# Lines & Polygons
# A Love Story

2023 OKSCAUG Edmond Meeting

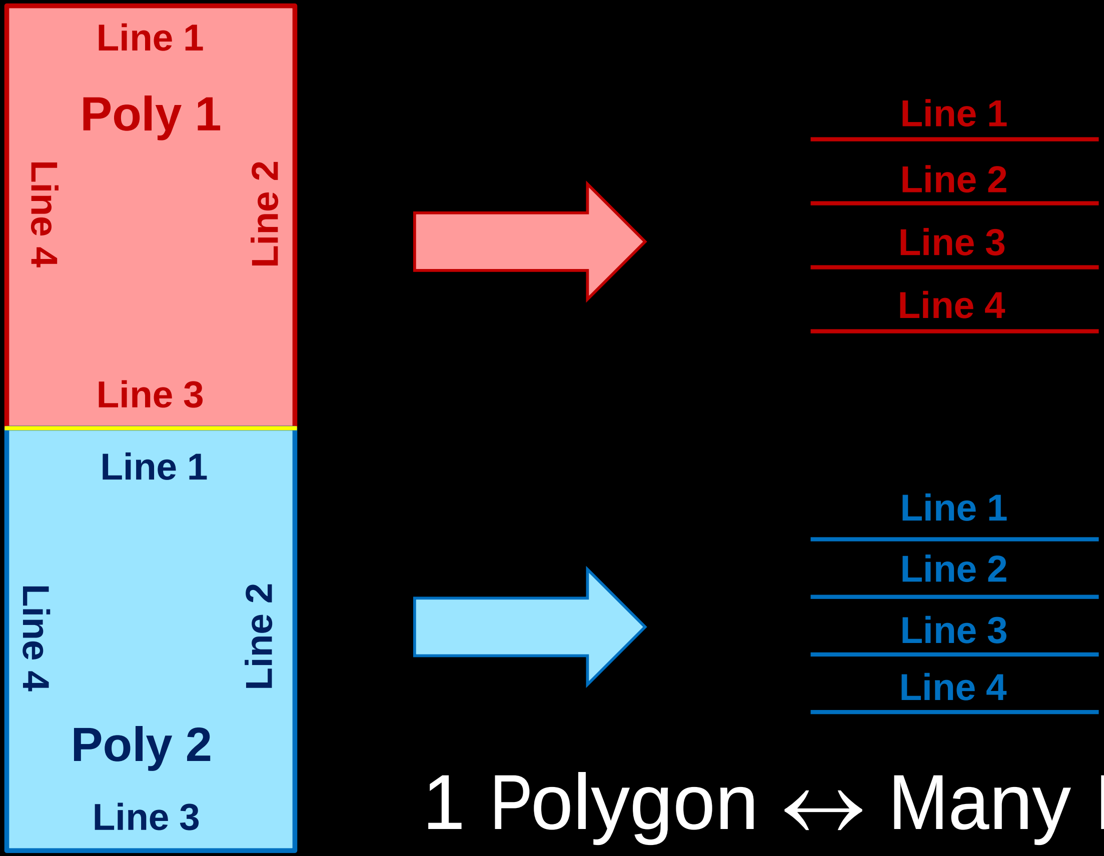
Joel A. Foster

GIS Coordinator

Canadian County Assessor's Office

# Our story begins...

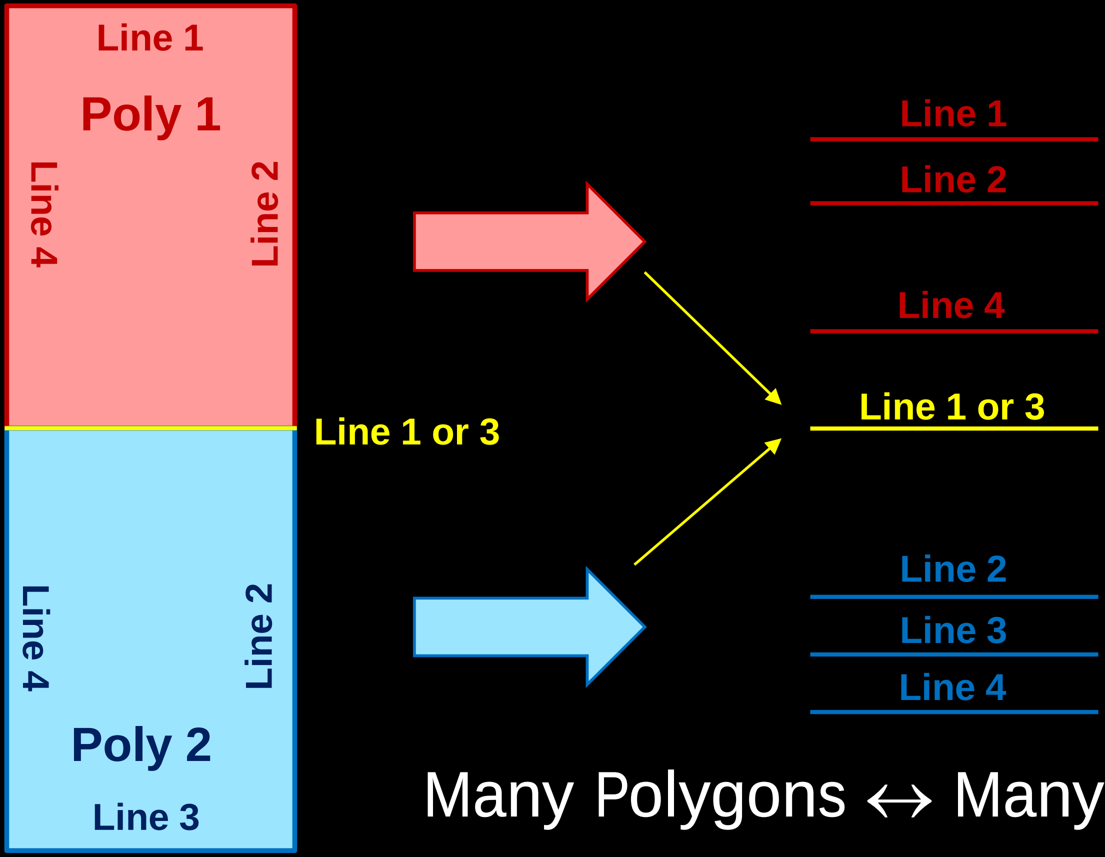
## ArcMap Data Model



1 Polygon ↔ Many Lines

# Relationship Lost...

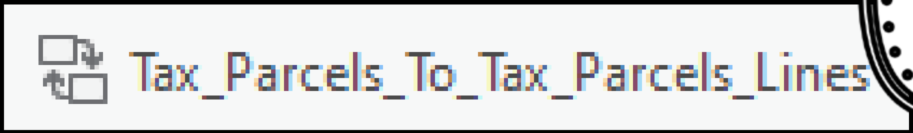
## ArcGIS Pro Data Model

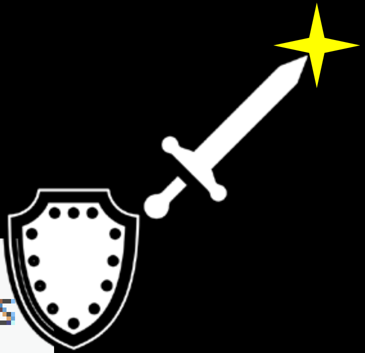


Many Polygons ↔ Many Lines

# Relationship Found?

Many to Many Relationship Class

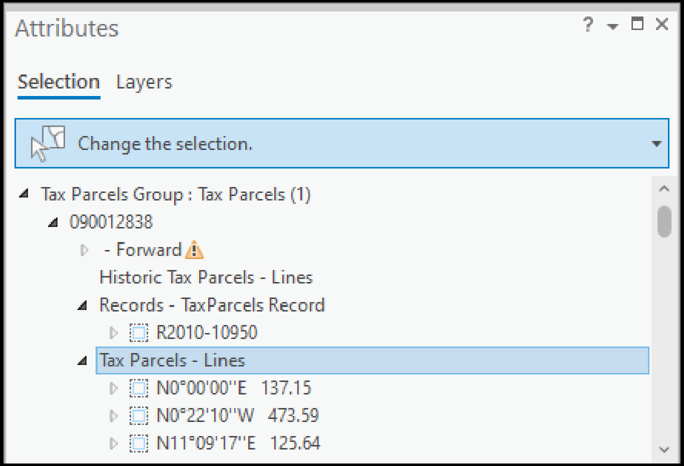

Tax_Parcels_To_Tax_Parcels_Lines



# Relationship Found?

Relationship Classes Allow...

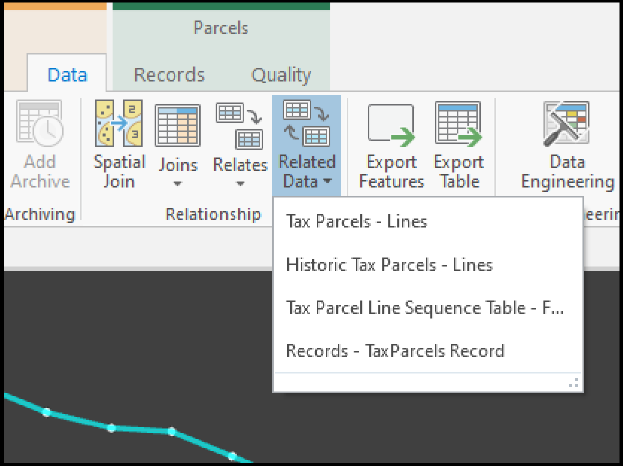
- Show related features in Attribute Pane
- Select related features on map
- Additional attributes if needed


Attributes

Selection Layers

Change the selection.

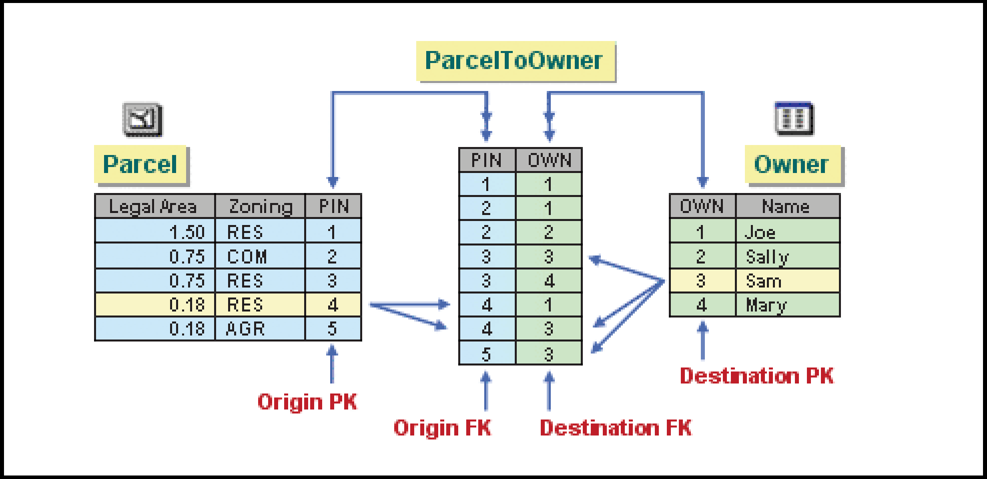
- Tax Parcels Group : Tax Parcels (1)
  - 090012838
    - - Forward
    - Historic Tax Parcels - Lines
    - Records - TaxParcels Record
      - R2010-10950
    - Tax Parcels - Lines
      - N0°00'00"E 137.15
      - N0°22'10"W 473.59
      - N11°09'17"E 125.64



Tax Parcels - Lines
Historic Tax Parcels - Lines
Tax Parcel Line Sequence Table - F...
Records - TaxParcels Record

# Many to Many Relationship Class

Many to Many requires an intermediate table



Parcel

| Legal Area | Zoning | PIN |
|---|---|---|
| 1.50 | RES | 1 |
| 0.75 | COM | 2 |
| 0.75 | RES | 3 |
| 0.18 | RES | 4 |
| 0.18 | AGR | 5 |

| PIN | OWN |
|---|---|
| 1 | 1 |
| 2 | 1 |
| 2 | 2 |
| 3 | 3 |
| 3 | 4 |
| 4 | 1 |
| 4 | 3 |
| 5 | 3 |

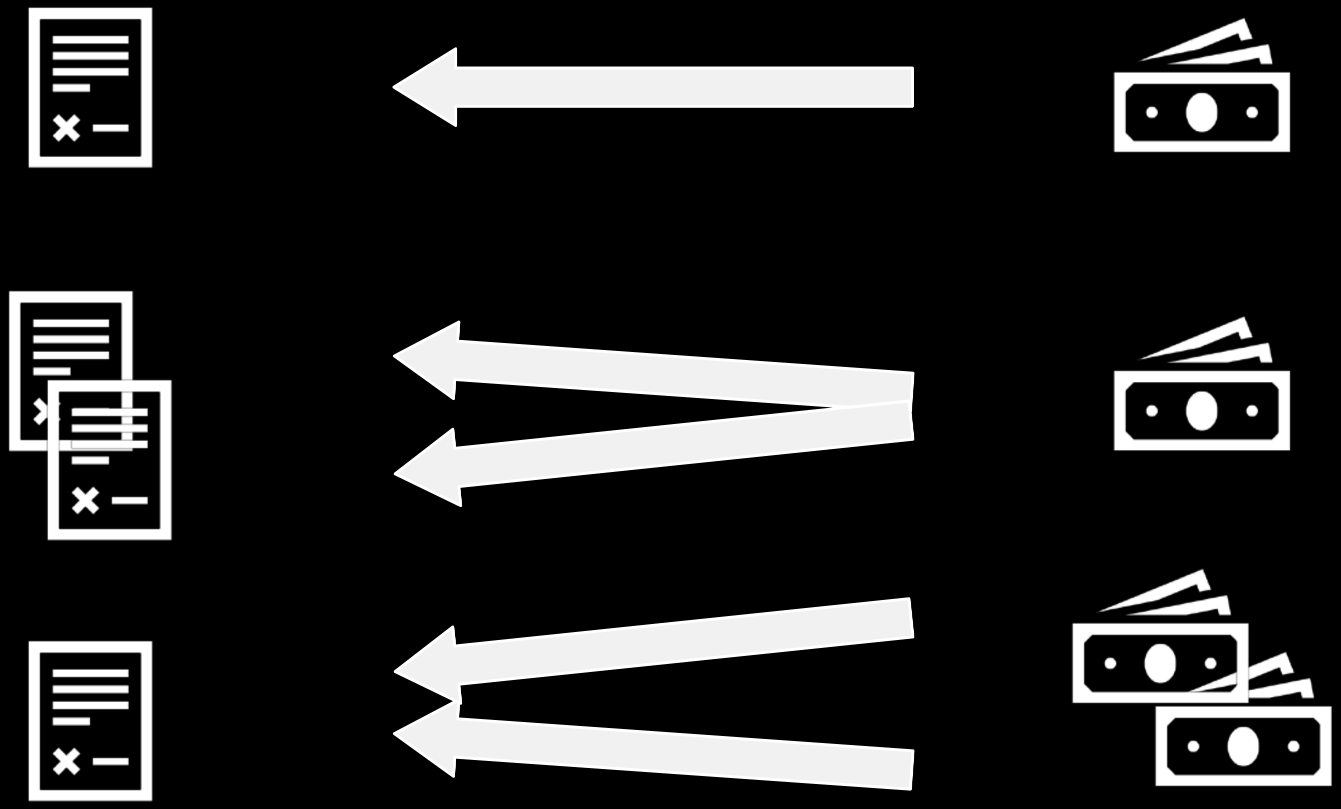
Owner

| OWN | Name |
|---|---|
| 1 | Joe |
| 2 | Sally |
| 3 | Sam |
| 4 | Mary |

# Many to Many Relationships

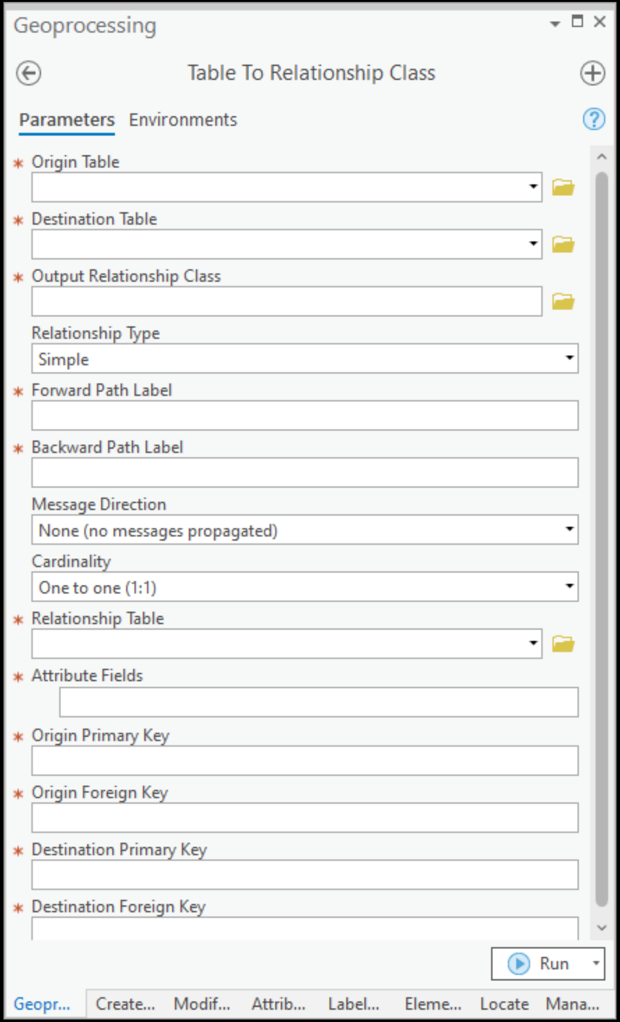
Example-

Invoices and Payments



# Populating the Relationship Class Table

- Table to Relationship Class Geoprocessing tool



Geoprocessing

Table To Relationship Class

Parameters Environments

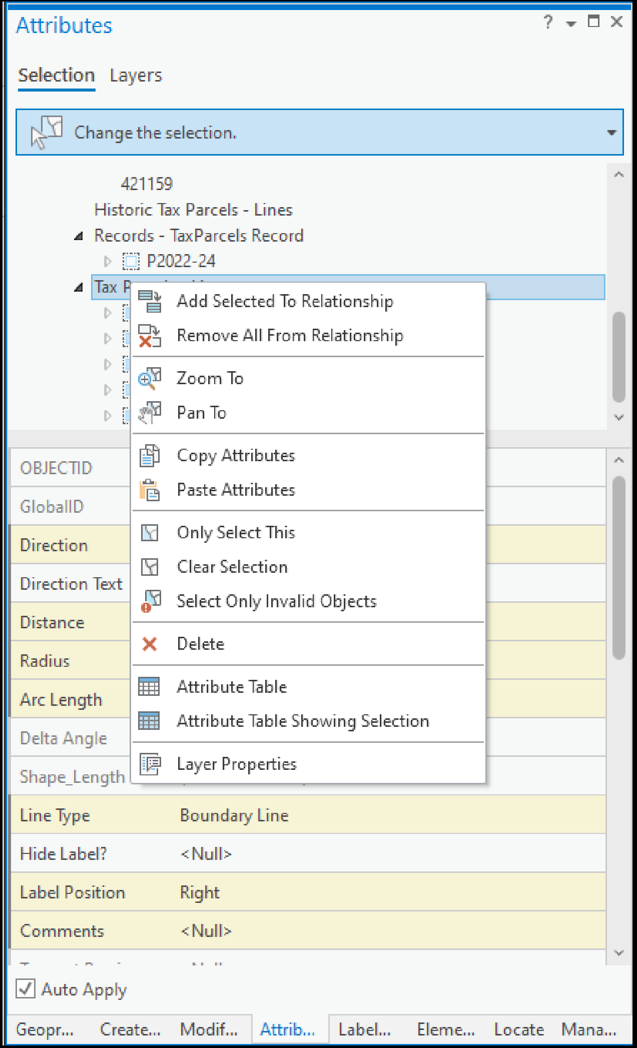
- * Origin Table
- * Destination Table
- * Output Relationship Class
- Relationship Type: Simple
- * Forward Path Label
- * Backward Path Label
- Message Direction: None (no messages propagated)
- Cardinality: One to one (1:1)
- * Relationship Table
- * Attribute Fields
- * Origin Primary Key
- * Origin Foreign Key
- * Destination Primary Key
- * Destination Foreign Key

Run
Geopr... Create... Modif... Attrib... Label... Eleme... Locate Mana...

# Populating the Relationship Class Table

- Manual Editing using Attributes Pane

1. Select polygon and lines you want to relate
2. Choose “Add Selected To Relationship” in Attributes Pane

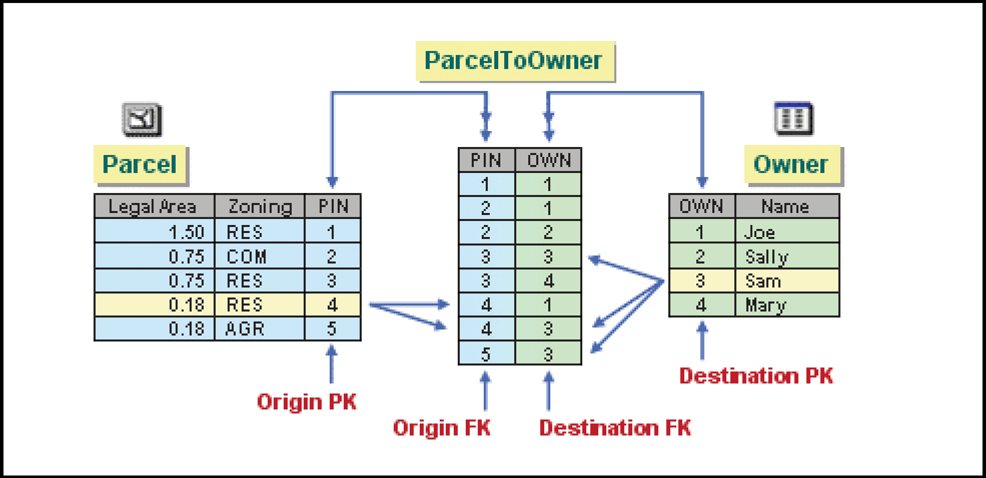


421159
Historic Tax Parcels - Lines
Records - TaxParcels Record
P2022-24
Tax P

| | |
|---|---|
| OBJECTID | |
| GlobalID | |
| Direction | |
| Direction Text | |
| Distance | |
| Radius | |
| Arc Length | |
| Delta Angle | |
| Shape_Length | |
| Line Type | Boundary Line |
| Hide Label? | <Null> |
| Label Position | Right |
| Comments | <Null> |

# Relationship not Quite Found...

## Populating the intermediate table



Parcel

| Legal Area | Zoning | PIN |
|---|---|---|
| 1.50 | RES | 1 |
| 0.75 | COM | 2 |
| 0.75 | RES | 3 |
| 0.18 | RES | 4 |
| 0.18 | AGR | 5 |

ParcelToOwner

| PIN | OWN |
|---|---|
| 1 | 1 |
| 2 | 1 |
| 2 | 2 |
| 3 | 3 |
| 3 | 4 |
| 4 | 1 |
| 4 | 3 |
| 5 | 3 |

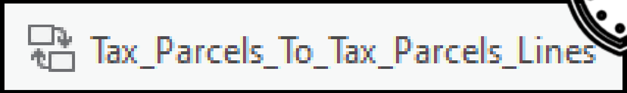
Owner

| OWN | Name |
|---|---|
| 1 | Joe |
| 2 | Sally |
| 3 | Sam |
| 4 | Mary |

When the intermediate table is created, only the fields are generated for you. ArcGIS does not know which origin objects are associated with which destination objects, so you must manually create the rows in the table. Populating this table is the most time-consuming part of setting up the relationship.

# Another Way

Populating the intermediate table...programmatically

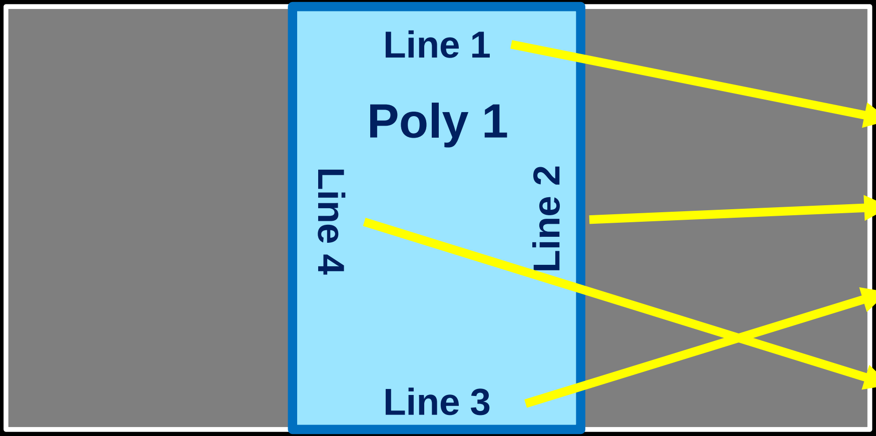


Tax_Parcels_To_Tax_Parcels_Lines

python™

# Python Table Population

1. Find the lines that define the boundaries of the polygon
2. Insert a row in table for each line with the line's unique ID and the poly's unique ID

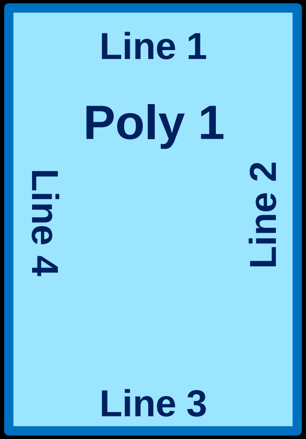


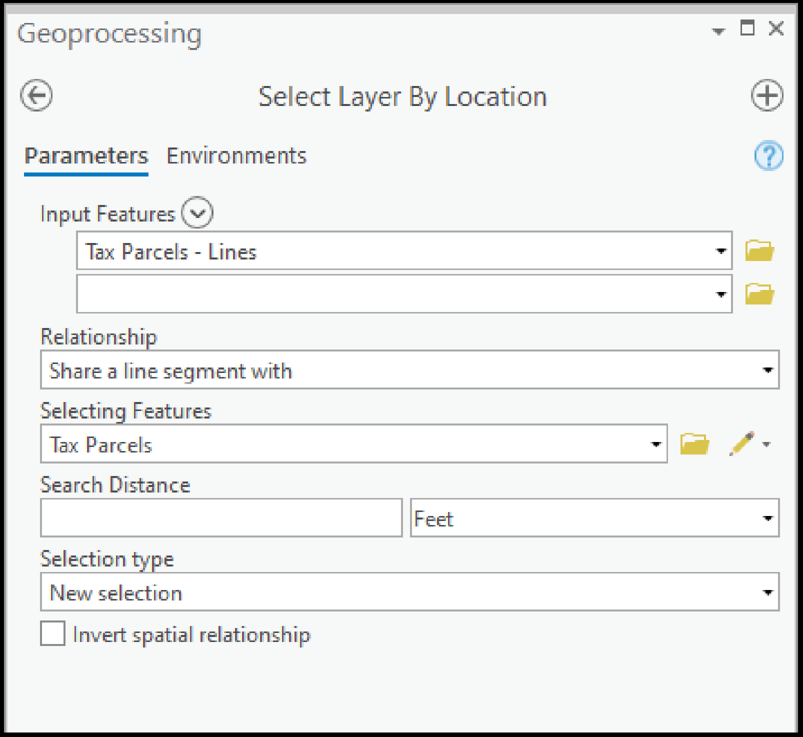
| Poly ID | Line ID |
|---|---|
| Poly 1 | Line 1 |
| Poly 1 | Line 2 |
| Poly 1 | Line 3 |
| Poly 1 | Line 4 |

# Find the Lines

## Select By Location

- Select lines that share a boundary with the polygon





Select Layer By Location

Input Features: Tax Parcels - Lines

Relationship: Share a line segment with

Selecting Features: Tax Parcels

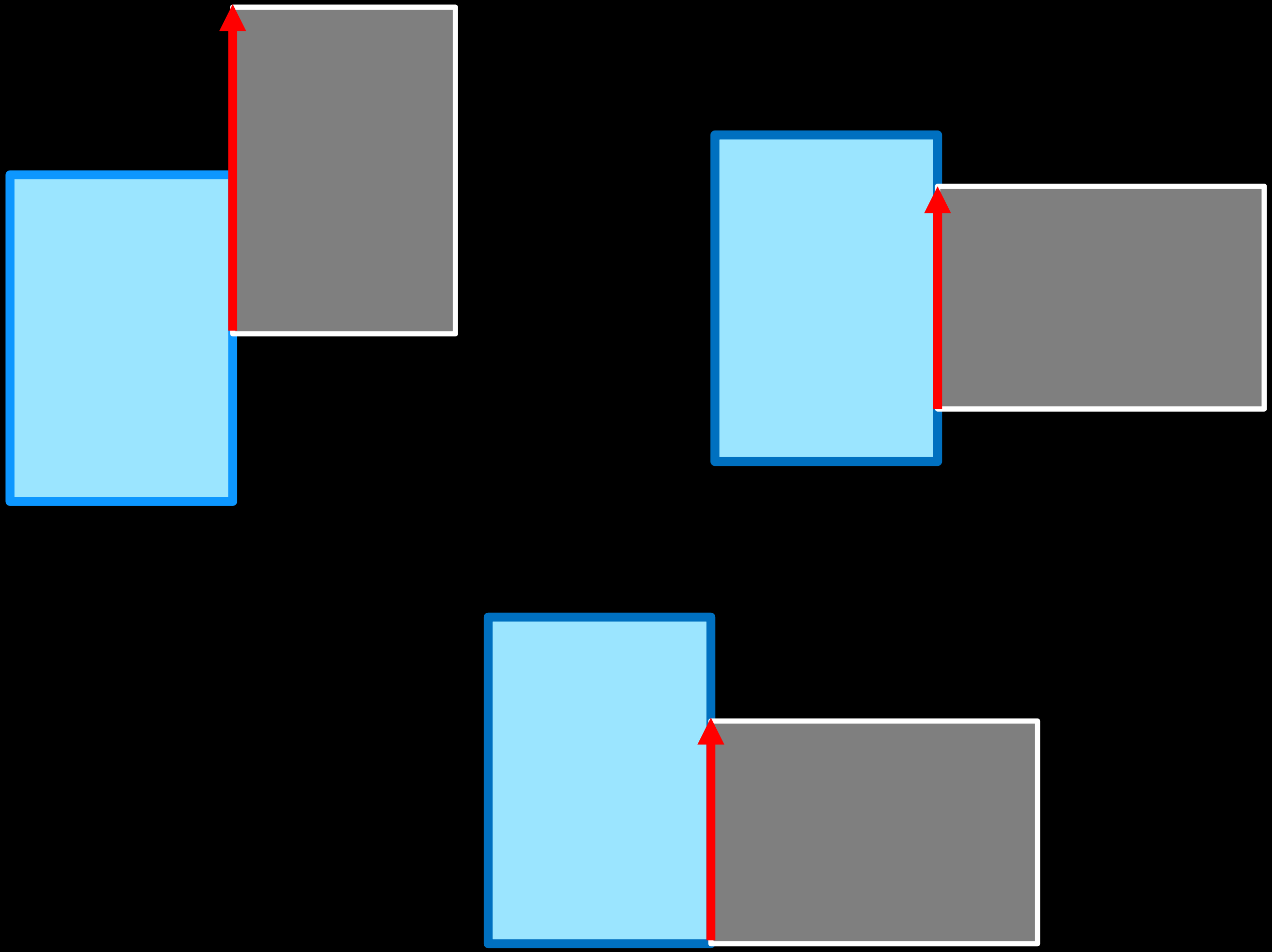
Search Distance: Feet

Selection type: New selection

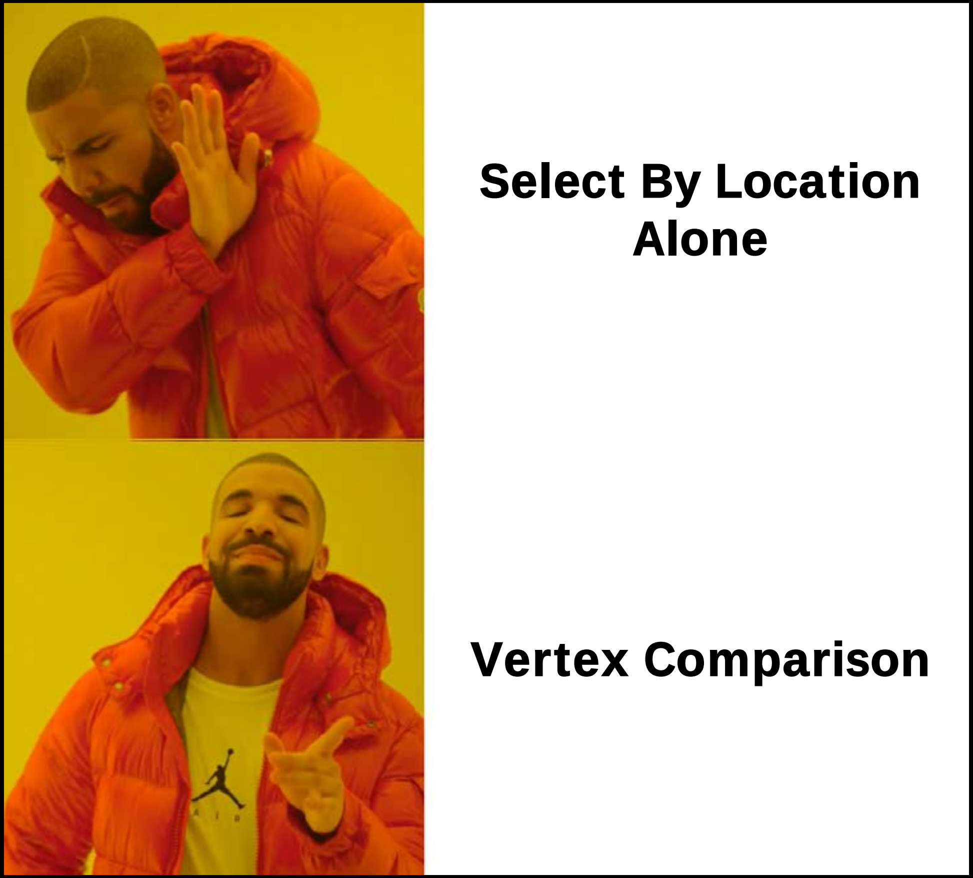
☐ Invert spatial relationship

# NOT SO FAST!!!

S19°08'33"W
314.14
N19°10'55"E
411.07
090125484
14
N89°57'00"
S89°57'00"W
995.75
N89°57'00"E
892.75
N42°33'06"E
152.94
N24°33'22"W
256.33
090125484
15
N21°53'55"W
191.57
S21°57'25"E
188.68
26
S0°51'29"E
N88°01'13"W
497.86
S88°01'13"E
497.88
S5°28'01"E
515.87
N5°24'31"W
299.18
N0°08'47"E
619.24

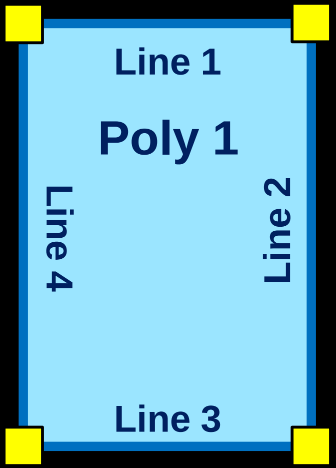


# Find the Lines

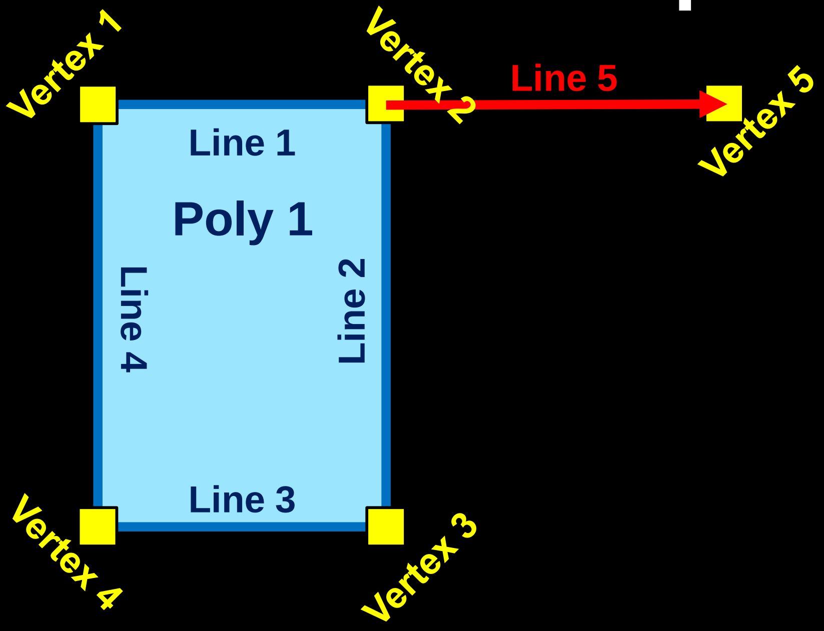


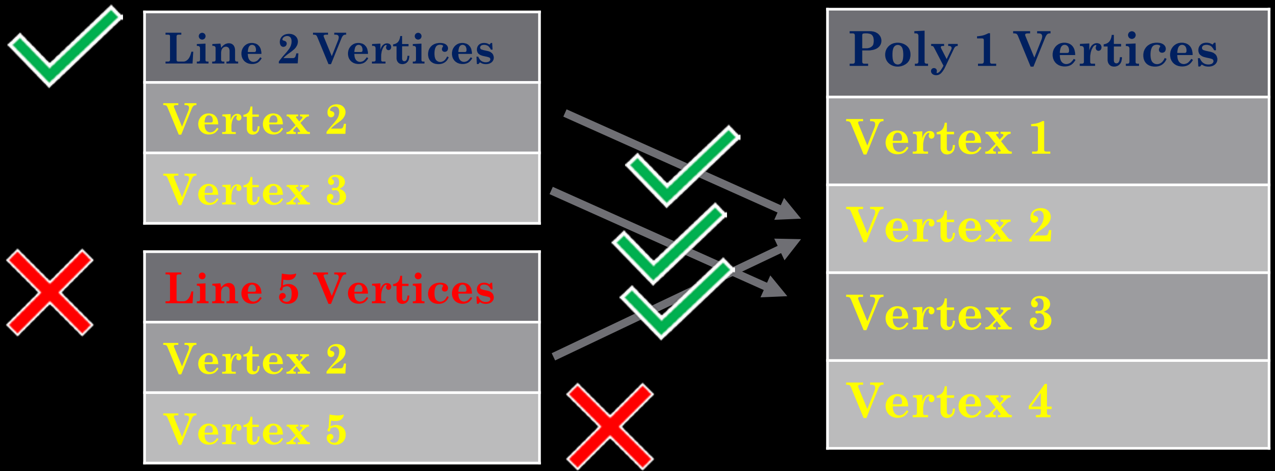
# Python Vertex Comparison

1. Find lines that might define the boundary of a polygon
2. Check the vertices of the line against the vertices of the polygon
3. If all the vertices in the line have a matching vertex in the polygon, the line forms part of the boundary



# Python Vertex Comparison





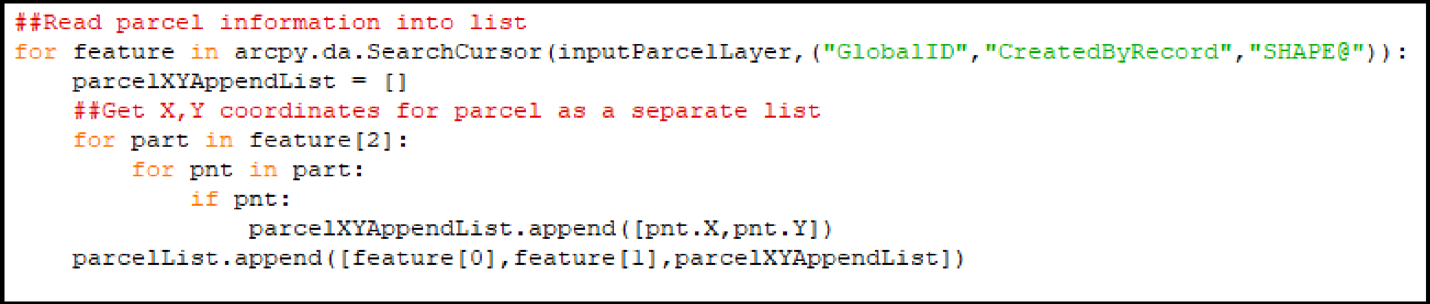
| Line 2 Vertices |
|---|
| Vertex 2 |
| Vertex 3 |

| Line 5 Vertices |
|---|
| Vertex 2 |
| Vertex 5 |

| Poly 1 Vertices |
|---|
| Vertex 1 |
| Vertex 2 |
| Vertex 3 |
| Vertex 4 |

# Python Vertex Comparison

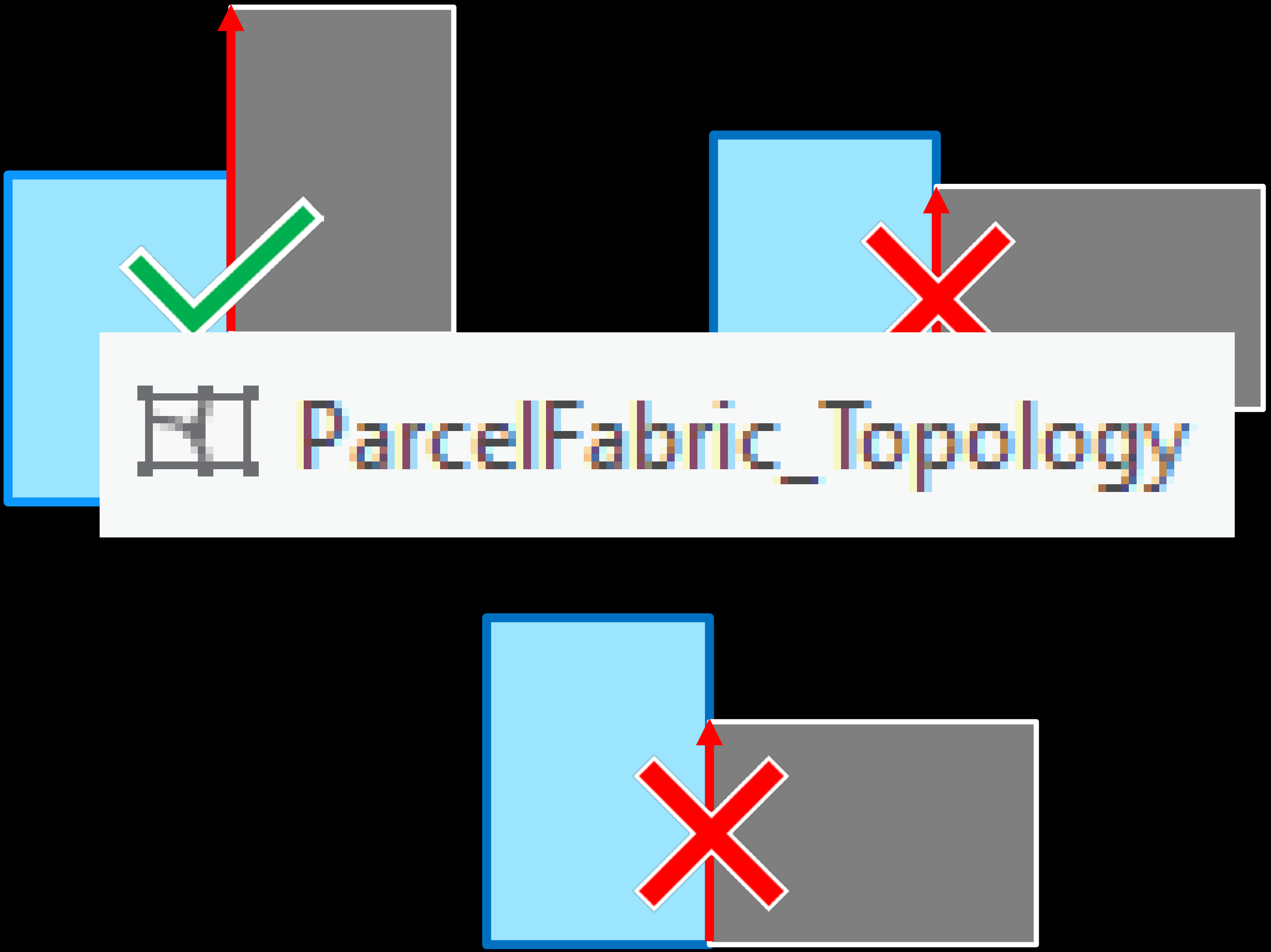
1. Bring in the polygon or line’s Shape field with a cursor
2. Read the points (for each feature part) into a list



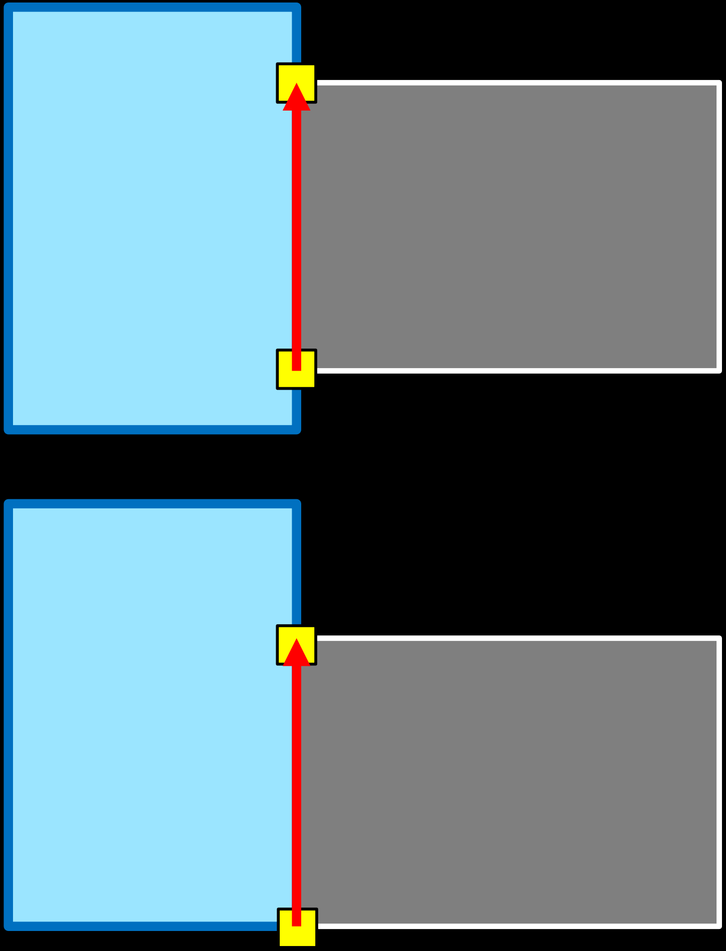
```
##Read parcel information into list
for feature in arcpy.da.SearchCursor(inputParcelLayer,("GlobalID","CreatedByRecord","SHAPE@")):
    parcelXYAppendList = []
    ##Get X,Y coordinates for parcel as a separate list
    for part in feature[2]:
        for pnt in part:
            if pnt:
                parcelXYAppendList.append([pnt.X,pnt.Y])
    parcelList.append([feature[0],feature[1],parcelXYAppendList])
```

090125484
15
N89°57'00"E
892.75
S89°57'00"W
995.75
N88°01'13"W
497.86
S88°01'13"E
497.88
26

# NOT SO FAST!!!

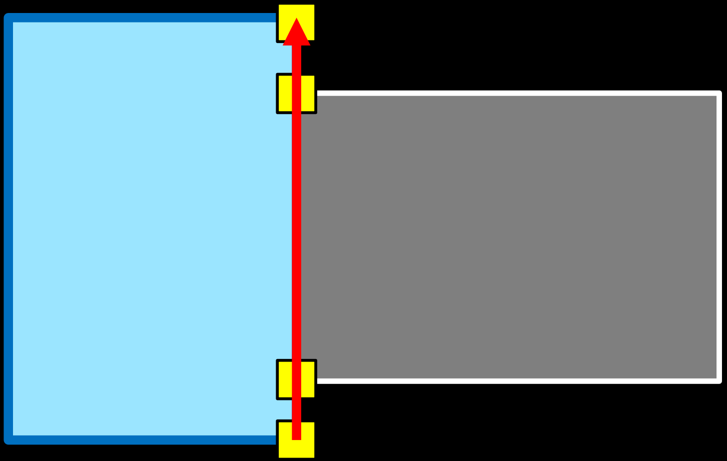
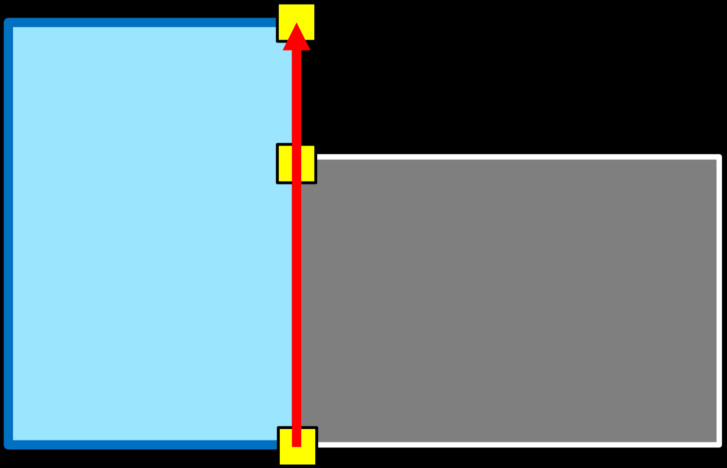

ParcelFabric_Topology

# Topology Induced Difficulty



Topology adds vertices to any participating features that are within the tolerance distance of each other

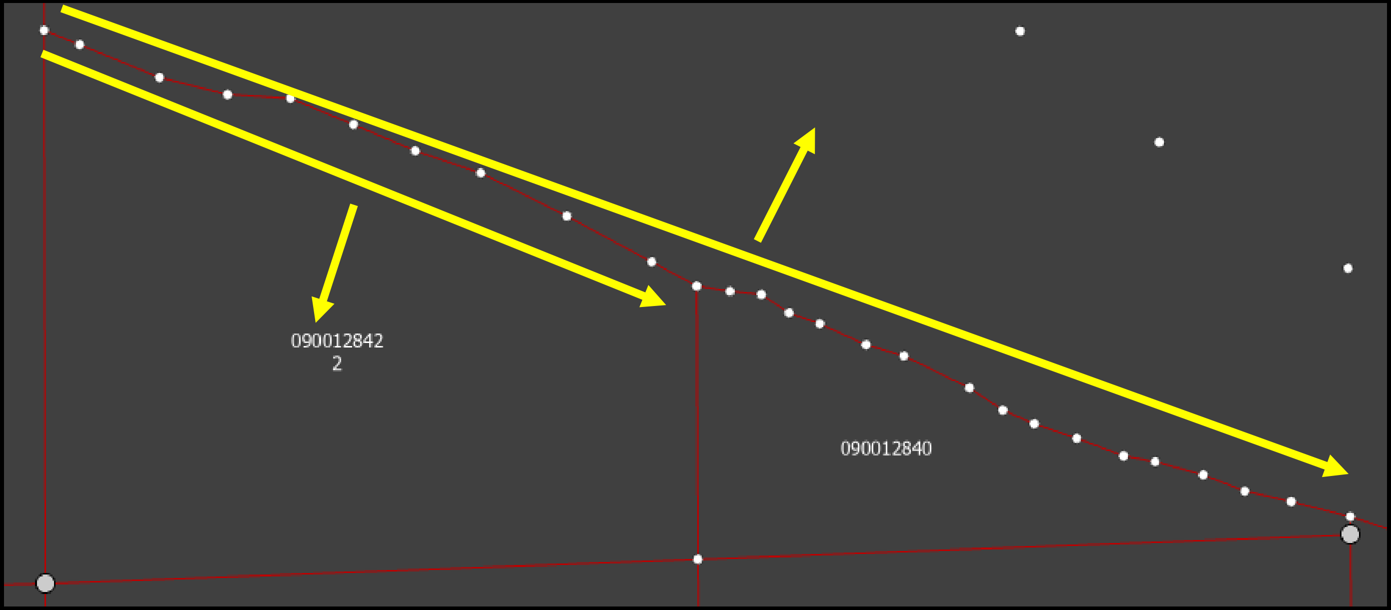
# Remove the Imposters

1. Check any line that has more than two vertices for all overlapping lines
2. Keep the line that has the most vertices because it covers more of the boundary

# Remove the Imposters

Works for valid >2 point lines too



# Lines & Polygons Reunited

1. Many-to-Many Relationship Class handles keeping the two feature classes related, with additional attributes if needed
2. Python speeds up & partially automates populating the relationship class making maintenance easier

# Final Thoughts

- Tool is too slow, about 10 seconds per polygon
- Select by Location using the "Within" relationship alone does work but vertex comparison seems to be faster
- Working on a way to automate populating the line sequence attribute

# Lines & Polygons A Love Story

2023 OKSCAUG Edmond Meeting

Joel Foster

GIS Coordinator

Canadian County Assessor's Office

(405) 295-6331

foster@canadiancounty.org