Splitting Geometry Algorithm

A **Polygon** is a two-dimensional geometric figure that has a finite number of sides. *(Eg. Figure-A)*

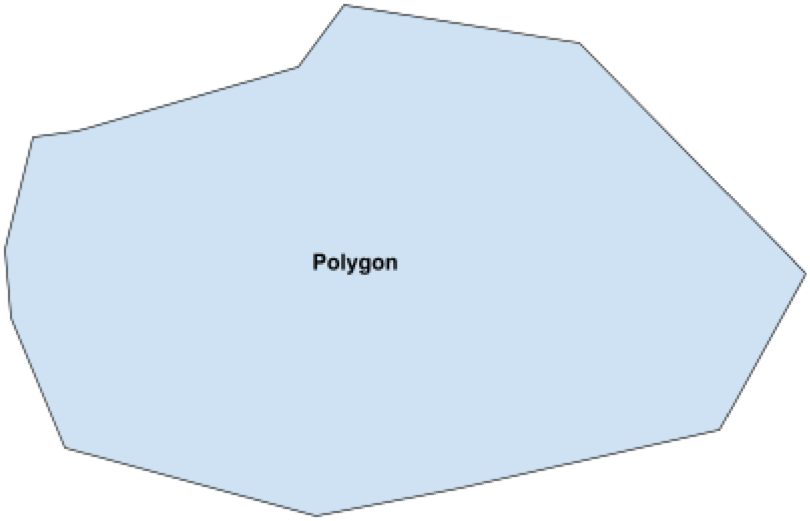

Figure- (A)

Each **Geometry** can be contained in an ***Envelope*** (acting as a bounding box that contains all the geometry coordinates). *(Eg. Figure-B)*

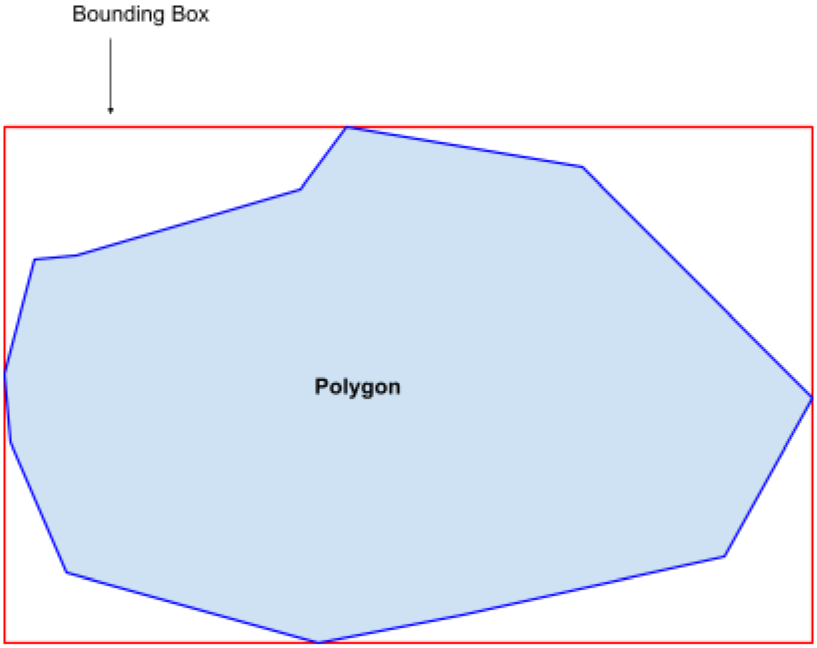

Figure- (B)

Envelope is defines a rectangular region of the 2D coordinate plane. It is often used to represent the bounding box of a Geometry, e.g. the minimum and maximum x and y values of the **Coordinates**. *(Eg. Figure- C)*

Coordinates : (xmin, ymin) and (xmax, ymax)

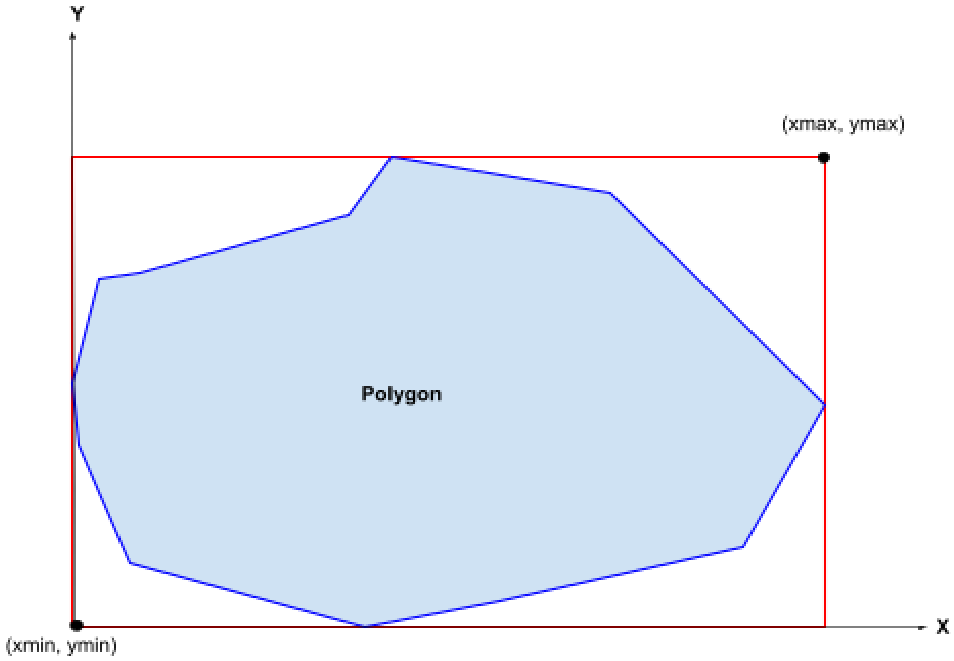

Figure (C)

BoundingBox is an extension of **Envelope** for working with 2D data (defining X and Y coordinate) and also, it's a 2-D plane, so I had created a grid on this plane.

The grid can be split equally on the X-axis and Y-axis or split unequally on both the axes. *(Eg. Figure-D)*

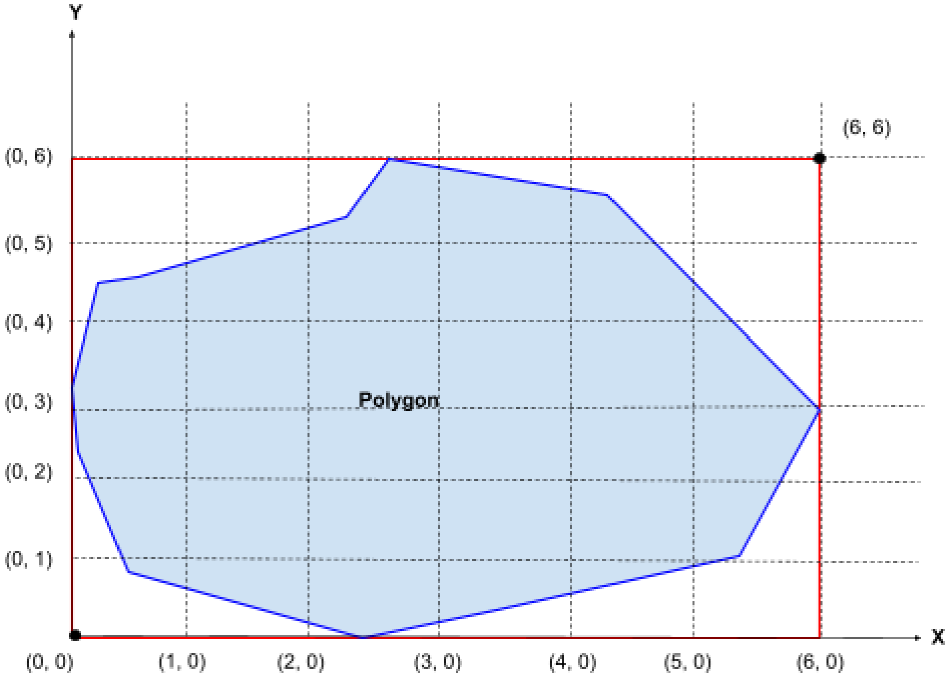

Figure (D)

Let's assume,

We have equally split the bounding box on both the axis and selected the first split grid. *(see, **shaded** grid on figure-E)*

Coordinates for first grid are:

```
(xmin, ymin) = (0,0)
(xmax, ymax) = (1,1)
```

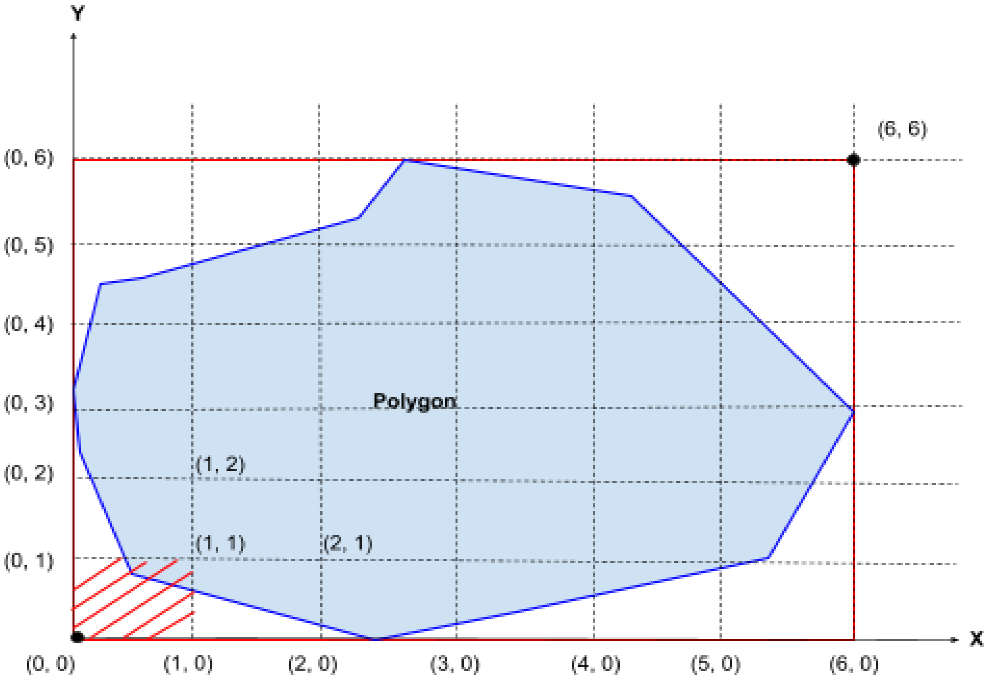

Figure (E)

The first grid is an **Envelope** in the plane and we have converted this **Envelope** into a new **Geometry**.

***Envelope** declaration is*

```
Envelope env = new Envelope(x1, x2, y1, y2);
```

Using the above envelope geometry, we intersect the original geometry, which means Polygon *(Figure-A)* and we will get the ***splitted*** geometry.

Therefore,

```
GeometryFactory gf = new GeometryFactory();
Envelope env = new Envelope(0, 1, 0, 1);
Geometry splitGeom = gf.toGeometry(env).intersection(geom);
```

Where,

geom is an object of original Polygon.

Result of split geometry can be as below image *(figure-F)*

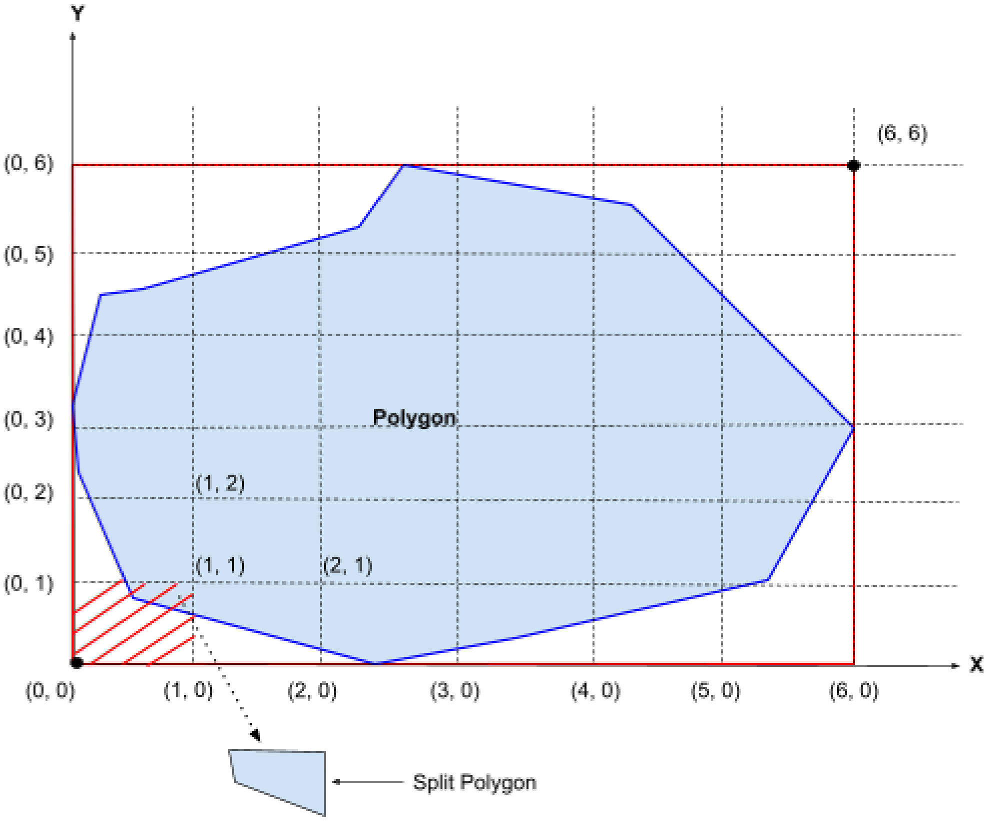

Figure (F)

Coordinates for second grid are:

```
    (xmin, ymin) = (1,0)
    (xmax, ymax) = (2,1)
```

The split geometry for second grid can be as below image *(figure-G)*

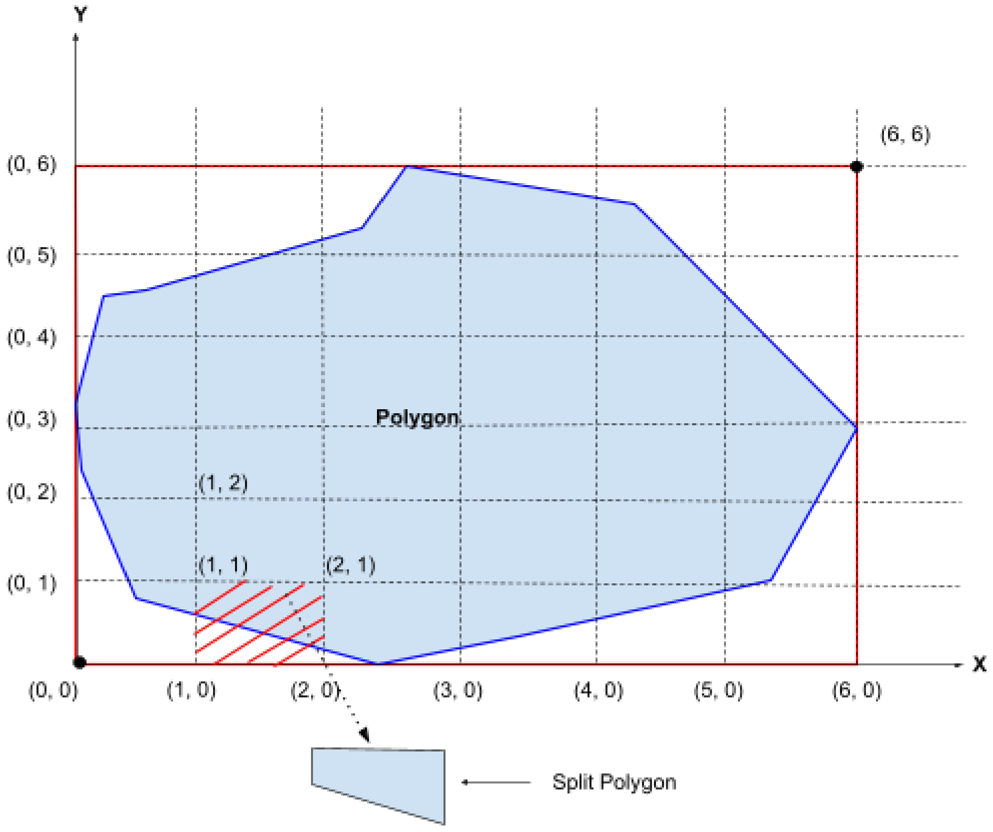

Figure (G)

Code

```
import java.util.ArrayList;
import java.util.List;
import com.vividsolutions.jts.geom.Envelope;
import com.vividsolutions.jts.geom.Geometry;
import com.vividsolutions.jts.geom.GeometryCollection;
import com.vividsolutions.jts.geom.GeometryFactory;
import com.vividsolutions.jts.geom.LineString;
import com.vividsolutions.jts.geom.MultiPolygon;
import com.vividsolutions.jts.geom.Point;
import com.vividsolutions.jts.geom.Polygon;

public class SplitGeometryAlgorithm {
private static GeometryFactory geomFactory = new GeometryFactory();

public static List<Geometry> splitOriginalGeometry(Geometry geom, int grid){
      return splitOriginalGeometry(geom, grid, grid);
}
```

```
public static List<Geometry> splitOriginalGeometry(Geometry geom, int gridX,
int gridY){

      final Envelope envelope = geom.getEnvelopeInternal();
      double minX = envelope.getMinX();
      double maxX = envelope.getMaxX();
      double minY = envelope.getMinY();
      double maxY = envelope.getMaxY();

      double grid_x = ((maxX - minX) / gridX);
      double grid_y = ((maxY - minY) / gridY);

      List<Geometry> list = new ArrayList<>();


      boolean otherGeomFlag = false;

      for(int i = 0; i < gridX; i++){
      for(int j = 0; j < gridY; j++){

            double _xmin = minX + (i * grid_x);
            double _ymin = minY + (j * grid_y);

            double _xmax = _xmin + grid_x;
            double _ymax = _ymin + grid_y;

            Envelope env = new Envelope(_xmin, _xmax, _ymin, _ymax);

            Geometry splitGeom = null;
            try{
                  splitGeom =
geomFactory.toGeometry(env).intersection(geom);
            }catch(com.vividsolutions.jts.geom.TopologyException ex){
                  U.log(ex);
                  otherGeomFlag = true;
                  break;
            }

            if(!splitGeom.isEmpty()){

                  if(splitGeom instanceof Polygon){
                        splitGeom = convertPolygonToMultiPolygon(splitGeom);
                  }
                  if(splitGeom instanceof MultiPolygon)
                        list.add(splitGeom);

                  else if(splitGeom instanceof GeometryCollection){
                        Geometry collectionGeom =
convertGeometryCollectionToMultiPolygon(splitGeom);
```

```
                        if(collectionGeom != null)
                              list.add(collectionGeom);
                        else{
                              otherGeomFlag = true;
                              break;
                        }
                  }
                  else{
                        otherGeomFlag = true;
                        break;
                  }
            }
      }//eof for inner

      if(otherGeomFlag)break;
      }//eof for outer

      if(otherGeomFlag) list.clear();
      return list;
}

private static Geometry convertGeometryCollectionToMultiPolygon(Geometry
input){
      List<Geometry> list = new ArrayList<>();

      int countLine = 0;
      int countPoint = 0;
      for(int i = 0; i< input.getNumGeometries(); i++){
            if(input.getGeometryN(i) instanceof Polygon ||
input.getGeometryN(i) instanceof MultiPolygon)
                  list.add(input.getGeometryN(i));

            else if(input.getGeometryN(i) instanceof LineString){
                  countLine++;
            }
            else if(input.getGeometryN(i) instanceof Point){
                  countPoint++;
            }
      }
      /*
       * If more than one LineString or Point contains in
GeometryCollection, then we don't have confidence whether we will get the
proper polygon or not.
       */
      if(countLine > 1 || countPoint > 1){
            return null;
      }

      MultiPolygon result = null;
      if(list.size() > 1){
          result = (MultiPolygon) geomFactory.buildGeometry(list);
          result = (MultiPolygon) result.union();

          return result;
      }
      else if(list.size() == 1){
          return convertPolygonToMultiPolygon(list.get(0));
      }
      return null;
}

private static Geometry convertPolygonToMultiPolygon(Geometry input){
      Polygon[] polys = new Polygon[1];
      polys[0] = (Polygon) input;
      return geomFactory.createMultiPolygon(polys);
}
}
```

Algorithm Outputs

Polygon without hole

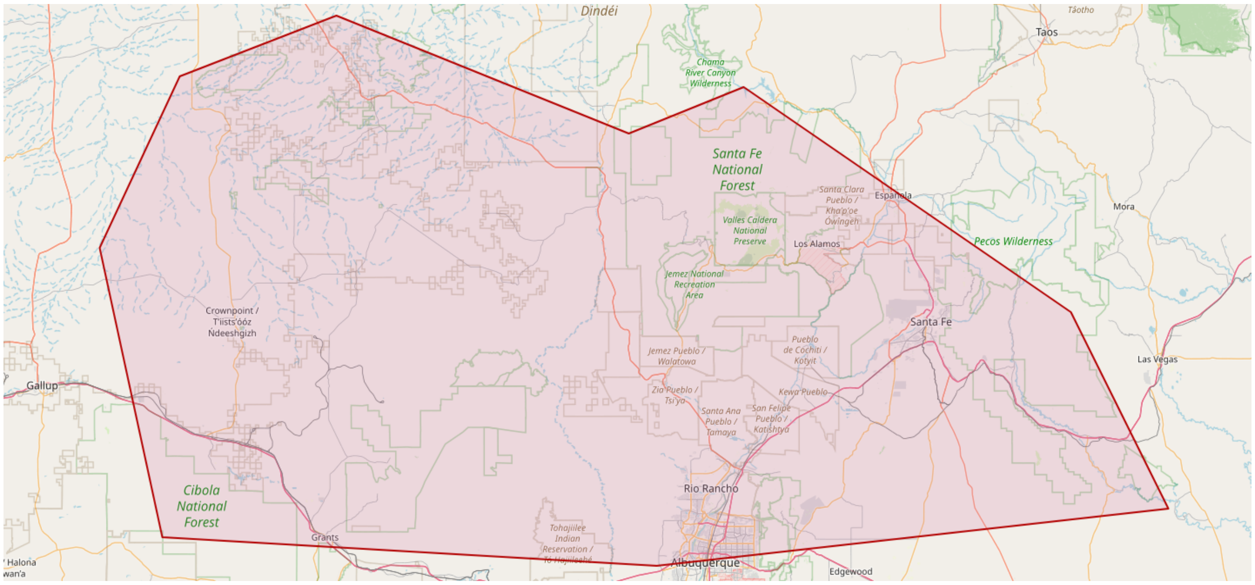
Dindéi
Taos
Chama River Canyon Wilderness
Santa Fe National Forest
Valles Caldera National Preserve
Santa Clara Pueblo / Kha'p'oe Ówingeh
Española
Mora
Los Alamos
Pecos Wilderness
Jemez National Recreation Area
Crownpoint / T'iists'óóz Ndeeshgizh
Santa Fe
Las Vegas
Gallup
Jemez Pueblo / Walatowa
Pueblo de Cochiti / Kotyit
Zia Pueblo / Tsi'ya
Kewa Pueblo
Santa Ana Pueblo / Tamaya
San Felipe Pueblo / Katishtya
Rio Rancho
Cibola National Forest
Grants
Tohajiilee Indian Reservation
Albuquerque
Edgewood
Halona

Fig. Original geometry

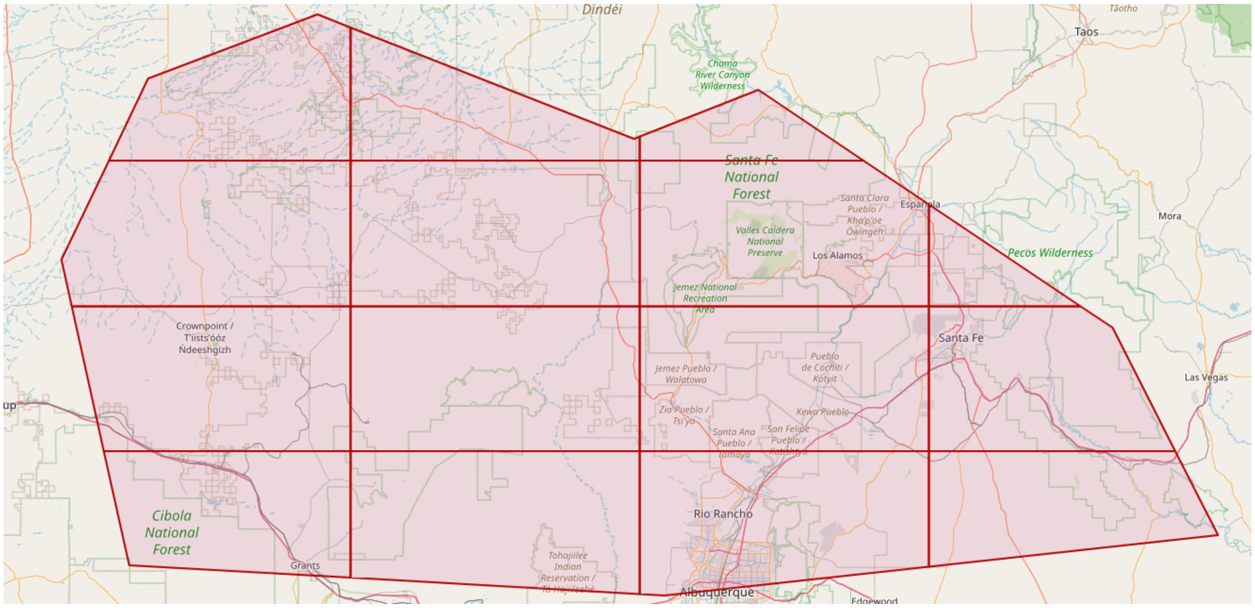
Dinděi
Táotho
Taos
Chama River Canyon Wilderness
Santa Fe National Forest
Santa Clara Pueblo / Kha'p'oe Owingeh
Española
Mora
Valles Caldera National Preserve
Los Alamos
Pecos Wilderness
Jemez National Recreation Area
Crownpoint / T'iists'óóz Ndeeshgizh
Santa Fe
Pueblo de Cochiti / Kotyit
Las Vegas
Jemez Pueblo / Walatowa
Zia Pueblo / Tsi'ya
Kewa Pueblo
San Felipe Pueblo / Katishtya
Santa Ana Pueblo / Tamaya
Rio Rancho
Cibola National Forest
Grants
Tohajiilee Indian Reservation / Tó Hajiileehé
Albuquerque
Edgewood

Fig. Splitted geometries

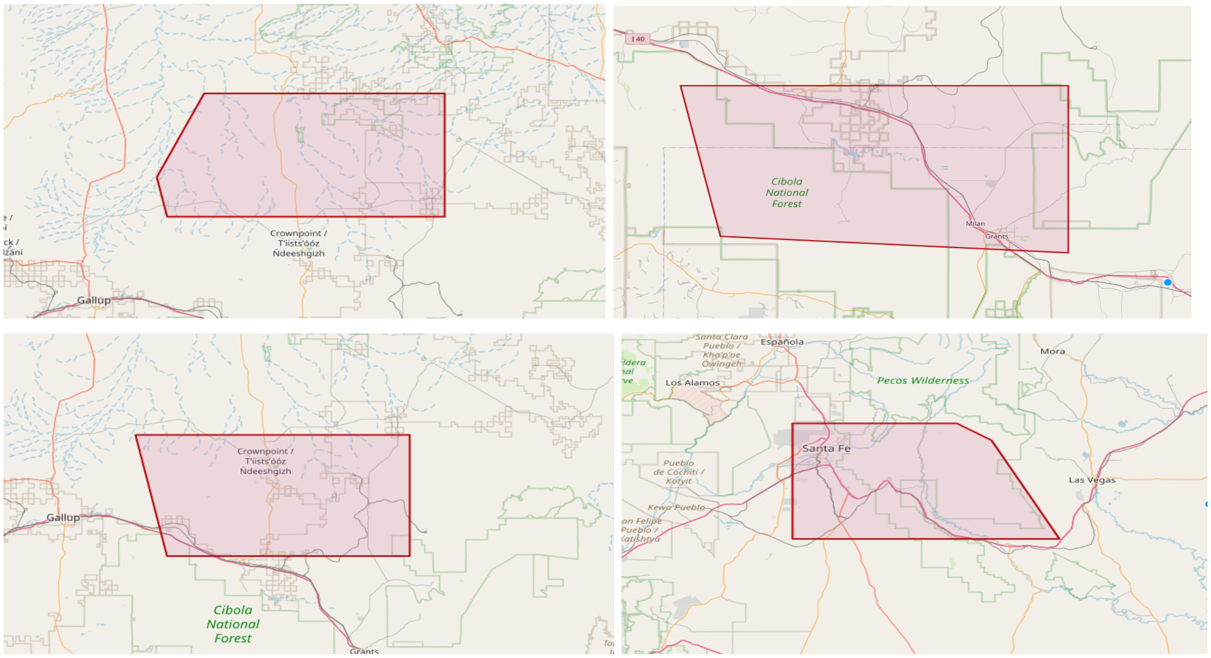

Fig. Examples of each split geometry

Polygon with holes

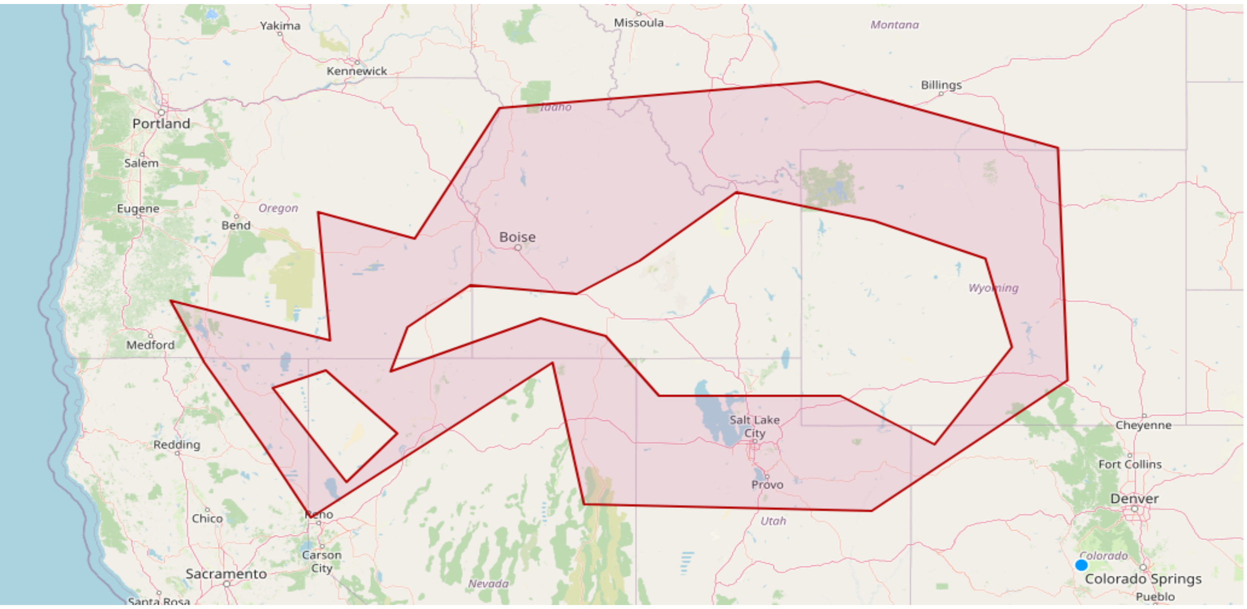

Fig. Original geometry

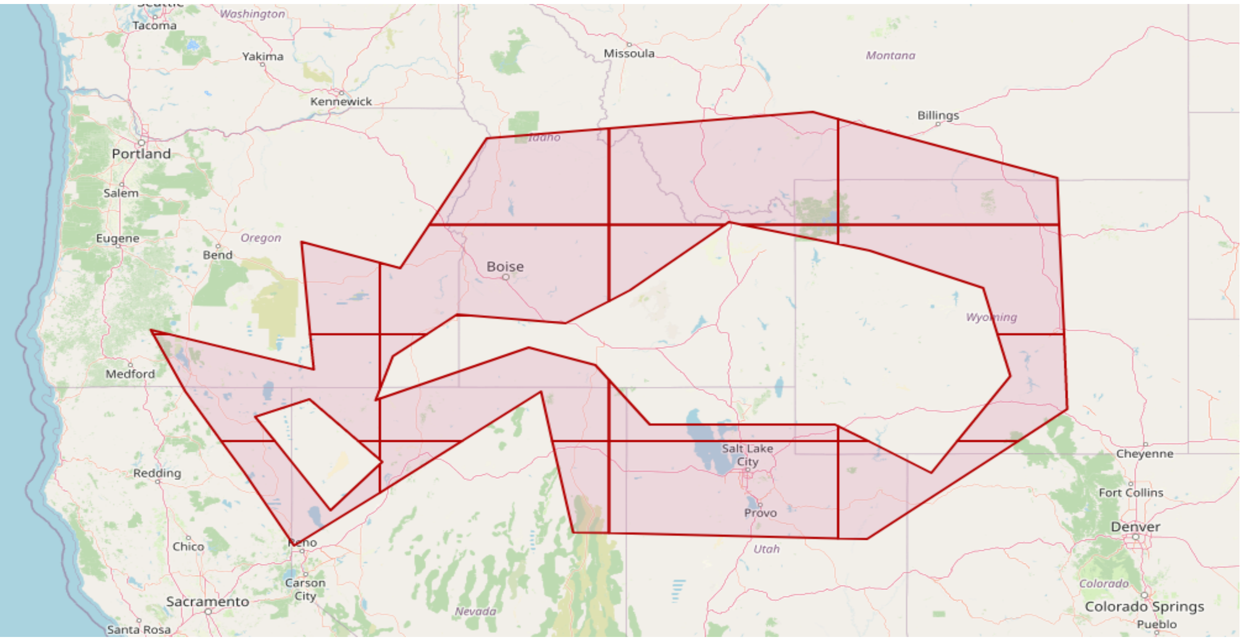

Fig. Splitted geometries

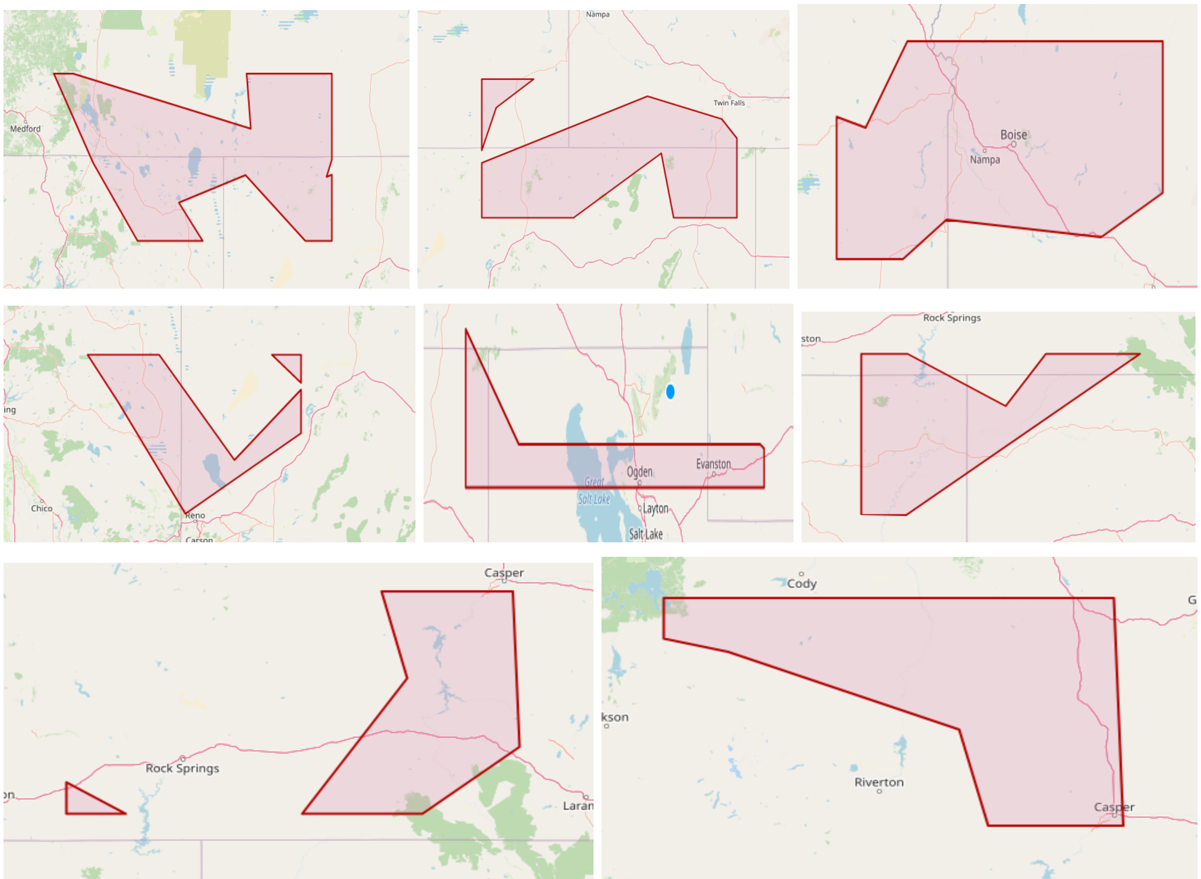

Fig. Examples of each split geometry