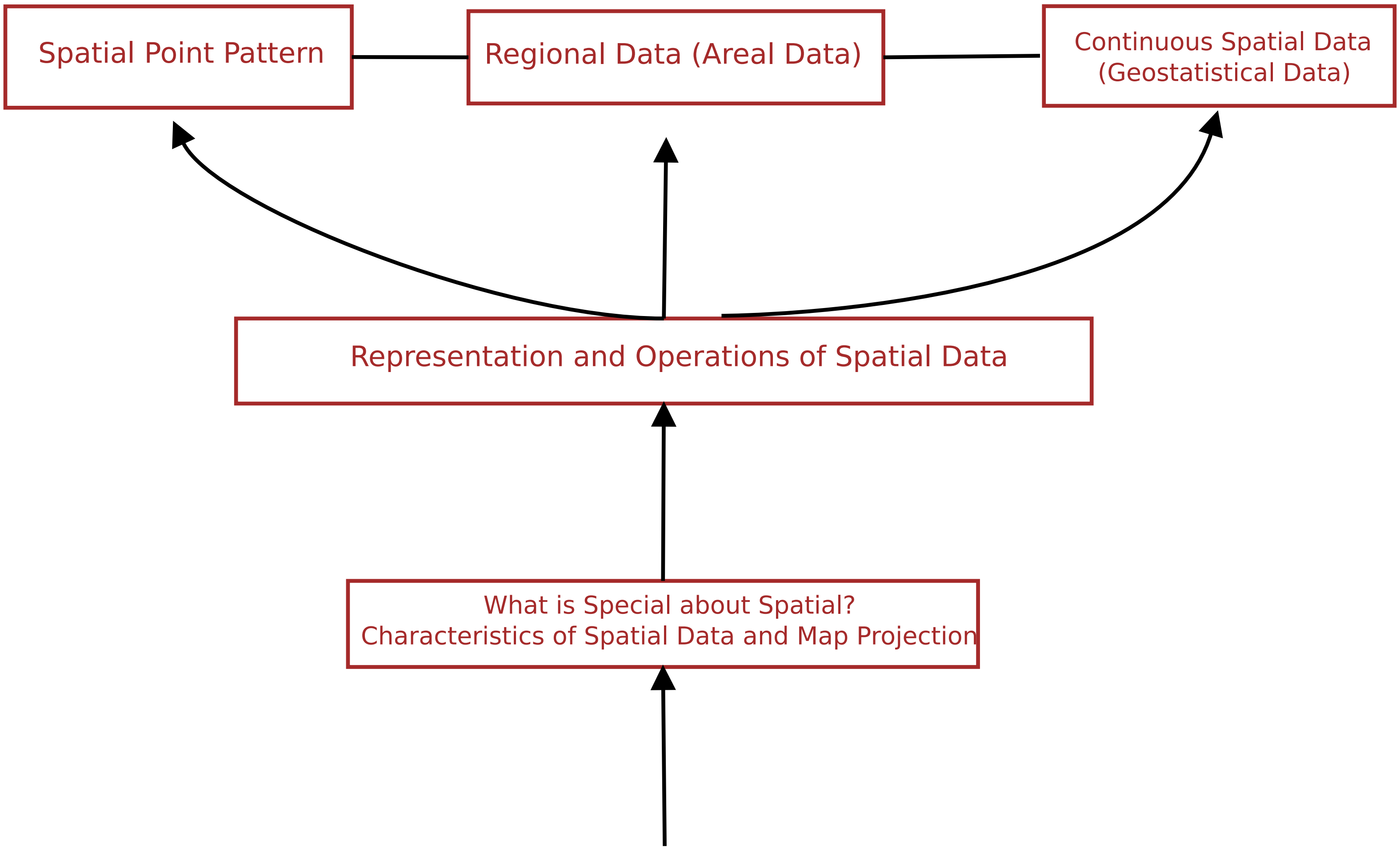
Spatial Analysis and Modeling (GIST 4302/5302)

Guofeng Cao
Department of Geosciences
Texas Tech University

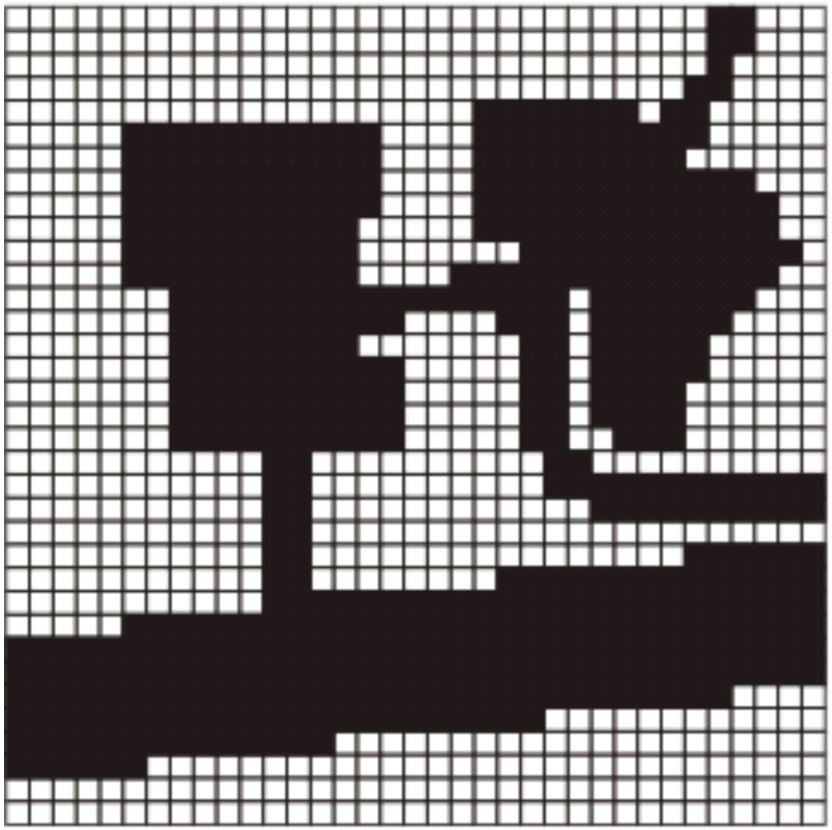
Class Outlines



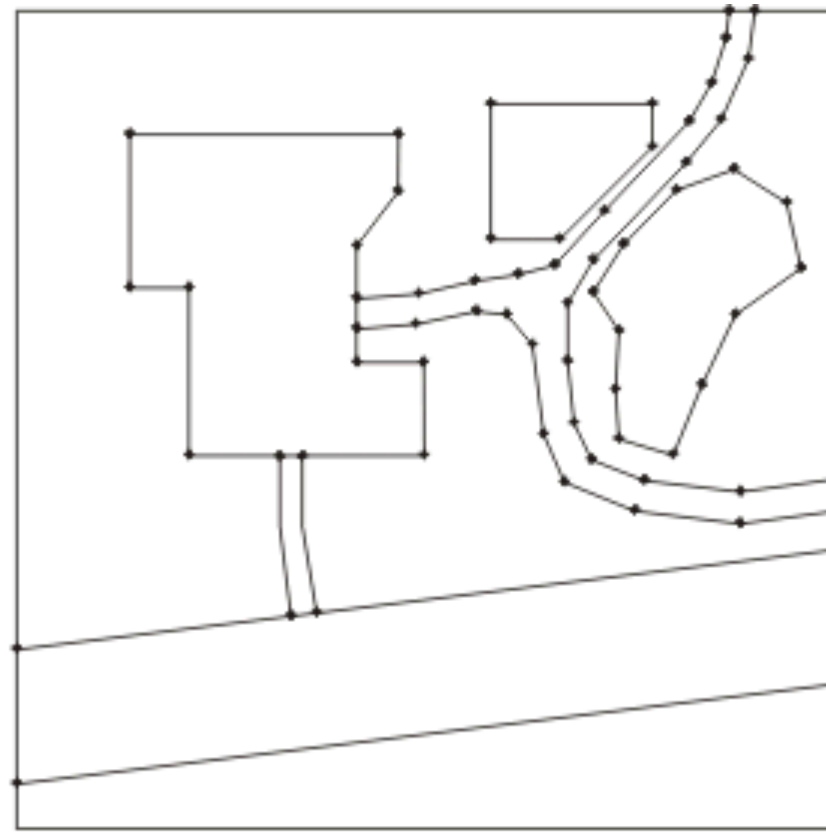
Representation of Spatial Data

Representation of Spatial Data Models

- ***Object-based model:*** treats the space as populated by discrete, identifiable entities each with a geospatial reference
  - Buildings or roads fit into this view
  - GIS Softwares: ArcGIS
- ***Field-based model:*** treats geographic information as collections of spatial distributions
  - Distribution may be formalized as a mathematical function from a spatial framework to an attribute domain
  - Patterns of topographic altitudes, rainfall, and temperature fit neatly into this view.
  - GIS Software: Grass



Raster



Vector

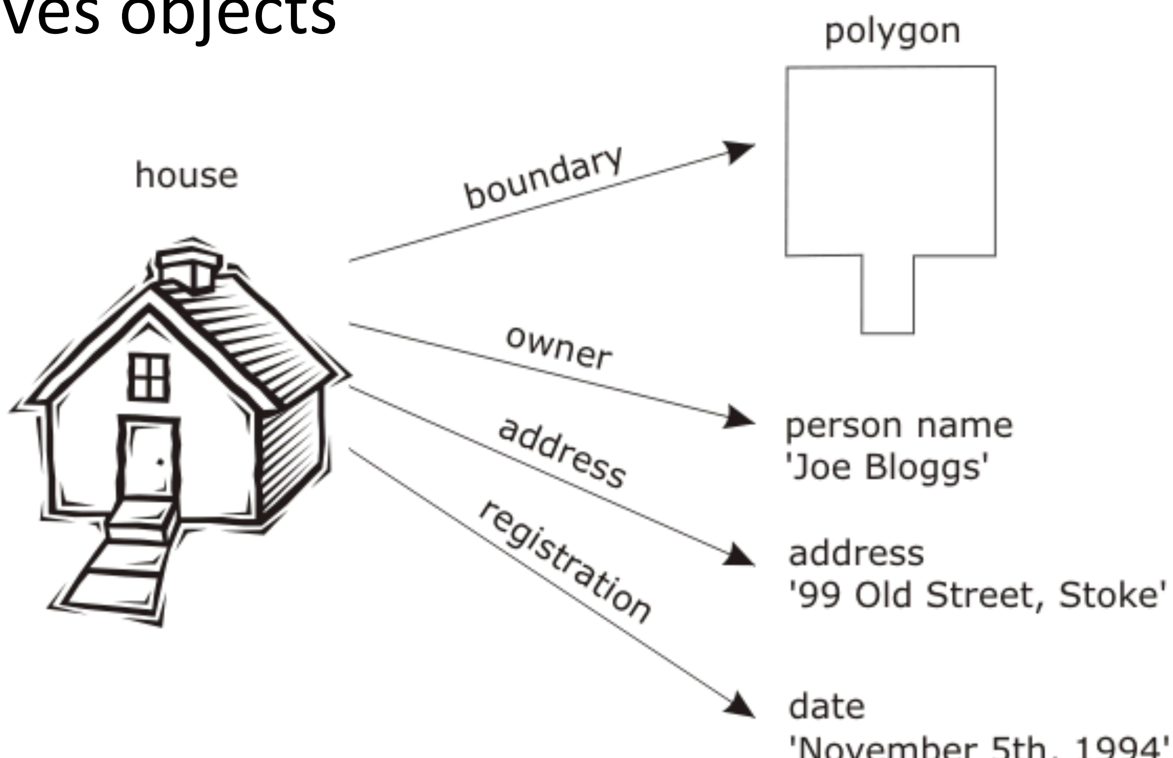
Object-based Approach

Entity

- Object-based models decompose an information space into objects or ***entities***
- An entity must be:
  - Identifiable
  - Relevant (be of interest)
  - Describable (have characteristics)
- The frame of spatial reference is provided by the entities themselves

Example: House object

Has several attributes, such as registration date, address, owner and boundary, which are themselves objects



Attribute Tables

3	Point	74	13	7.181452	70074	34.199444	-118.534722	-118.53472	34.19944
4	Point	75	13	6.076613	70075	34.066944	-117.751389	-117.75139	34.06694
5	Point	84	8	3.157258	70084	33.929167	-118.209722	-118.20972	33.92917
6	Point	85	11	5.201613	70085	34.015	-118.059722	-118.05972	34.015
7	Point	87	11	4.717742	70087	34.067222	-118.226389	-118.22639	34.06722
8	Point	88	15	6.532258	70088	34.083333	-118.106944	-118.10694	34.08333
9	Point	89	15	7.540323	70089	34.3875	-118.534722	-118.53472	34.3875
10	Point	91	10	4.891129	70091	34.050833	-118.454167	-118.45417	34.05083
11	Point	94	10	4.149194	70094	33.92357	-118.37085	-118.37085	33.92357
12	Point	96	13	7.266129	70096	34.69012	-118.1334	-118.1334	34.69012
13	Point	3176	9	4.100806	30176	33.820278	-117.9125	-117.9125	33.82028
14	Point	3177	10	4.737903	30177	33.926111	-117.951389	-117.95139	33.92611
15	Point	3186	11	4.600806	30186	33.6275	-117.690278	-117.69028	33.6275
16	Point	3195	8	4.358871	30195	33.67511	-117.9269	-117.9269	33.67511
17	Point	4137	13	8.241935	33137	33.85	-116.543056	-116.54306	33.85
18	Point	4144	17	8.794355	33144	34.010278	-117.426389	-117.42639	34.01028
19	Point	4149	15	8.899194	33149	33.708333	-117.243056	-117.24306	33.70833

Attribute Tables

- Basic operators on attribute tables
  - Selection: picking certain rows
  - Projection: picking certain columns
  - Join: compositions of relations

Project Operator

- The **project** operator is unary
  - It outputs a new relation that has a subset of attributes
  - Identical tuples in the output relation are coalesced

Relation Sells:

bar	beer	price
Chimy's	Bud	2.50
Chimy's	Miller	2.75
Cricket's	Bud	2.50
Cricket's	Miller	3.00

Prices := $PROJ_{beer,price}$(Sells):

beer	price
Bud	2.50
Miller	2.75
Miller	3.00

Select Operator

- The **select** operator is unary
  - It outputs a new relation that has a subset of tuples
  - A condition specifies those tuples that are required

Relation Sells:

bar	beer	price
Chimy's	Bud	2.50
Chimy's	Miller	2.75
Cricket's	Bud	2.50
Cricket's	Miller	3.00

ChimyMenu := $\text{SELECT}_{\text{bar="Chimy's"}}$(Sells):

bar	beer	price
Chimy's	Bud	2.50
Chimy's	Miller	2.75

Join Operator

- The **join** operator is binary
  - It outputs the combined relation where tuples agree on a specified attribute (natural join)

Sells(bar, beer, price)

bar	beer	price
Chimy's	Bud	2.50
Chimy's	Miller	2.75
Cricket's	Bud	2.50
Cricket's	Coors	3.00

Bars(bar, address)

bar	address
Chimy's	2417 Broadway St.
Cricekt's	2412 Broadway St.

BarInfo := Sells JOIN Bars

Note Bars.name has become Bars.bar to make the natural join "work."

BarInfo(bar, beer, price, address)

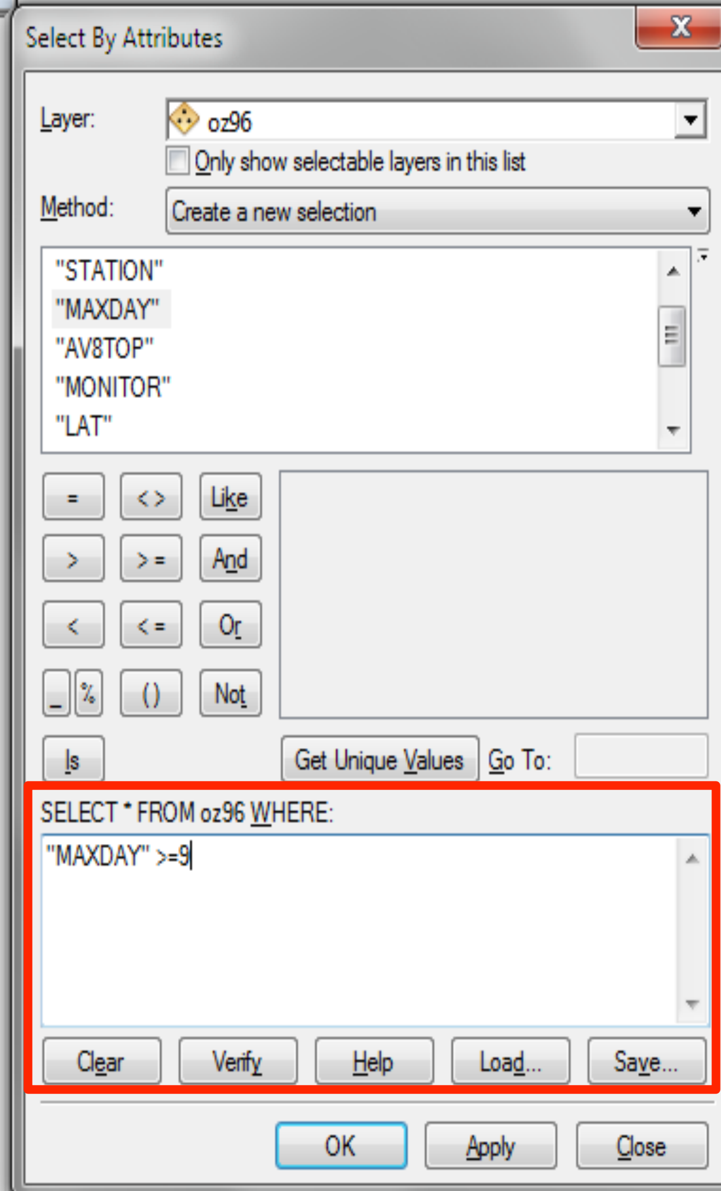
bar	beer	price	address
Chimy's	Bud	2.50	2417 Broadway St.
Chimy's	Milller	2.75	2417 Broadway St.
Cricket's	Bud	2.50	2412 Broadway St.
Cricket's	Coors	3.00	2412 Broadway St.

Join Operator

- Join is the most time-consuming of all relational operators to compute
  - In general, relational operators may not be arbitrarily reordered (left join, right join)
  - Query optimization aims to find an efficient way of processing queries, for example reordering to produce equivalent but more efficient queries

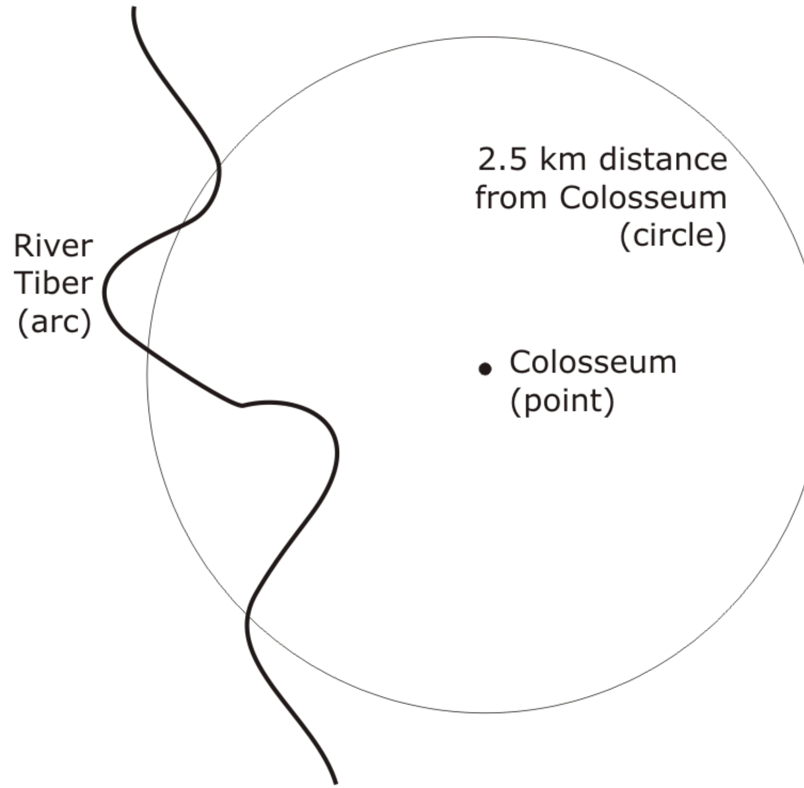
oz96

FID	Shape	STATION	MAXDAY	AV8TOP	MONITOR	LAT	LON	X_COORD	Y_COORD
0	Point	60	16	7.225806	70060	34.135833	-117.923611	-117.92361	34.13583
1	Point	69	14	5.899194	70069	34.176111	-118.315278	-118.31528	34.17611
2	Point	72	9	4.052885	70072	33.823611	-118.1875	-118.1875	33.82361
3	Point	74	13	7.181452	70074	34.199444	-118.534722	-118.53472	34.19944
4	Point	75	13	6.076613	70075	34.066944	-117.751389	-117.75139	34.06694
5	Point	84	8	3.157258	70084	33.929167	-118.209722	-118.20972	33.92917
6	Point	85	11	5.201613	70085	34.015	-118.059722	-118.05972	34.015
7	Point	87	11	4.717742	70087	34.067222	-118.226389	-118.22639	34.06722
8	Point	88	15	6.532258	70088	34.083333	-118.106944	-118.10694	34.08333
9	Point	89	15	7.540323	70089	34.3875	-118.534722	-118.53472	34.3875
10	Point	91	10	4.891129	70091	34.050833	-118.454167	-118.45417	34.05083
11	Point	94	10	4.149194	70094	33.92357	-118.37085	-118.37085	33.92357
12	Point	96	13	7.266129	70096	34.69012	-118.1334	-118.1334	34.69012
13	Point	3176	9	4.100806	30176	33.820278	-117.9125	-117.9125	33.82028
14	Point	3177	10	4.737903	30177	33.926111	-117.951389	-117.95139	33.92611
15	Point	3186	11	4.600806	30186	33.6275	-117.690278	-117.69028	33.6275
16	Point	3195	8	4.358871	30195	33.67511	-117.9269	-117.9269	33.67511
17	Point	4137	13	8.241935	33137	33.85	-116.543056	-116.54306	33.85
18	Point	4144	17	8.794355	33144	34.010278	-117.426389	-117.42639	34.01028
19	Point	4149	15	8.899194	33149	33.708333	-117.243056	-117.24306	33.70833
20	Point	4150	14	7.16129	33150	33.925	-116.876389	-116.87639	33.925
21	Point	4157	13	8.133065	33157	33.714555	-116.233894	-116.23389	33.71455
22	Point	4158	13	7.858871	33158	33.67401	-117.32114	-117.32114	33.67401
23	Point	4162	17	9.762097	33162	33.97562	-117.33263	-117.33263	33.97562
24	Point	5175	16	8.149194	36175	34.103611	-117.629167	-117.62917	34.10361
25	Point	5181	22	11.649194	36181	34.243889	-117.273611	-117.27361	34.24389
26	Point	5197	17	8.604839	36197	34.099444	-117.504167	-117.50417	34.09944
27	Point	5203	20	9.947581	36203	34.107222	-117.273611	-117.27361	34.10722
28	Point	5204	19	10.274194	36204	34.066667	-117.151389	-117.15139	34.06667
29	Point	5212	17	8.524194	36512	33.9846	-117.512733	-117.51273	33.9846
30	Point	5213	9	4.447917	36513	34.262276	-117.188543	-117.18854	34.26228
31	Point	591	18	8.241935	70591	34.143889	-117.851389	-117.85139	34.14389



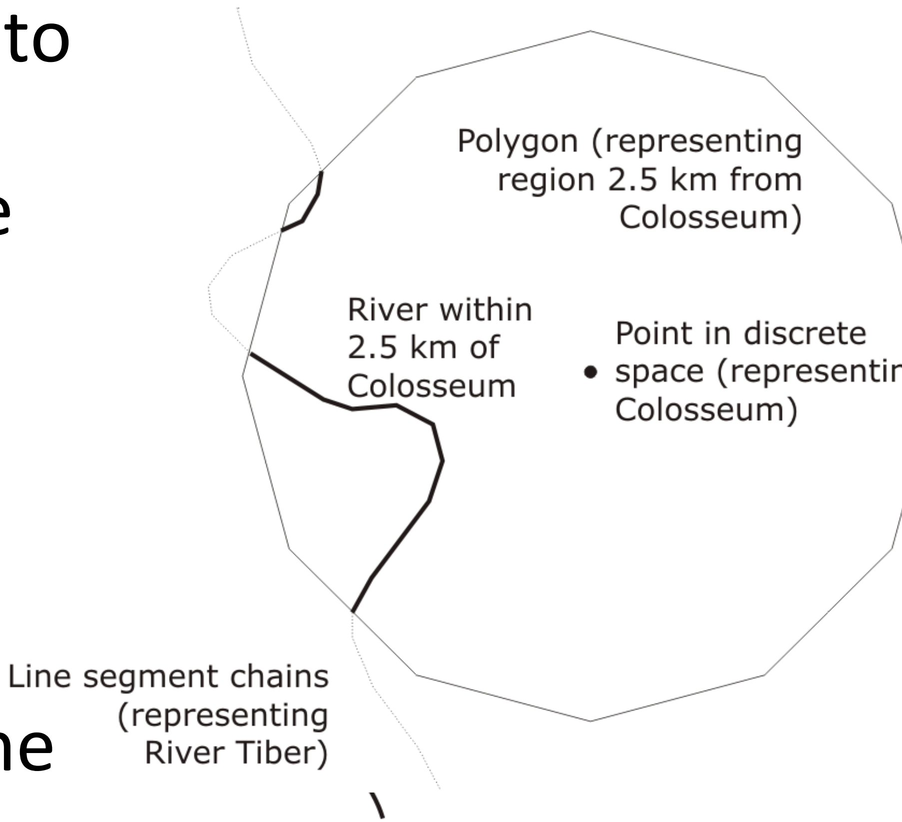
Example: GIS analysis

- For Italy's capital city, Rome, calculate the total length of the River Tiber which lies within 2.5 km of the Colosseum
  - First we need to model the relevant parts of Rome as objects
  - Operation ***length*** will act on ***arc***, and ***intersect*** will apply to form
    the piece of the ***arc*** in common
    with the disc



Example: GIS analysis

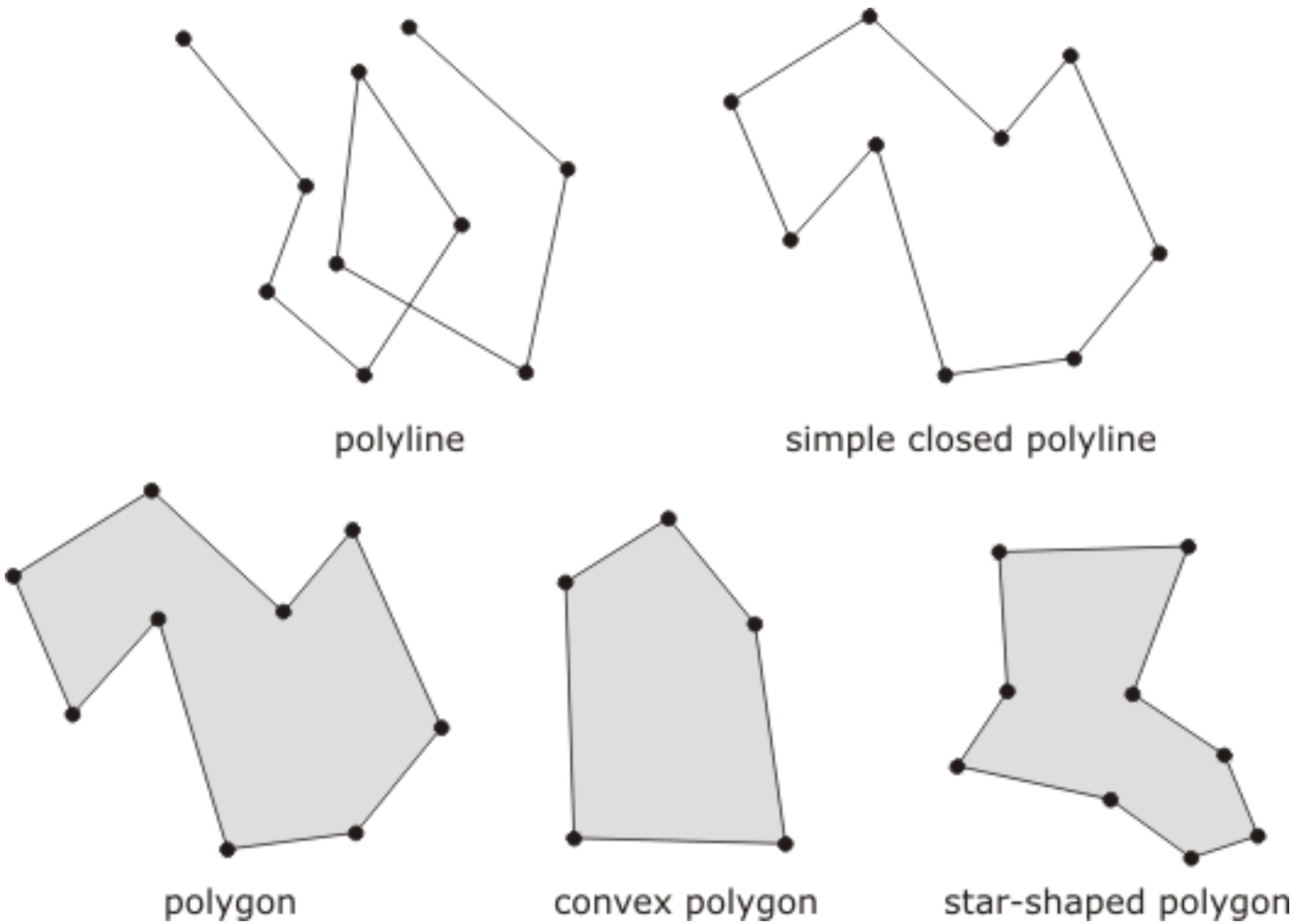
- A process of discretization must convert the objects to types that are computationally tractable
- A circle may be represented as a discrete polygonal area, arcs by chains of line segments, and points may be embedded in some discrete space



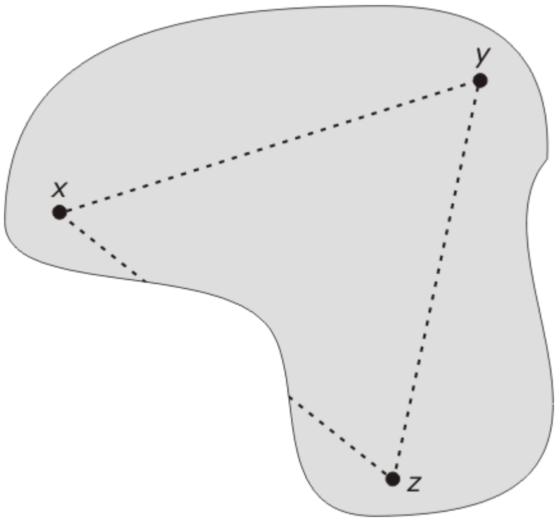
Primitive Objects

- ***Euclidean Space***: coordinatized model of space
  - Transforms spatial properties into properties of tuples of real numbers
  - Coordinate frame consists of a fixed, distinguished point (origin) and a pair of orthogonal lines (axes), intersecting in the origin
- Point objects
- Line objects
- Polygonal objects

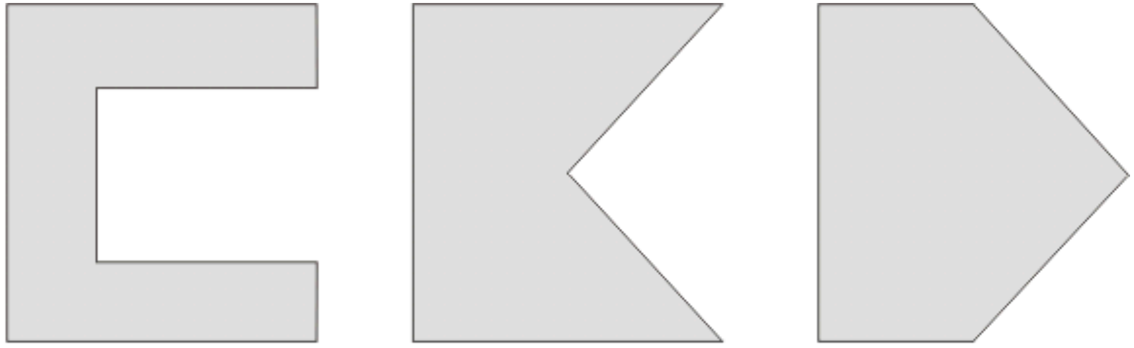
Polygonal objects



Convexity



Visibility between points x, y, and z

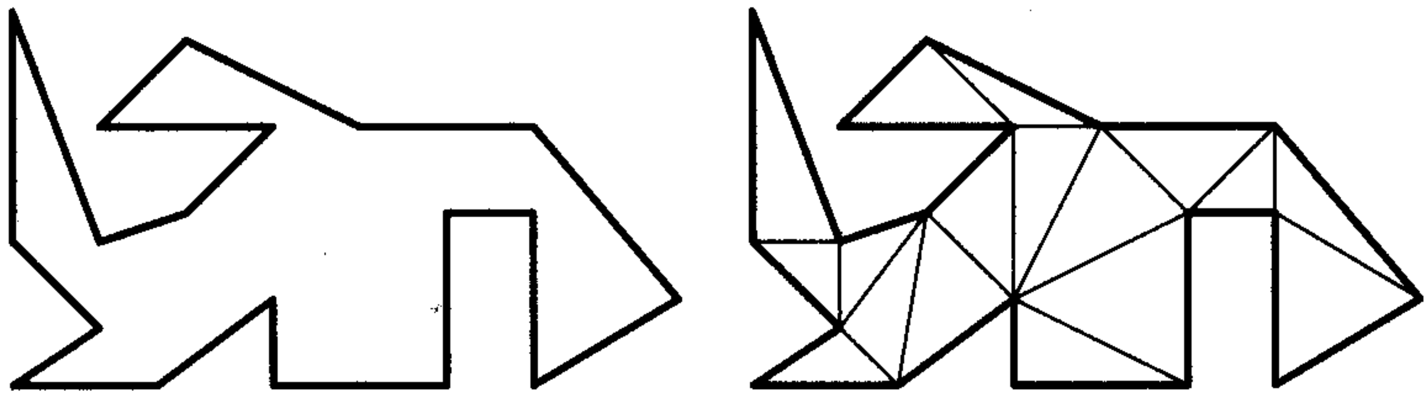


Not semi-convex

Semi-convex
Not convex

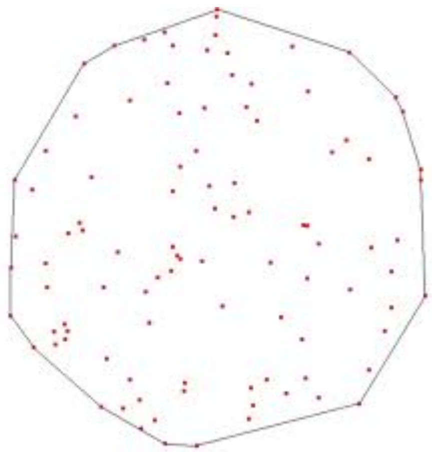
Convex

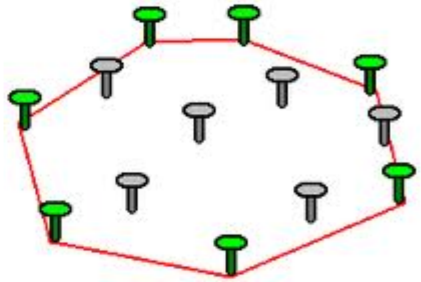
Example: Triangulation

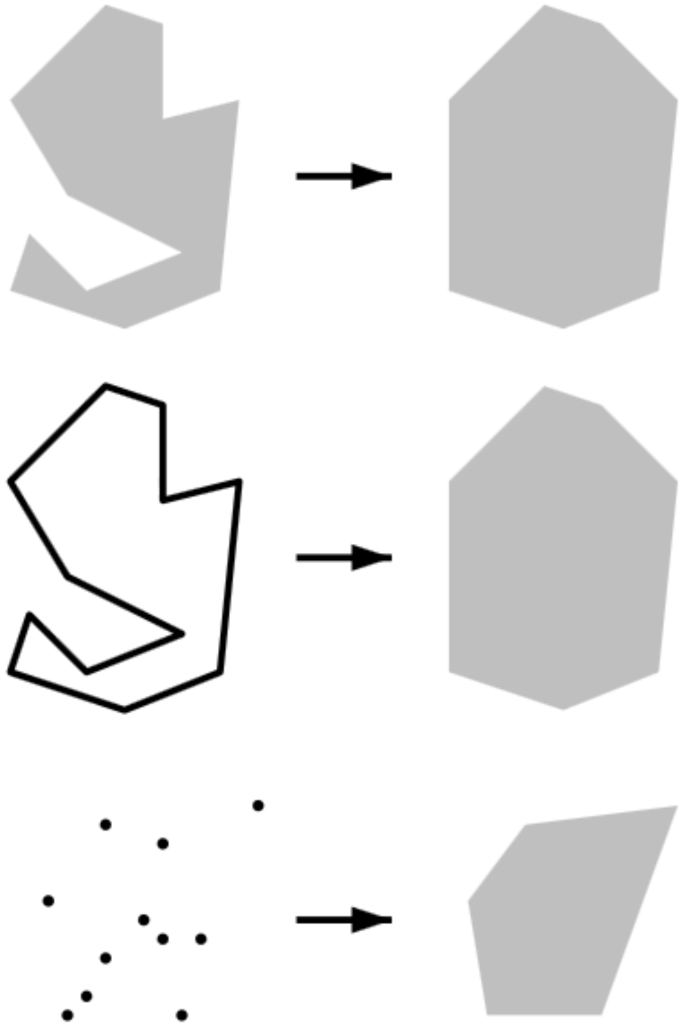


- Every simple polygon has a triangulation. Any triangulation of a simple polygon with n vertices consists of exactly n – 2 triangles
- Art Gallery Problem
  - How many cameras are needed to guard a gallery and how should they be placed?
  - Upper bound N/3

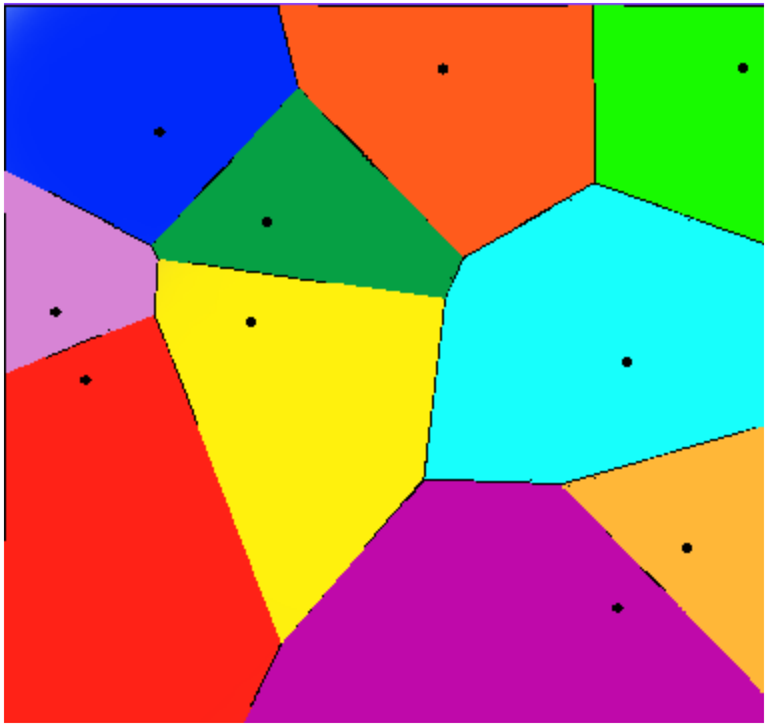
Related: Convex Hull



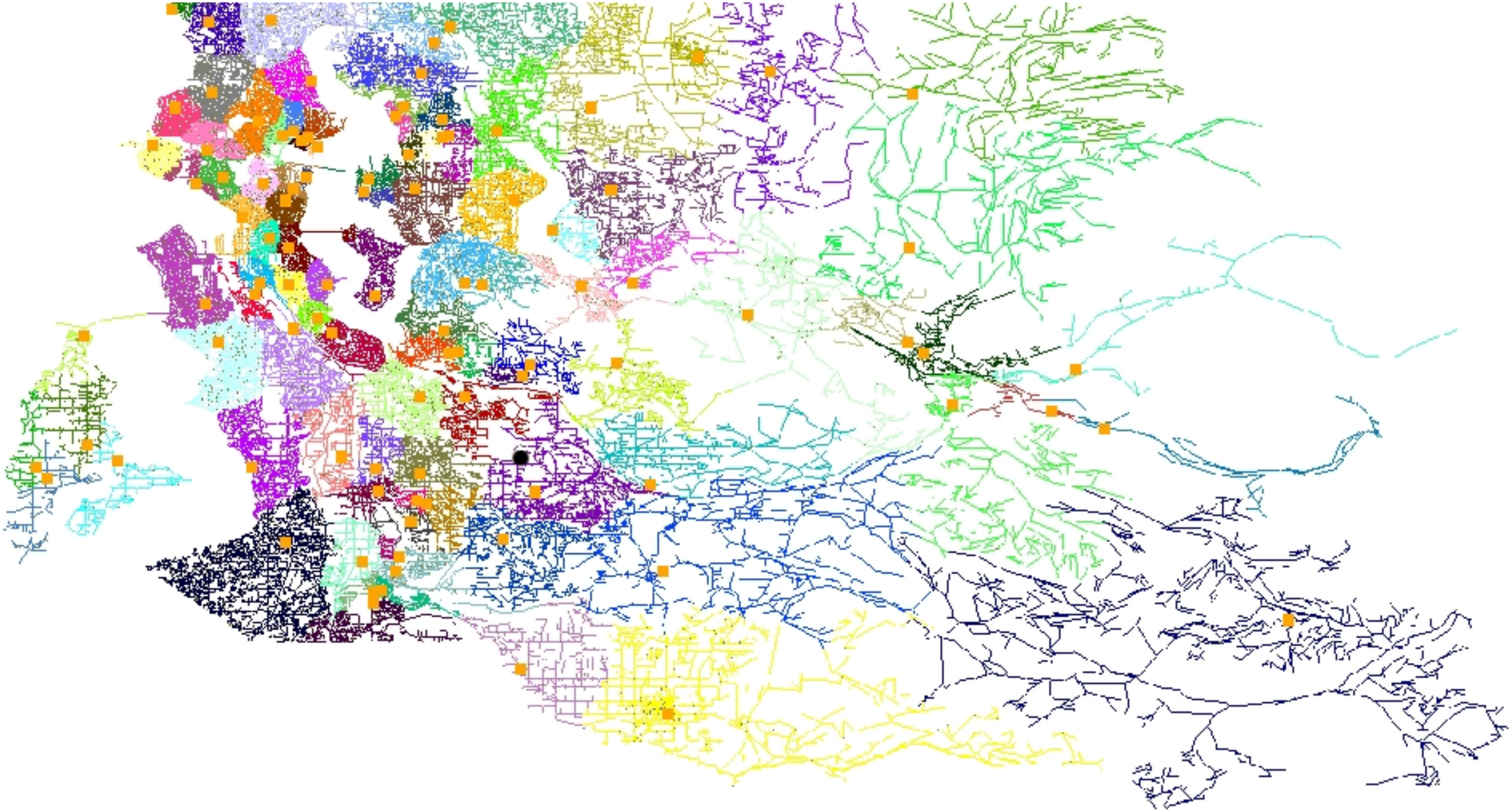




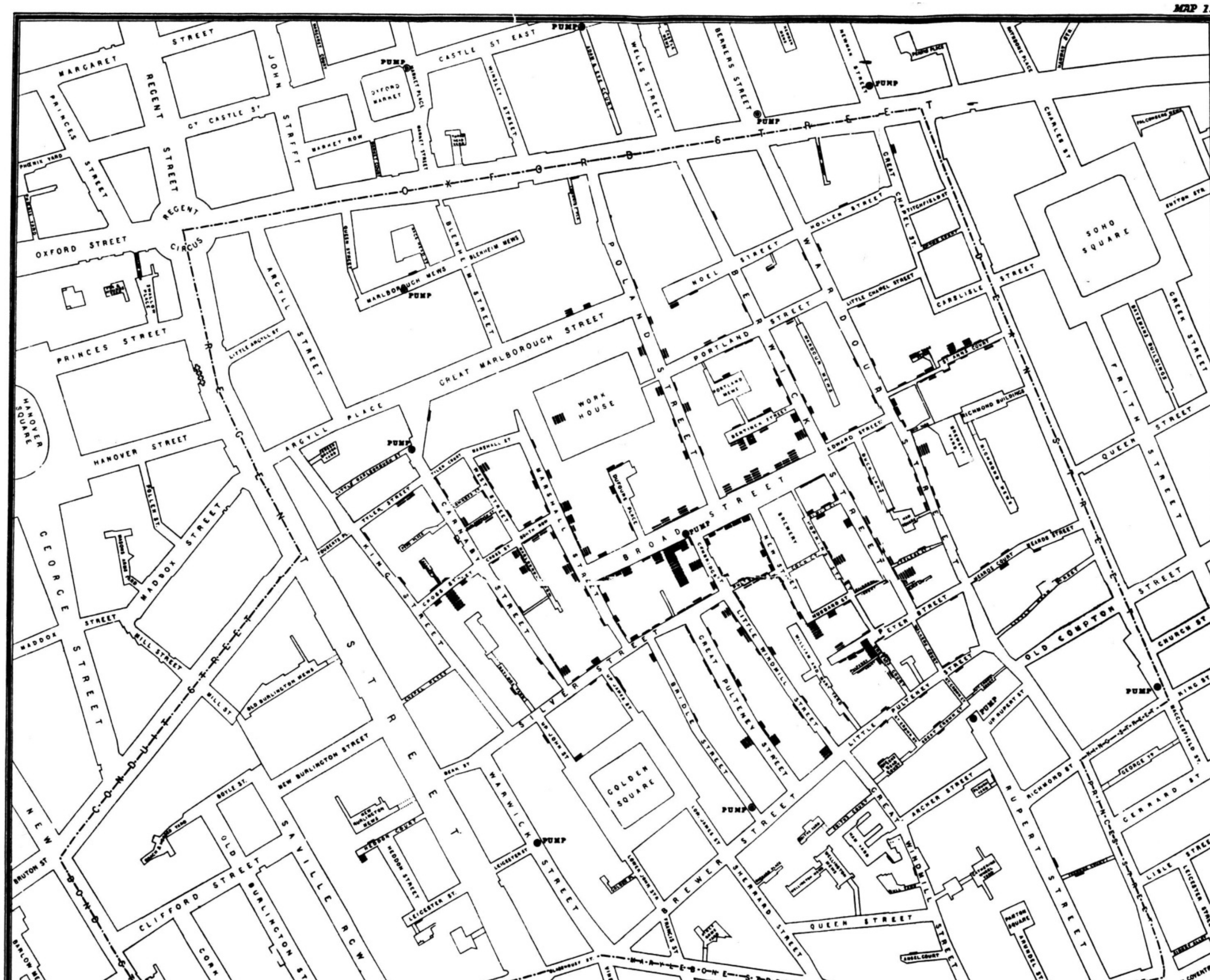
Related: Voronoi Diagram

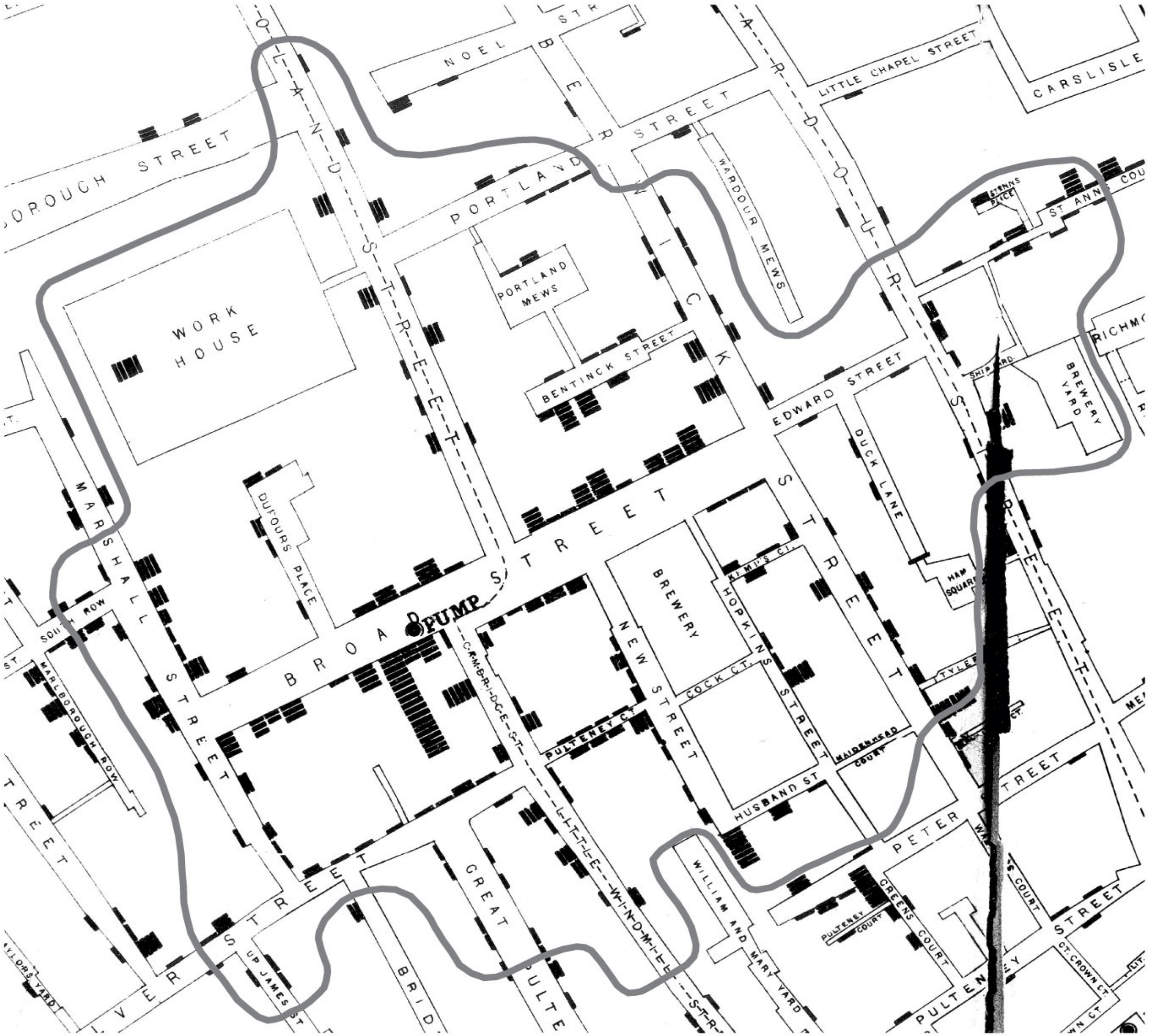


Voronoi Diagram on Road Network



John Snow, Pumps and Cholera Outbreak





MARLBOROUGH STREET
NOEL
PORTLAND STREET
LITTLE CHAPEL STREET
CARSLISLE
WARDOUR MEWS
WORK HOUSE
PORTLAND MEWS
BENTINCK STREET
EDWARD STREET
ST ANNS PLACE
ST ANN'S COURT
RICHMOND
SHEPHERD
BREWERY YARD
DUCK LANE
MARSHALL STREET
DUFOURS PLACE
BROAD STREET
PUMP
SOUTH ROW
MARLBOROUGH ROW
CAMBRIDGE ST
NEW STREET
BREWERY
HOPKINS STREET
KEMPS CT.
HAM SQUARE
COCK CT.
PULTENEY CT.
HUSBAND ST
MAIDENHEAD COURT
TYLER
PETER STREET
GREAT PULTENEY
LITTLE WINDMILL STREET
WILLIAM AND MARY YARD
GREENS COURT
PULTENEY COURT
PULTENEY STREET
UP JAMES ST
TAYLORS YARD
CT. CROWN CT.

Primitive GIS Operations

- in Euclidean spaces
  - Length, bearing, area
    - How many ways you can think of to calculate the area of a polygon?
    - How to test which side of a point corresponding to a line?
  - Distance between objects (points, lines, polygons)
    - Distance could be ambiguous, e.g., what is the difference from Lubbock to Dallas (from city center or city boundary?).
  - Centroid
    - Not necessarily within in the boundary of polygon
  - Point in polygon
    - Ray casting method
  - Point on line
    - area
  - Buffer
  - Intersection/overlay
- In topological spaces
  - Spatial relations (within, touch, cover, …)

Distance and angle between points

- Length of a line segment can be computed as the ***distance*** between successive pairs of points

$$|pq| = \sqrt{(x_q - x_p)^2 + (y_q - y_p)^2}$$

- The bearing, θ, of *q* from *p* is given by the unique solution in the interval [0,360] of the simultaneous equations:

$$\cos\theta = \frac{y_q - y_p}{|pq|} \qquad \sin\theta = \frac{x_q - x_p}{|pq|}$$

Area

- Let P be a simple polygon (no boundary self-intersections) with vertex vectors: $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ where $(x_1, y_1) = (x_n, y_n)$ .Then the area is:

$$\textbf{area}(P) = \frac{1}{2}\sum_{i=1}^{n-1} x_i y_{i+1} - x_{i+1} y_i$$

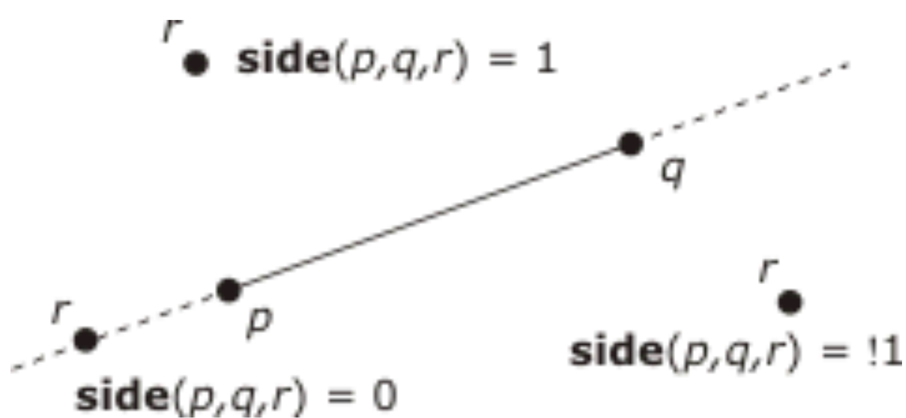
- In the case of a triangle *pqr*

$$\textbf{area}(pqr) = \frac{x_p y_q - x_q y_p + x_q y_r - x_r y_q + x_r y_p - x_p y_r}{2}$$

Area of a simple polygon

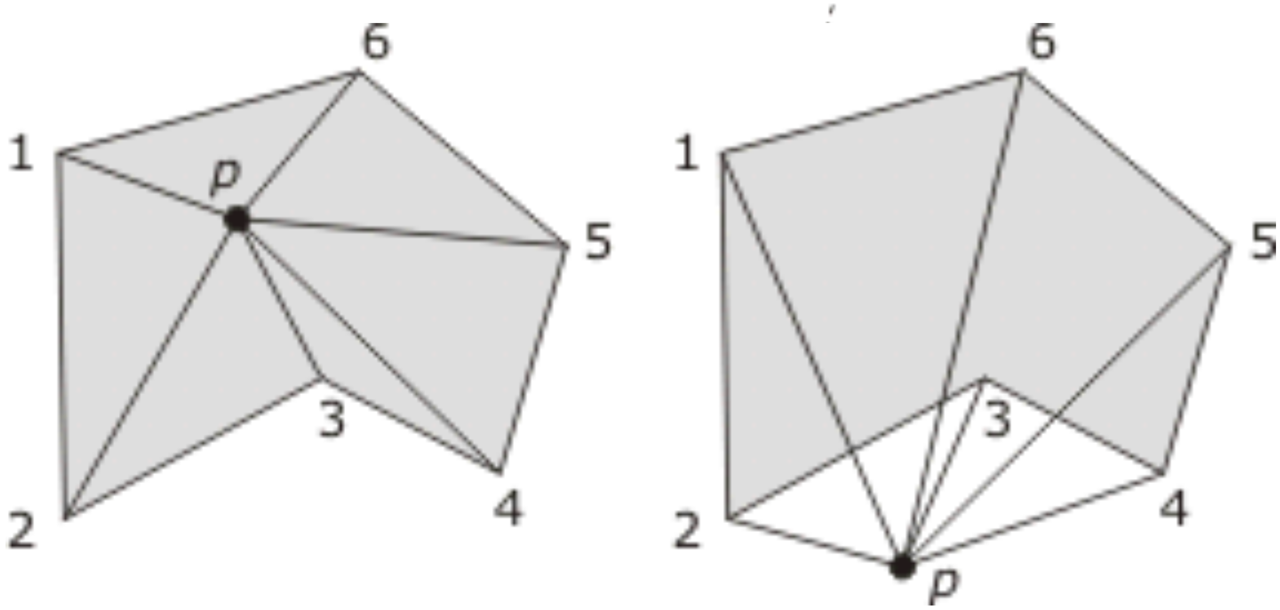
- Note that the area may be positive or negative
- In fact, **area**(*pqr*) = -**area**(*qpr*)
- If *p* is to the left of *qr* then the area is positive, if *p* is to the right of *qr* then the area is negative

$$\textbf{side}(p,q,r) = \begin{cases} 1 & \text{if } \textbf{area}(pqr) > 0 \quad (p \text{ is left of } qr) \\ 0 & \text{if } \textbf{area}(pqr) = 0 \quad (pqr \text{ are collinear}) \\ -1 & \text{if } \textbf{area}(pqr) < 0 \quad (p \text{ is right of } qr) \end{cases}$$



Point in polygon

- Determining whether a point is inside a polygon is one of the most fundamental operations in a spatial database
- ***Semi-line method (ray casting)*** : checks for odd or even numbers of intersections of a semi-line with polygon
- ***Winding method***: sums bearings from point to polygon vertices



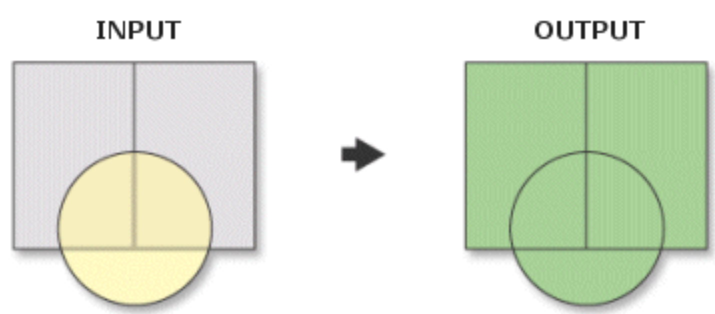

Is Biloxi in Mississippi?
1 intersection
= Hey Y'all!
1
Is New Orleans in Mississippi?
2 Intersections
= Reject-shon!
2
1

Primitive GIS operations: Overlay

- Union
- Intersect
- Erase
- Identity
- Update
- Spatial Join
- Symmetrical Difference

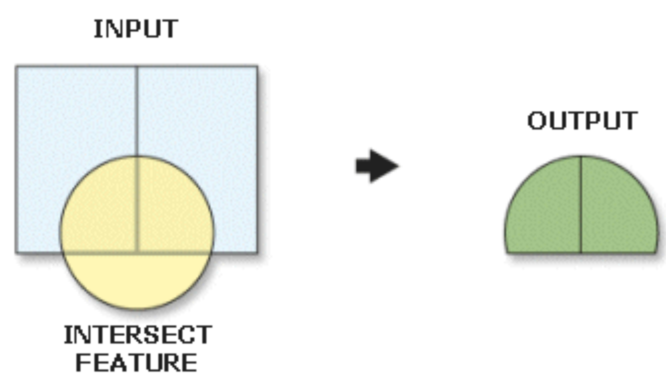
Overlay

- Union
  - Computes a geometric union of the input features. All features and their attributes will be written to the output feature class.



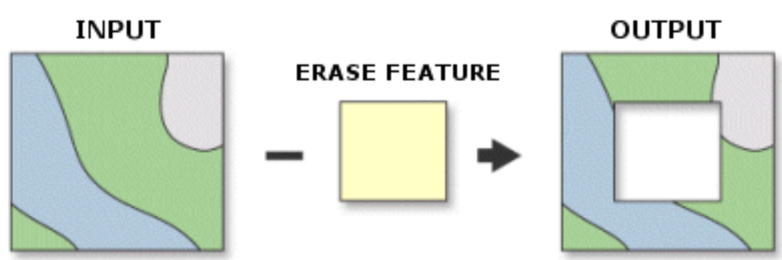
Overlay

- Intersect
  – Computes a geometric intersection of the input features. Features or portions of features which overlap in all layers and/or feature classes will be written to the output feature class.



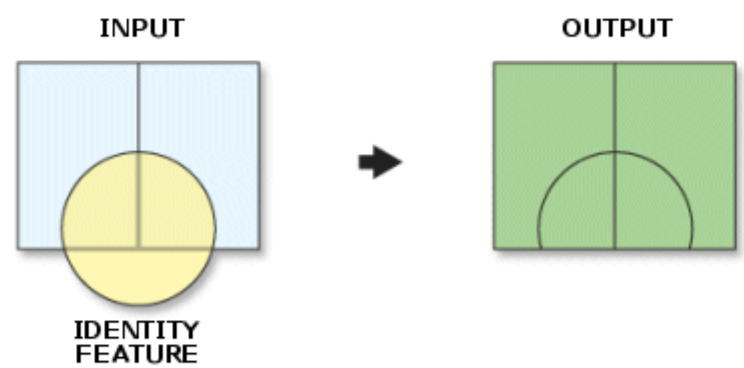
Overlay

- Erase
  - Creates a feature class by overlaying the Input Features with the polygons of the Erase Features. Only those portions of the input features falling outside the erase features outside boundaries are copied to the output feature class



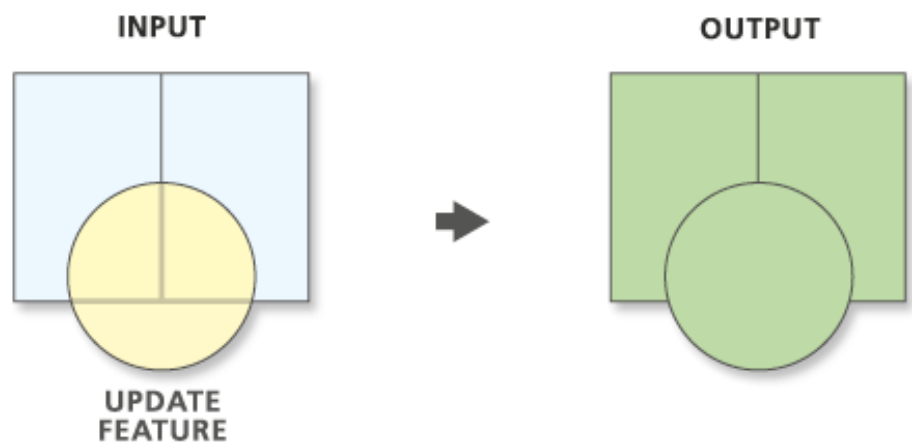
Overlay

- Identity
  - Computes a geometric intersection of the input features and identity features. The input features or portions thereof that overlap identity features will get the attributes of those identity features



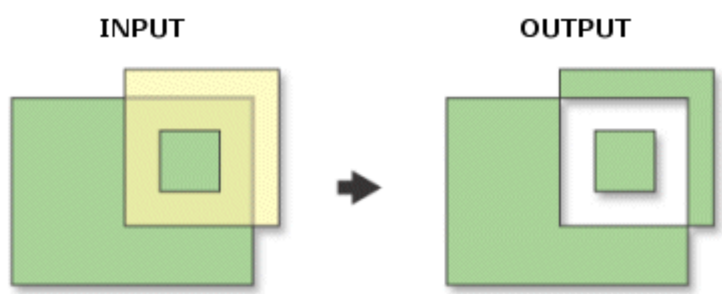
Overlay

- Update
  - Computes a geometric intersection of the Input Features and Update Features. The attributes and geometry of the input features are updated by the update features in the output feature class.



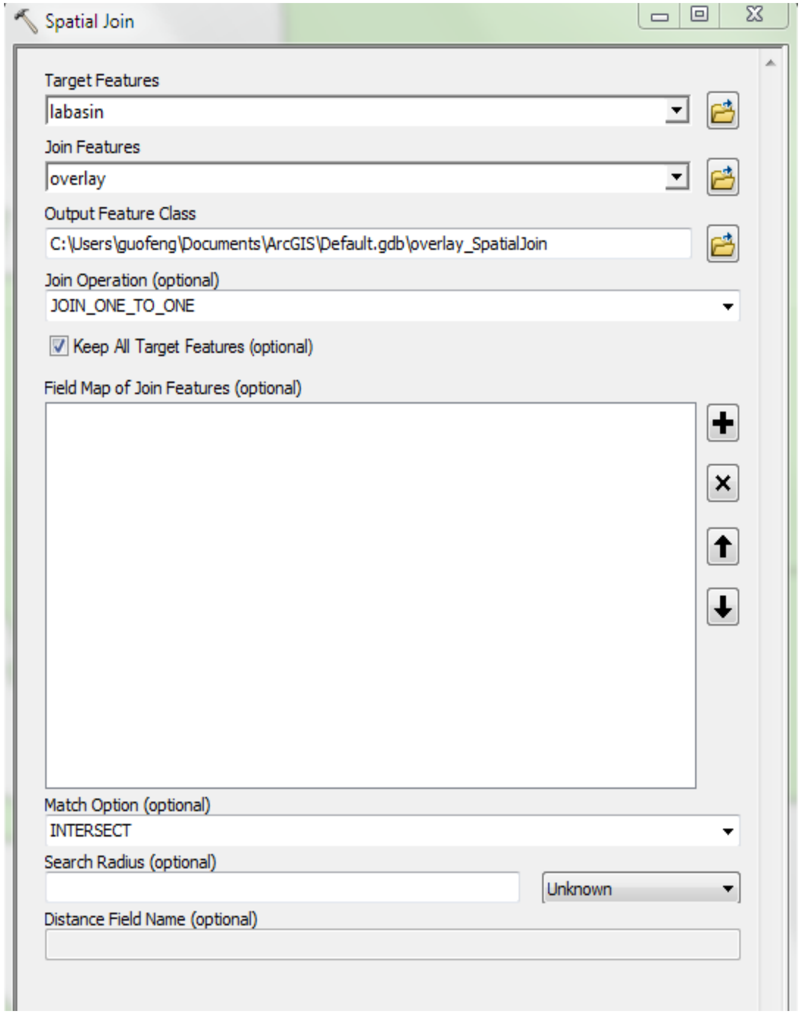
Overlay

- Symmetrical difference
  - Features or portions of features in the input and update features that do not overlap will be written to the output feature class.



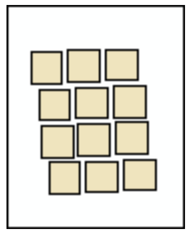
Overlay

- Spatial join
  - Joins attributes from one feature to another based on the spatial relationship. The target features and the joined attributes from the join features are written to the output feature class.

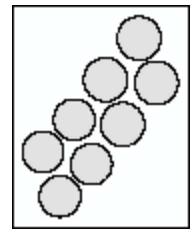


Quiz

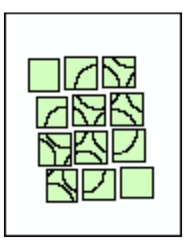
Input Feature



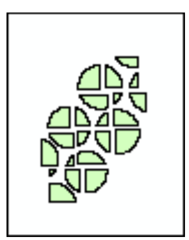
Overlay Feature



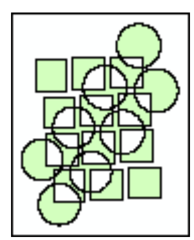
Which of the following is the result of identity, intersect, symmetrical difference, union and update respectively?



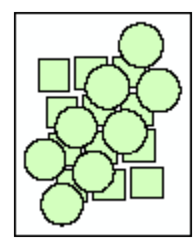
A



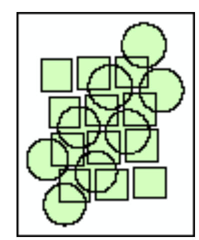
B



C

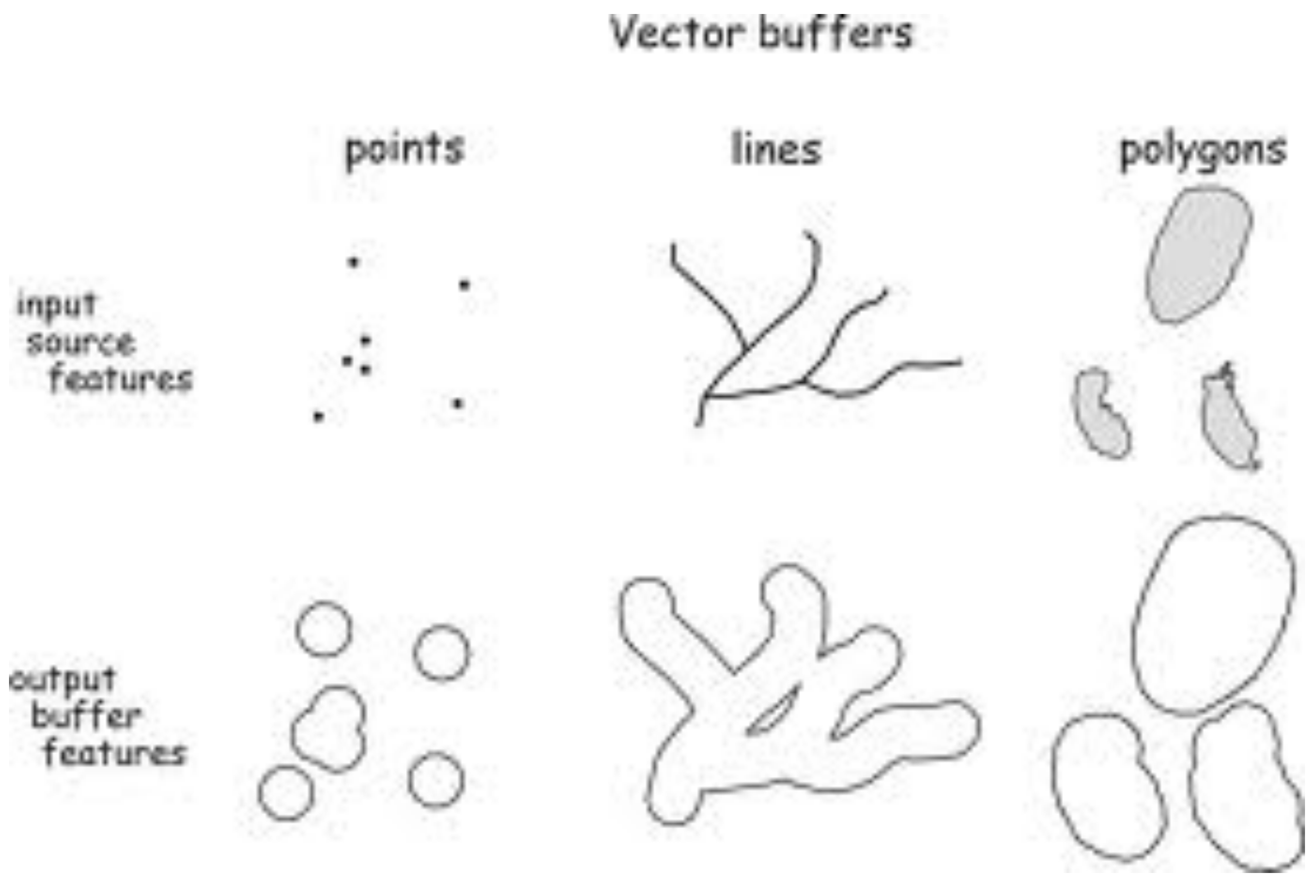


D



E

Buffer



Vector buffers
points
lines
polygons
input source features
output buffer features

- Primitive operators
  - you might already realized that these primitive operators are often used collaboratively with each other, and other analytical methods (e.g., dissolve, surface analysis, interpolation) that we will introduce in the coming lectures.

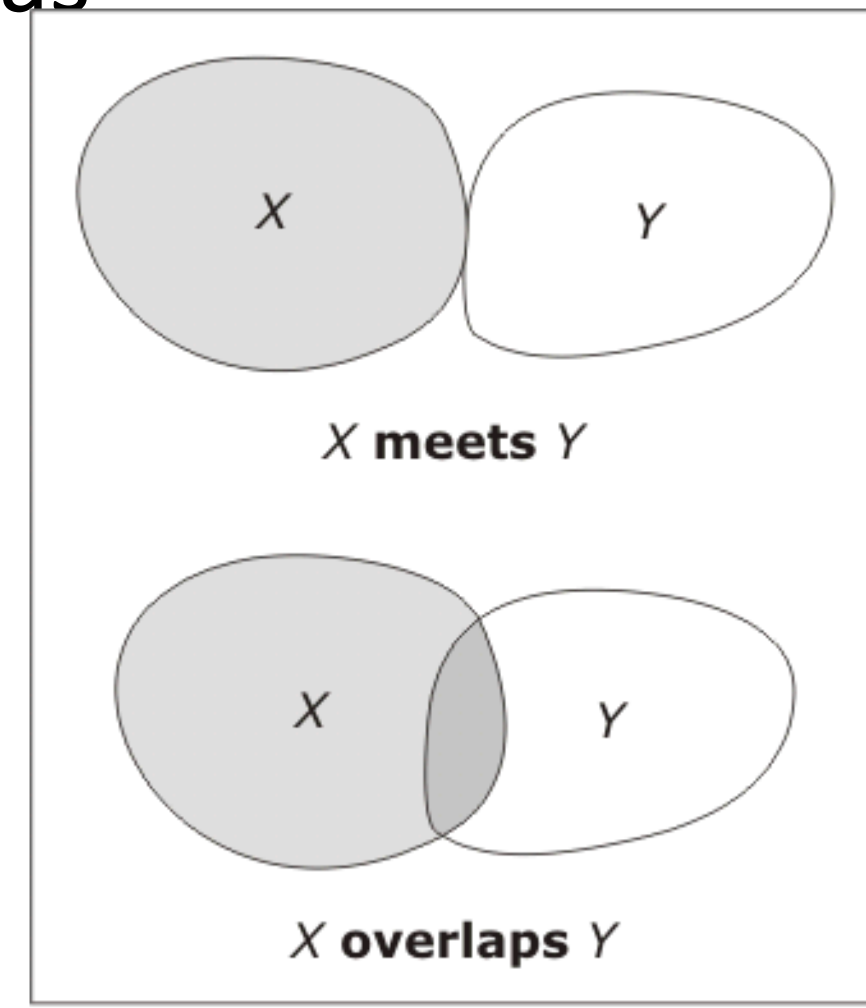
Topological spatial operations: spatial relationship

- Object types with an assumed underlying topology are ***point, arc***, ***loop*** and ***area***
- Operations:
  - ***boundary***, ***interior***, ***closure*** and ***connected*** are defined in the usual manner
  - ***components*** returns the set of maximal connected components of an area
  - ***extremes*** acts on each object of type arc and returns the pair of points of the arc that constitute its end points
  - ***is within*** provides a relationship between a point and a simple loop, returning true if the point is enclosed by the loop

Topological spatial operations for areas

– *X* ***meets*** *Y* if *X* and *Y* touch externally in a common portion of their boundaries

– *X* ***overlaps*** *Y* if *X* and *Y* impinge into each other's interiors

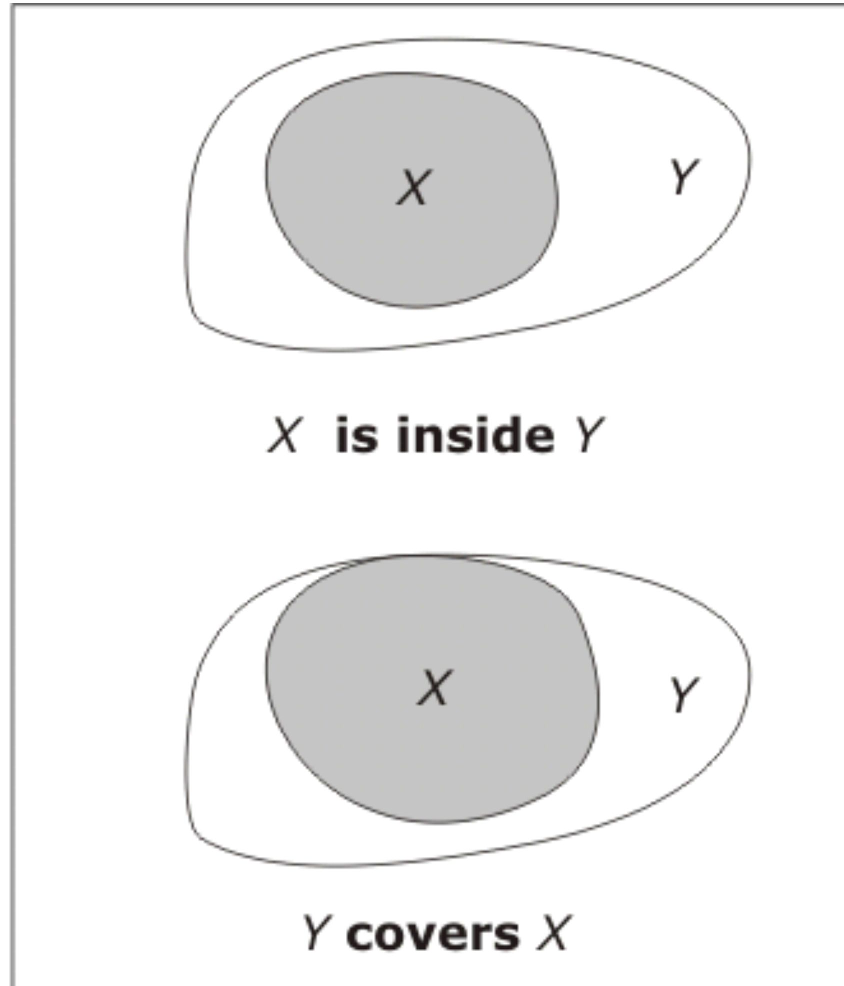


X is not **disjoint from** *Y*

Topological spatial operations for areas

- *X* ***is inside*** *Y* if *X* is a subset of *Y* and *X*, *Y* do not share a common portion of boundary

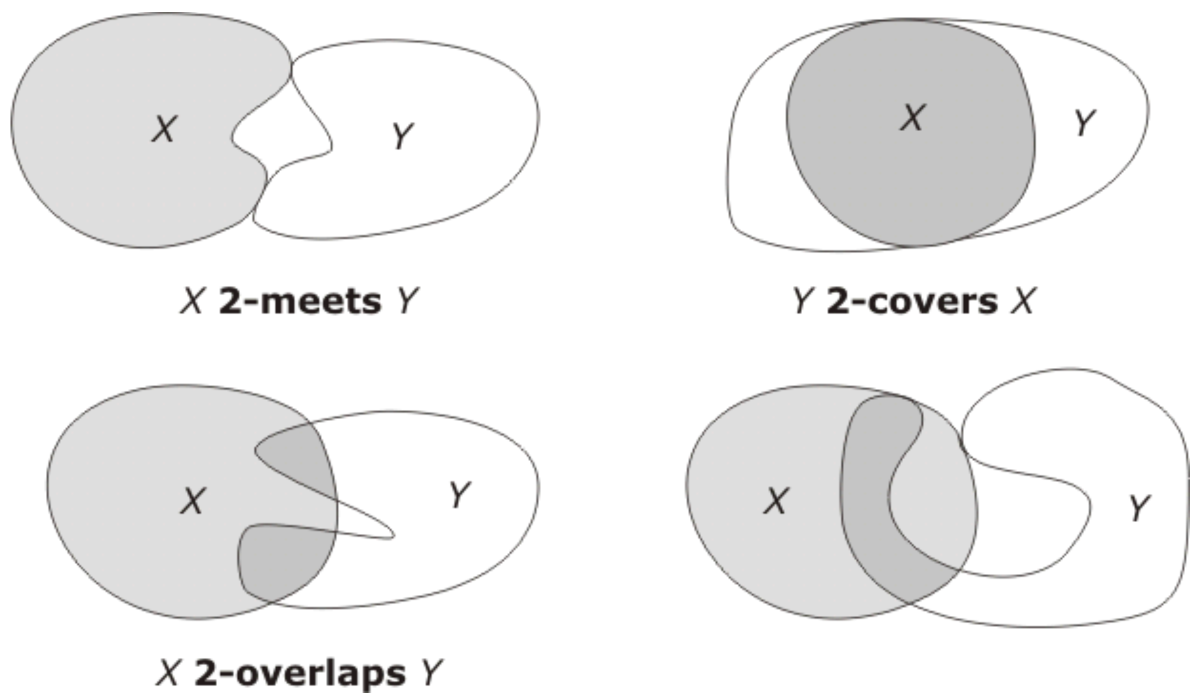
- *X* ***covers*** *Y* if *Y* is a subset of *X* and *X*, *Y* touch externally in a common portion of their boundaries

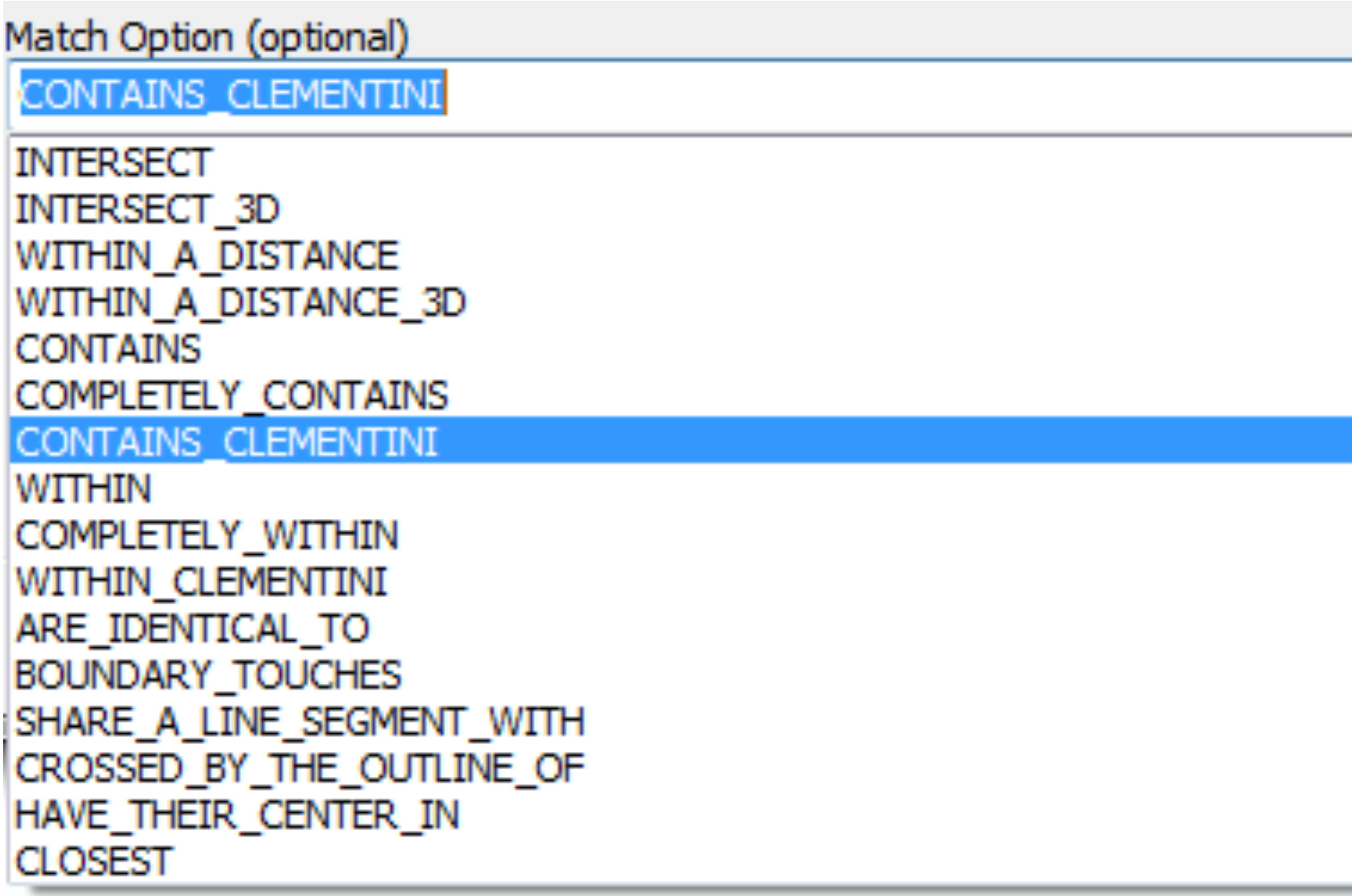


X is a **subset of** *Y*

Topological spatial operations

- There are an infinite number of possible topological relationships that are available between objects of type cell



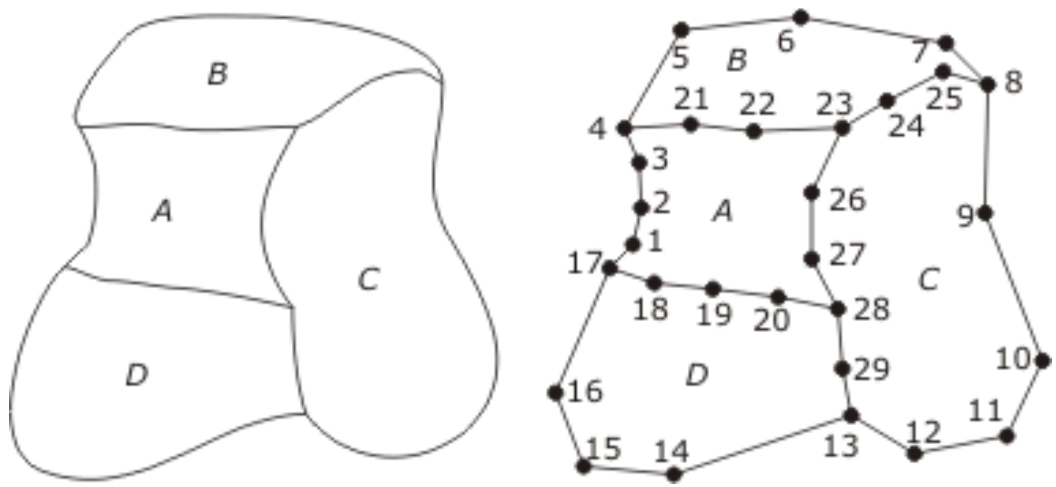


http://resources.esri.com/help/9.3/ArcGISDesktop/com/Gp_ToolRef/Data_Management_toolbox/select_by_location_colon_graphical_examples.htm

- Contain vs CONTAINS_CLEMENTINI:
  - the results of CONTAINS_CLEMENTINI will be identical to CONTAINS with the exception that if the feature in the Selecting Features layer is entirely on the boundary of the Input Feature Layer, with no part of the contained feature properly inside the feature in the Input Feature Layer, the input feature will not be selected.

Spaghetti

- ***Spaghetti*** data structure represents a planar configuration of points, arcs, and areas
- Geometry is represented as a set of lists of straight-line segments



Spaghetti- example

- Each polygonal area is represented by its boundary loop
- Each loop is discretized as a closed polyline
- Each polyline is represented as a list of points

A:[1,2,3,4,21,22,23,26,27,28,20,19,18,17]

B:[4,5,6,7,8,25,24,23,22,21]

C:[8,9,10,11,12,13,29,28,27,26,23,24,25]

D:[17,18,19,20,28,29,13,14,15,16]

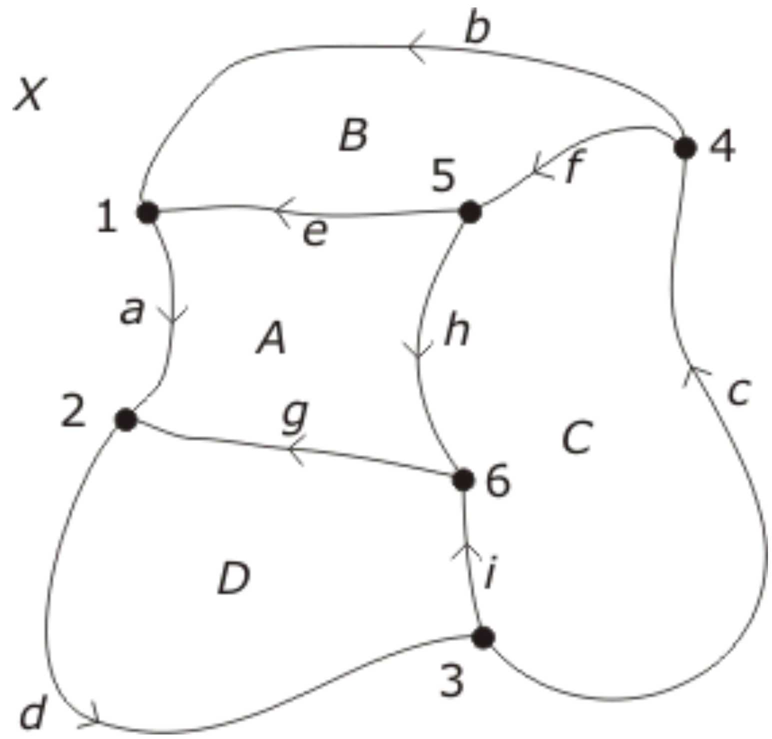
Issues

- There is **NO** explicit representation of the topological interrelationships of the configuration, such as adjacency
- Data consistence issues
  - *Silver polygons*
  - *Data redundancy*

NAA: node arc area

- Each directed arc has exactly one start and one end node.
- Each node must be the start node or end node (maybe both) of at least one directed arc.
- Each area is bounded by one or more directed arcs.
- Directed arcs may intersect only at their end nodes.
- Each directed arc has exactly one area on its right and one area on its left.
- Each area must be the left area or right area (maybe both) of at least one directed arc.

NAA: planar decomposition



Arc	Begin	End	Left	Right
a	1	2	*A*	*X*
b	4	1	*B*	*X*
c	3	4	*C*	*X*
d	2	3	*D*	*X*
e	5	1	*A*	*B*
f	4	5	*C*	*B*
g	6	2	*D*	*A*
h	5	6	*C*	*A*
i	3	6	*D*	*C*

- End of this topic