");
⑧  // Execute the tool.
⑨  gp2.Execute("Buffer_analysis", parameters, null);
```

8.6 地理处理工具

- 访问GP工具执行结果接口
  - IGeoProcessorResult result=GP.Execute(...);
  - 接口常用属性和方法
    - ReturnValue : 获取执行返回值，如缓冲区计算返回的缓冲区要素类名称
    - GetMessage( ) : 获取返回的消息
    - GetOutput( ) : 获取输出结果
    - OutputCount : 输出结果数量
    - MessageCout : 消息数量
    - Status : 执行状态
    - Cancel( ) : 取消当前任务
  - 示例:
    - IFeatureClass mFeatureClass=GP.Open(result.ReturnValue);

GP工具环境变量设置

① //获取和设置环境：单元格大小

② String env = (String) gp.getEnvironmentValue("cellsize");
gp.setEnvironmentValue("cellsize", Double.valueOf(10.0));

③ // 设置输出坐标系统
gp.setEnvironmentValue("outputCoordinateSystem", "c:/Program Files/ArcGIS/Coordinate Systems/Projected Coordinate Systems/UTM/Nad 1983/NAD 1983 UTM Zone 21N.prj");

④ // 重置

⑤ gp.resetEnvironments();

8.6 地理处理工具

利用GP工具列举工作区要素类

```
① List<IFeatureClass> lstFeatureClass = new List<IFeatureClass>();
② GP.SetEnvironmentValue("workspace", filePath);
③ IGpEnumList featureClasses = GP.ListFeatureClasses( “*” ,featureType,dataset);
④ string featureClass = featureClasses.Next();
⑤ while(featureClass!="")
⑥ {
⑦  lstFeatureClass.Add(featureClass);
⑧   featureClass = featureClasses.Next();
⑨ });
```

8.6 地理处理工具

- 调用GP工具的注意事项
  - (1)GP工具对输入、输出参数要求严格，因此首先要保证数据没有问题，数据路径中最好不出现中文。
  - (2)需要设置输入参数的多个值时，要使用**分号**隔开，如：
    - union.in_features = @"d:\data\a.shp;d:\data\b.shp";
  - (3) “比较复杂的参数，可以参考Python调用GP工具的示例来设置参数

8.7 栅格数据分析

- 栅格数据是由一系列等间距的格网矩阵组成，用来表达完整的主题、光谱、图像信息。
- 栅格数据模型分为
  - 栅格数据集（Raster dataset）,如jpg、tif等
  - 栅格目录（Raster catalog）,记录了一个或者多个栅格数据集，每一个栅格数据集都作为一条记录存储在栅格目录中
  - 镶嵌数据集（Mosaic dataset）,用于管理和发布海量多分辨率，多传感器影像，对栅格数据提供了动态镶嵌和实时处理的功能。

8.7 栅格数据分析

AE加载栅格数据

```
IWorkspaceFactory wsf = new RasterWorkspaceFactoryClass();
IRasterWorkspace rw = wsf.OpenFromFile(@"d:\temp\csu",0) as IRasterWorkspace;
IRasterDataset rds = rw.OpenRasterDataset("csu.tif");

//影像金字塔的判断与创建,使用IRasterPyramid3接口
IRasterPyramid3 pRasPyrmid;
pRasPyrmid = rds as IRasterPyramid3;    //接口转换
 if (pRasPyrmid != null)
    if (!(pRasPyrmid.Present))
        pRasPyrmid.Create();//创建金字塔

IRasterLayer rasterLayer = new RasterLayerClass();
//方法1:
//rasterLayer.CreateFromDataset(rds);
//方法2:
IRaster raster = rds.CreateDefaultRaster();
rasterLayer.CreateFromRaster(raster);

ILayer layer = rasterLayer as ILayer;
this.axMap.Map.AddLayer(layer);
```

8.7 栅格数据分析

- 栅格数据数据分析有两种方式
  - 利用接口和类
  - 利用GP工具
- [例]影像裁切

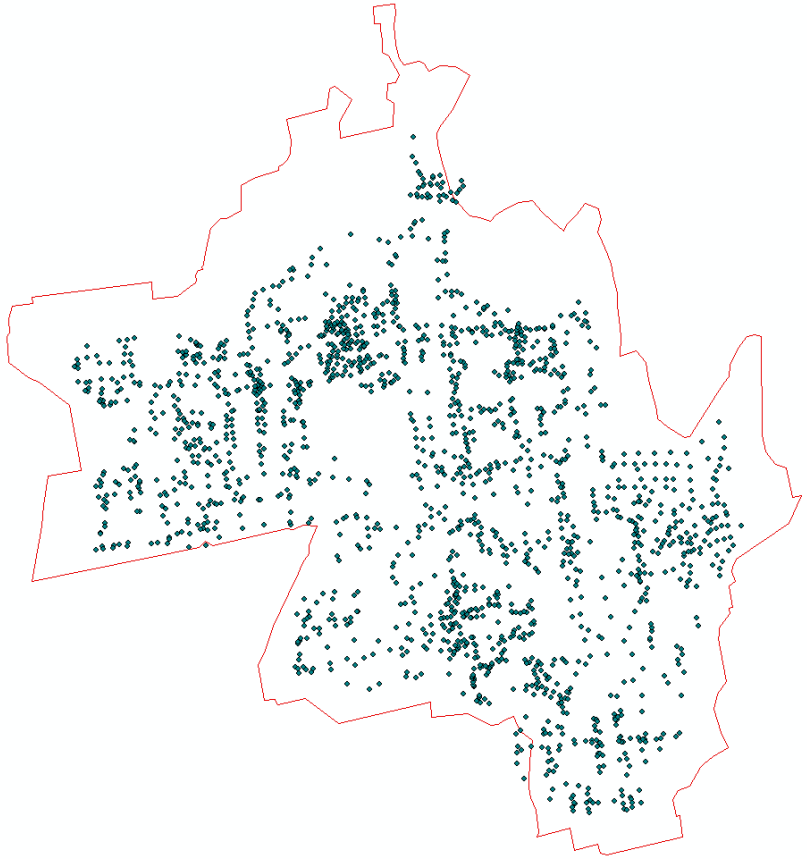
```
① IRasterLayer rl = this.selectedLayer as IRasterLayer;
② IRaster raster = rl.Raster;
③ IClipFunctionArguments rasterFuncArgs = new ClipFunctionArguments() as IClipFunctionArguments;
④ rasterFuncArgs.Raster = raster;
⑤ rasterFuncArgs.ClippingType = esriRasterClippingType.esriRasterClippingOutside;
⑥ rasterFuncArgs.ClippingGeometry = this.drawPolygon;

⑦ IRasterFunction clipFunction = new ClipFunctionClass();
⑧ IFunctionRasterDataset funcRasterDataset = new FunctionRasterDataset();
⑨ IFunctionRasterDatasetName funcRasterDatasetName = new FunctionRasterDatasetName() as
   IFunctionRasterDatasetName;
⑩ funcRasterDatasetName.FullName = @"d:\temp\clip.afr"; //不是保存
⑪ funcRasterDataset.FullName = (IName)funcRasterDatasetName;
⑫ funcRasterDataset.Init(clipFunction, rasterFuncArgs);

⑬ IRasterDataset rds = funcRasterDataset as IRasterDataset;
⑭ IRasterLayer rLayer = new RasterLayerClass();
⑮ rLayer.CreateFromDataset(rds);
⑯ this.axMap.Map.AddLayer(rLayer as ILayer);
```

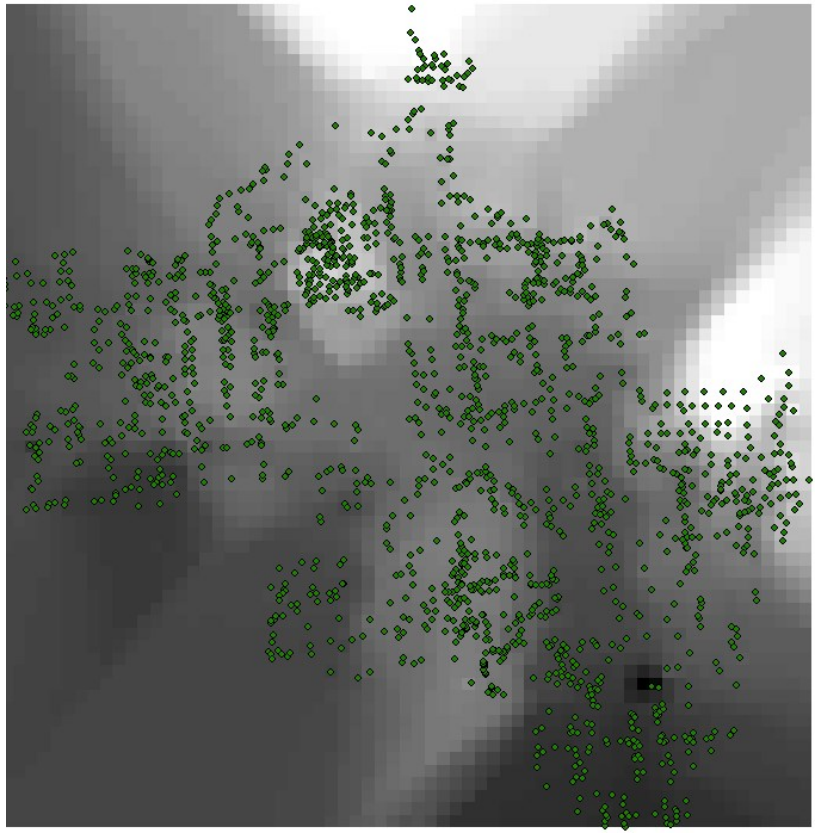
[例]保存影像

```
① string fmt = "TIFF";
② IRaster raster = ((IRasterLayer)this.selectedLayer).Raster;

③ //去黑边
④ IRasterProps pRasterProps = raster as IRasterProps;
⑤ pRasterProps.NoDataValue = 255;
⑥ //另存输出
⑦ ISaveAs2 pSaveAs = raster as ISaveAs2;
⑧ if (!pSaveAs.CanSaveAs("TIFF"))
⑨ {
⑩     MessageBox.Show("不支持指定像素类型或文件格式的输出", "提示"
⑪                     , MessageBoxButtons.OK, MessageBoxIcon.Information);
⑫     return;
⑬ }
⑭ IWorkspaceFactory worksapceFactory = new RasterWorkspaceFactoryClass();
⑮ IWorkspace workspace = worksapceFactory.OpenFromFile(@"d:\temp", 0);
⑯ IDataset dataset = pSaveAs.SaveAs("clip.tif", workspace, fmt);
```

8.8 综合实例

- 利用GP工具，根据校本部高程点数据，内插生成DEM数据，然后生成坡度图，最后进行重分类。
  - 操作步骤
    - (1)内插生成DEM
    - (2)生成坡度图
    - (3)重分类





本章小结