



# Vector storage and access; algorithms in GIS

---

**This is lecture 6**



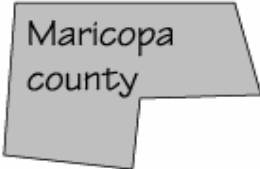
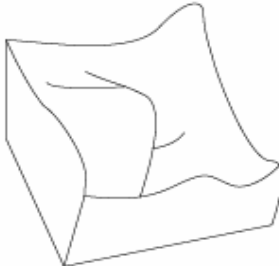

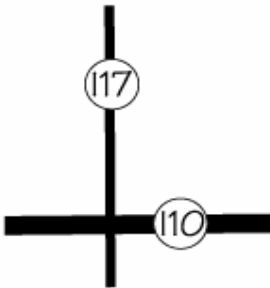

---

# Vector data storage and access

✦ Vectors are built from points, line and areas.

✦  $(x,y)$

✦ Surface:  $(x,y,z)$

Object	Point	Lines	Area	Surface
Dimension	0	1	2	3
Examples	<p>Telephone Pole</p>  <p>pole (645)</p>	<p>River</p> 	<p>County Boundary</p>  <p>Maricopa county</p>	<p>Physical Terrain</p> 
	<p>Town (at small scale)</p>  <p>Phoenix</p>	<p>Road</p> 	<p>Property Boundary</p>  <p>1479 East St.</p>	

Spatial Objects

# Vector data access

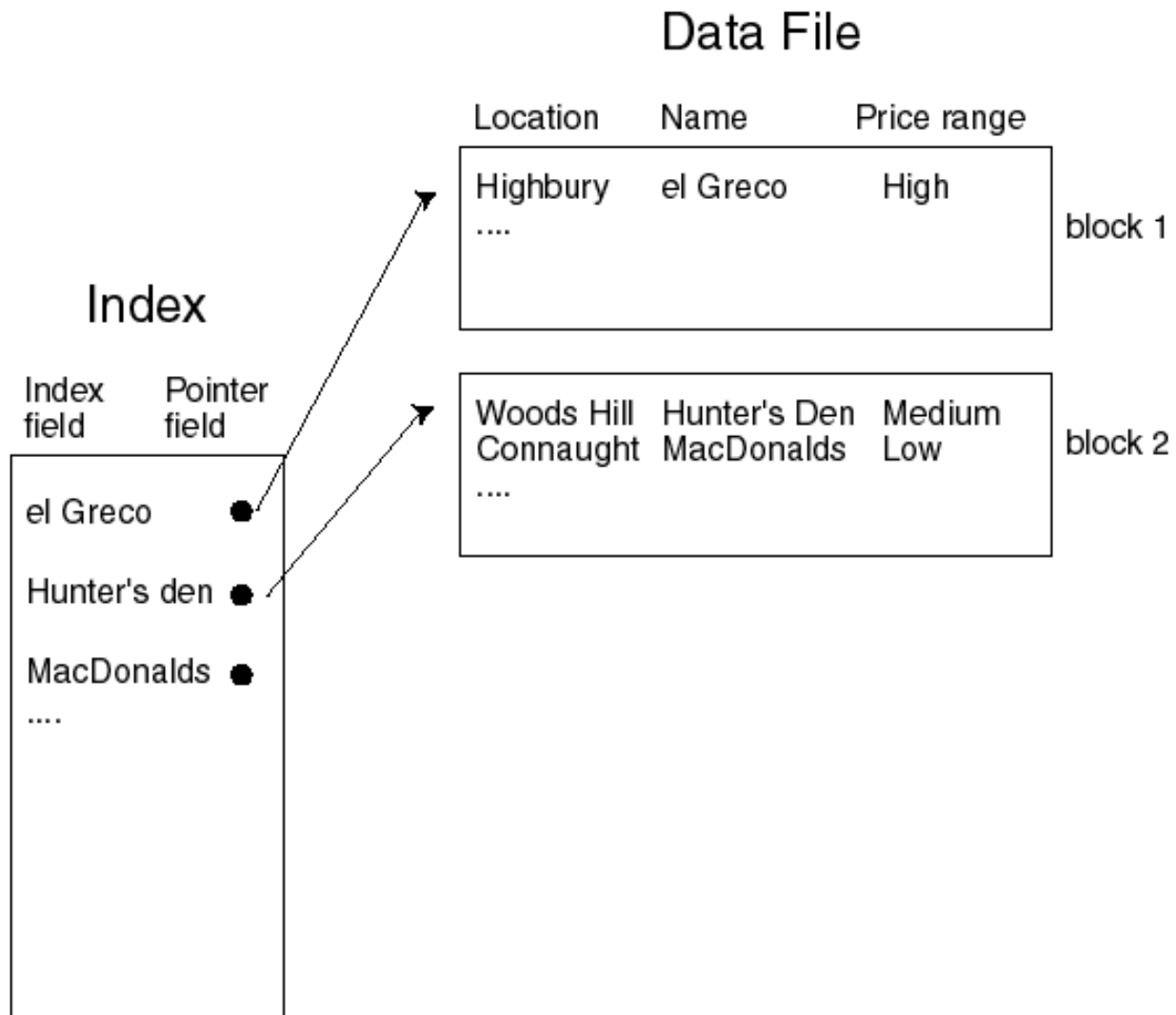
- ✦ **Access to vector data** requires some ingenuity.
- ✦ In the past, pointers to a sequential list were used to indicate the relationships between basic vector units (points, lines and areas).
- ✦ Problems with access time especially when points out of sequence.

# Database indexing

- ✦ **Database indexing introduced to speed up data access.**
- ✦ Indexing speeds up queries by keeping track of information.
- ✦ It organizes the data in sequential order. .
- ✦ Search time shifts from linear to logarithmic time.

The indexing system points to data blocks.

note  
that  
....  
means  
etc.



# Indexes

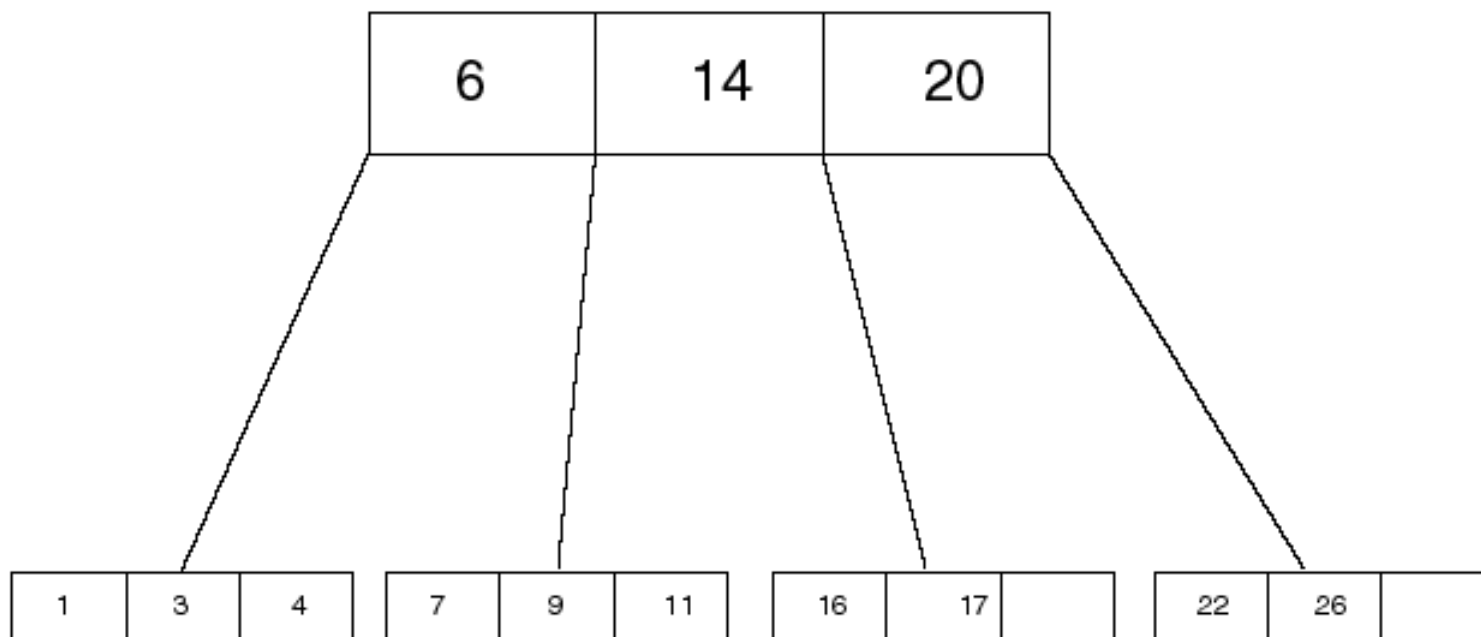
- ✦ Index field: contains the value of the indexing field (in sorted order).
- ✦ Pointer field: contains the addresses of disk blocks that have the index value
- ✦ Speed up performance of other fields too but at the expense of the extra space required for the index itself.

# Problem with indices

- ✦ They can get really big, and take up almost as much space as the information they point to.
- ✦ Search time can be further decreased by allowing the index to be indexed (recursive index).
- ✦ This is the principle behind the multi-level index.
- ✦ The *B Tree* is a multi-level index.



# B-tree for indexing records



At the root node of our B-tree, the first pointer can include values from 1 to 6. The second pointer contains values from 6 to 14, the third from 15 to 19 and the fourth from 20 to whatever. The insertion of a new record may require restructuring of the B-tree. For example, if we need to insert a record with the index value 18, then there is one node left. On the other hand, a record of value 8 would cause the B-tree to be restructured.

# B-tree as a multi-level index

Example:

We have a set of data records that are indexed using integers. Each index field has a pointer to a record that is indexed. Each node of the B-tree contains a list of index fields (with pointers to the data) interleaved. The value of the index field for all descendants is *within the range set by the index fields of the ancestor node*. Here, we have a range of 3 for each pointer. The *fan-order* of our B-tree is 4.

(see previous slide)

A search begins at the root node and a route is traced to one of the descendants or the bottom of the tree, whichever comes first. If the bottom is reached, a message to that effect is returned.

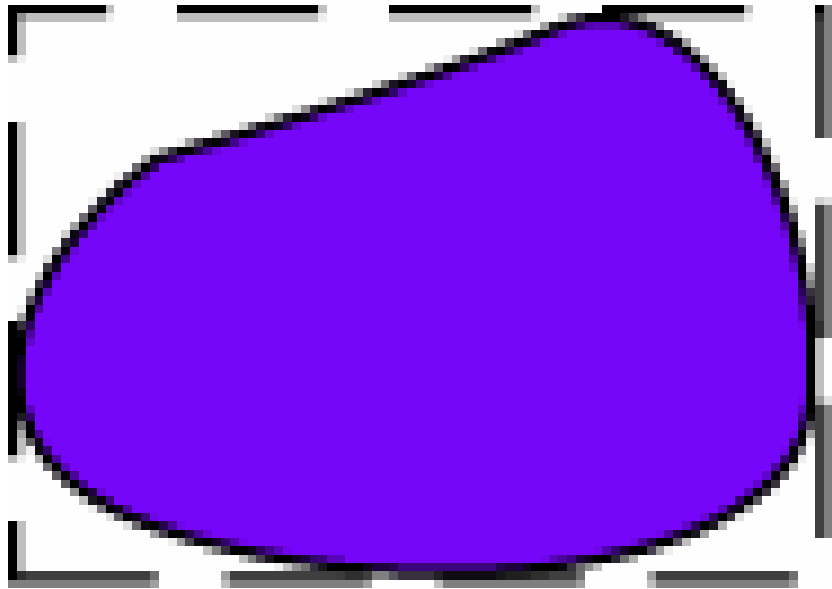
If a node runs out of spaces for pointers (3 per node in this case), the B-tree is restructured.

# Properties of B-trees

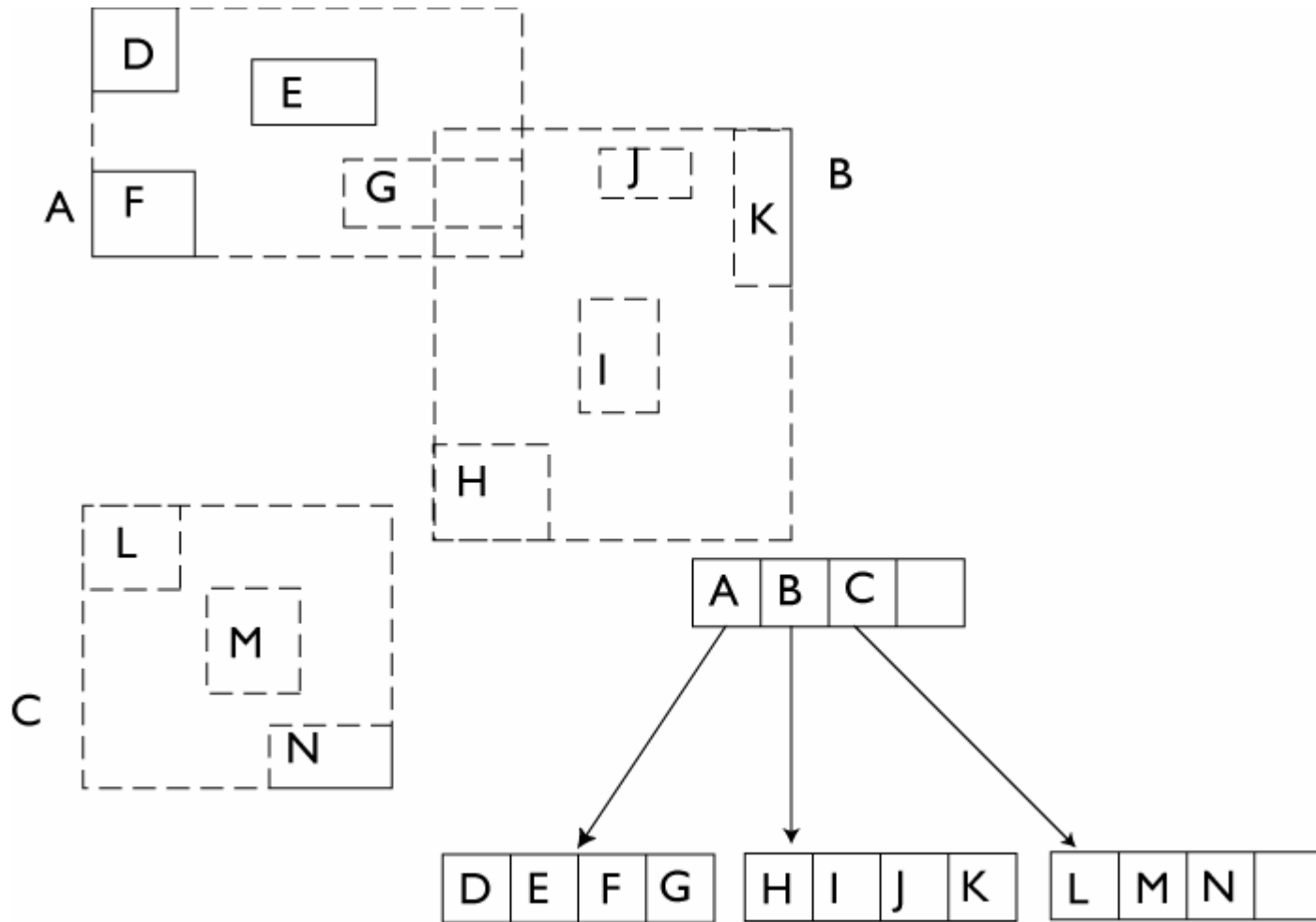
1. They are balanced because branches remain equal length as the tree grows.
2. Insertion and deletion can be calculated in terms of time complexity – using a logarithmic function.
3. Each node is guaranteed to be roughly half-full at all stages of the tree's evolution.

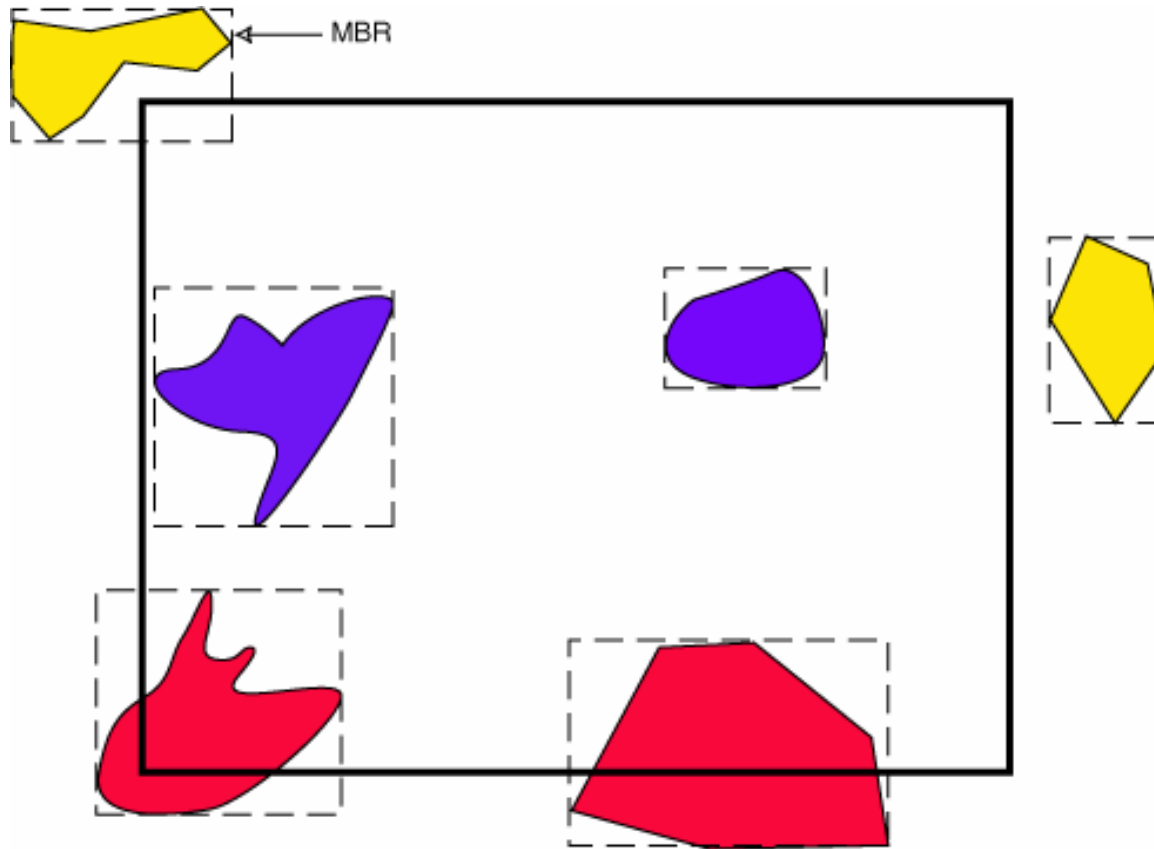
# R-tree index

- ✦ R-trees group objects using a rectangular approximation of their location (minimum bounding rectangle).
- ✦ Indexes points, lines or areas. Can be used for vector and true object data models.
- ✦ The MBR is assigned a parameter for maximum expansion.



R-tree index. Note applicability to GIS from a spatial perspective.





Polygon in polygon testing is also done using MBR. This enables fast searching. Note that the top left (yellow) object is outside, though the MBR does not indicate this. In this case, vertices would be tested to determine if any are within the study area.

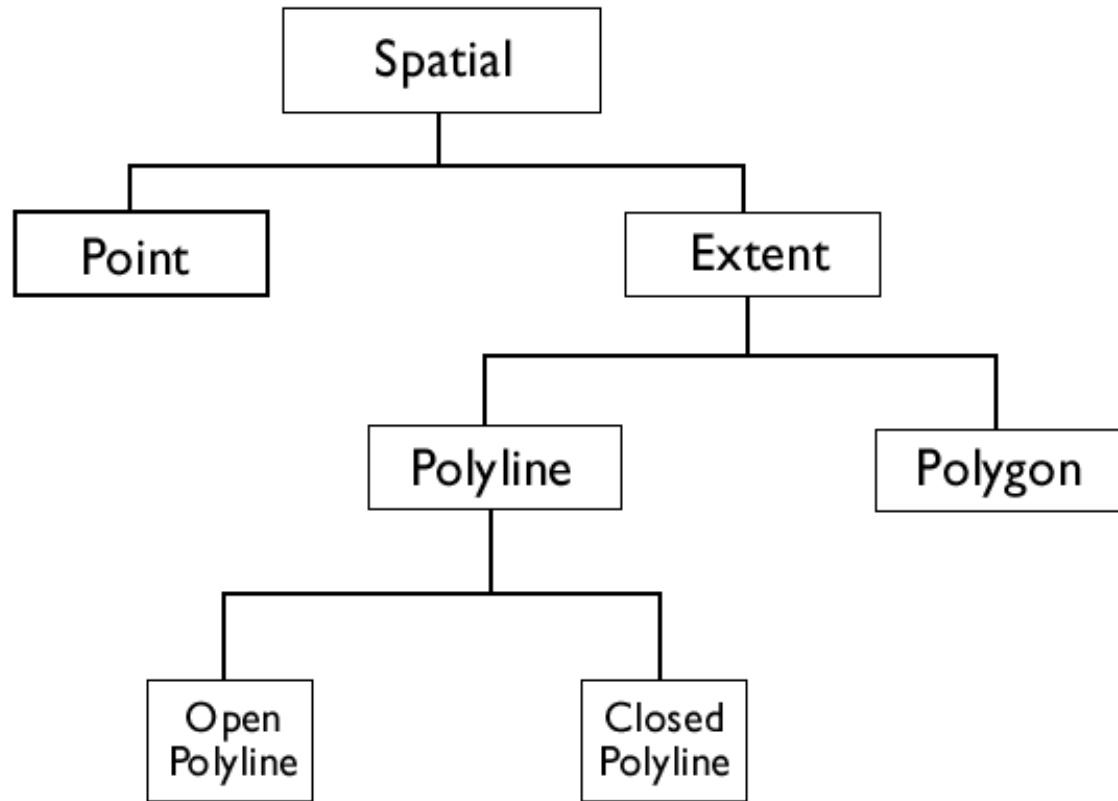
Other uses for MBR.



# Object storage

- ✦ Every object in a GIS is not defined individually. That would be a painstaking task.
- ✦ Groups of like objects are organized into classes. Each class and sub-classes have attributes which apply to the entire group.
- ✦ In addition, operations or methods that describe possible pertinent actions, can be defined with reference to the class. Attributes and procedures are bequethed through a hierarchical system of inheritance.

# Object Class Inheritance Hierarchy



Spatial is the most general class. It can then be divided into point or extent (sets of point like polygons). Note that spatial extents are further sub-divided into 1 and 2 dimensional objects. 1 dimensional objects (polylines) can be open or closed. The latter has the start and end points joined (but it is not a polygon because of its dimensionality).

# What is an algorithm?

- ✦ The specification of the computational processes used to perform an operation (in GIS).
- ✦ We write algorithms in text initially or using flowcharts and later **encode** them.
- ✦ They are encoded in high-level languages and later translated into machine instructions by a compiler.

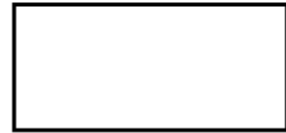
# What is a flowchart/cartographic model?

- ✦ A flowchart is a graphic representation of the algorithm. It shows the flow of processing from the beginning to the end of the solution.

# Basic flowchart symbols



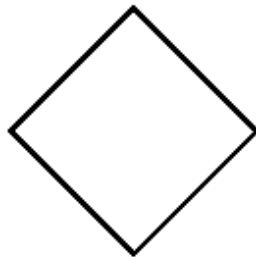
**Terminal** -- Indicates the start and the end of a module. The name of the module is used at the start. Use the word Stop or End in the end module.



**Processing** -- Indicates any sort of processing including calculations, moving fields around in memory or incrementing counters.



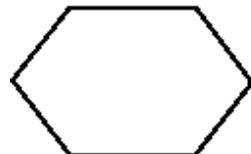
**Input/output** indicator



**Decision** -- Indicates a logical (true or false) query or a relational query (>, < or =). We use these types of queries repeatedly in GIS and therefore use this symbol in documenting algorithms in GIS.



**Predefined process module** -- This is a module that contains at least 3 related tasks that are processed in another place in the program. These modules have only one entrance and one exit.



**Automatic loop counter** -- This represents a loop with a counter that starts with the first processed entry and is incremented by a pre-determined value until it reaches its pre-determined end value.

# Rules for flowcharting

1. All boxes in the chart are connected with arrows (not plain lines)
2. Each flowchart symbol has a data entry point on top of its symbol with no other entry points.
3. The exit point for data is always on the bottom except for decisions symbols. Decision symbols have two exit points - on either side or on the bottom and one side.
4. The flow of data is generally from top to bottom.

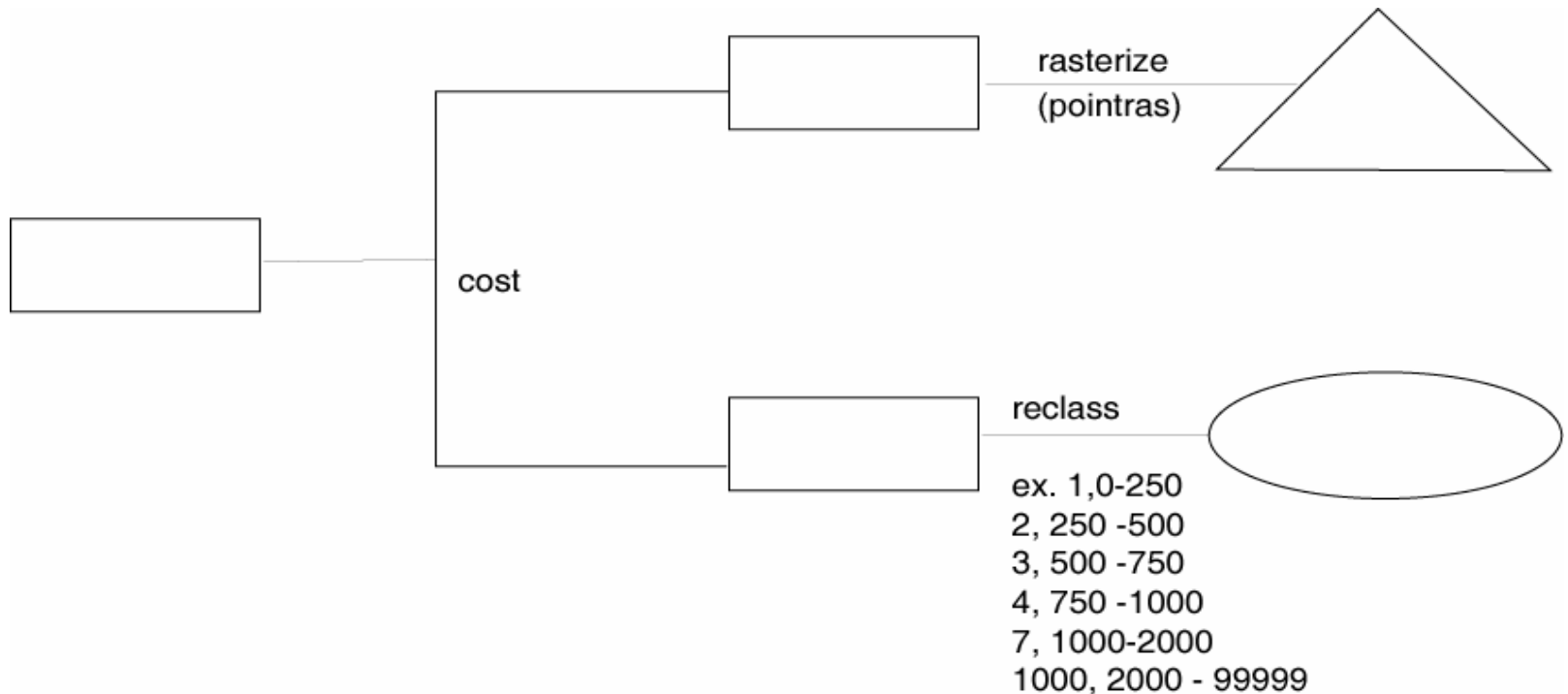
5. Connectors are used to connect breaks in the flowchart, like from one page to another or from the bottom of one page to the top of another. They generally look like this: O
6. Subroutines have their own flowcharts, independent of the main one.  
All flowcharts start with a Terminal symbol. They also end with one.

# Algorithms in IDRISI

- ✦ Algorithms are expressed as cartographic models in IDRISI.
- ✦ IDRISI cartographic modelling is designed to help plan and structure analysis, but also serves the purpose of documentation.
- ✦ Documentation is an important, but under-executed step in programming.



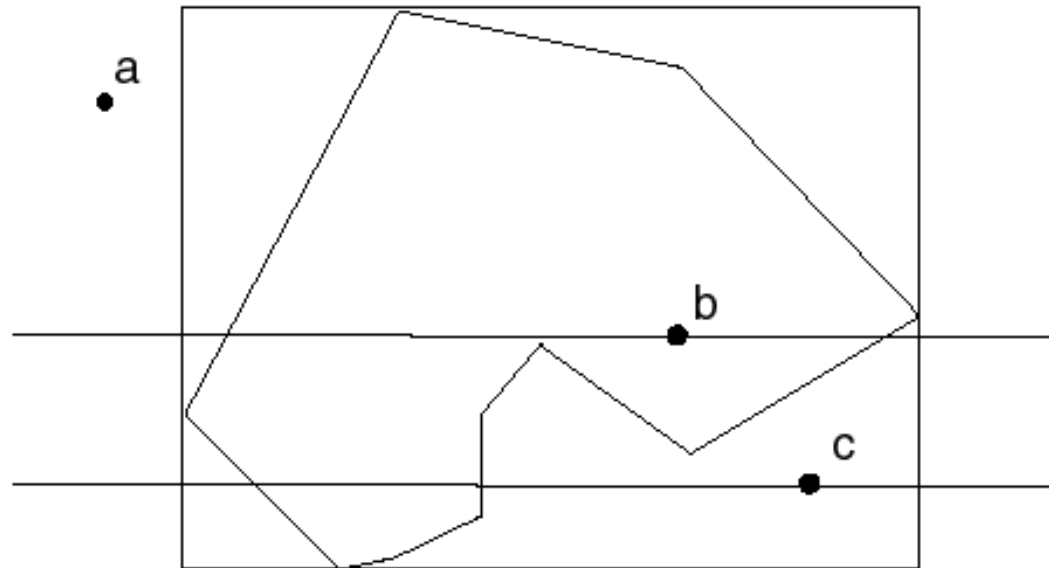
Squares are used for file names; triangles for vector files; raster files by rectangles; values files are ovals; and tabular data (like excel data) is represented by a page with the corner turned down.



# Geometric algorithms

- ✦ Used for many operations in GIS.
- ✦ They often have to deal with “special cases”
- ✦ What looks like a trivial problem is often more complex
- ✦ Classic case of point in polygon procedure.
- ✦ Easy for humans to see the solution.

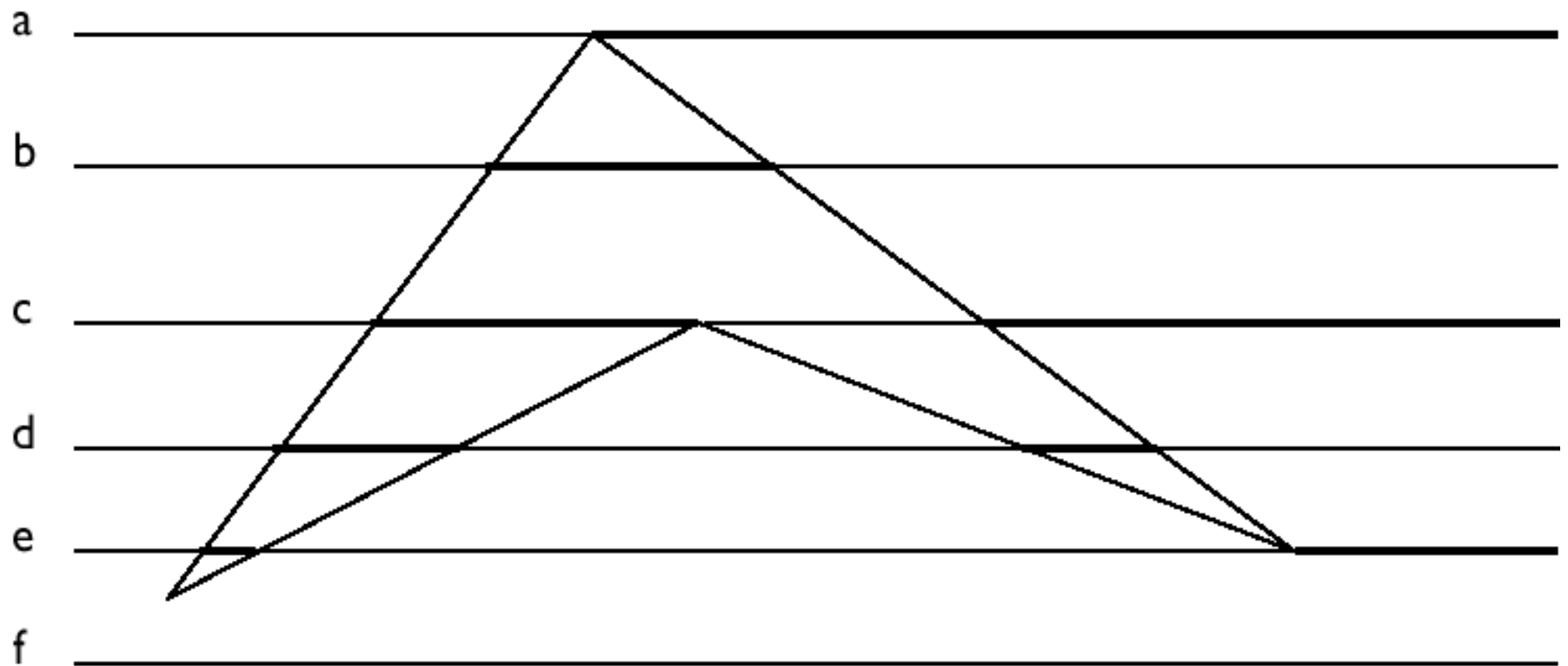
1. Draw a square around the polygon extent.
2. Compare coordinates of point with square vertices.  
This will eliminate many points such as A below.
3. Draw a horizontal line through the point and the square boundary.
4. Count the intersections between the line and the polygon.  
Odd number of intersections = inside the polygon.



*Point in polygon searches*

# Polygon painting

- ✦ Scan conversion is used to display lines on a CRT or LCD monitor.
- ✦ Electron beam scans the lines of the screen, and is turned off when it encounters the edge of lines.
- ✦ *Parity* is used to keep track of inside/outside of polygon position in current beam.
- ✦ Every time the beam crosses the boundary of a polygon, its on-off switch toggles (binary system)

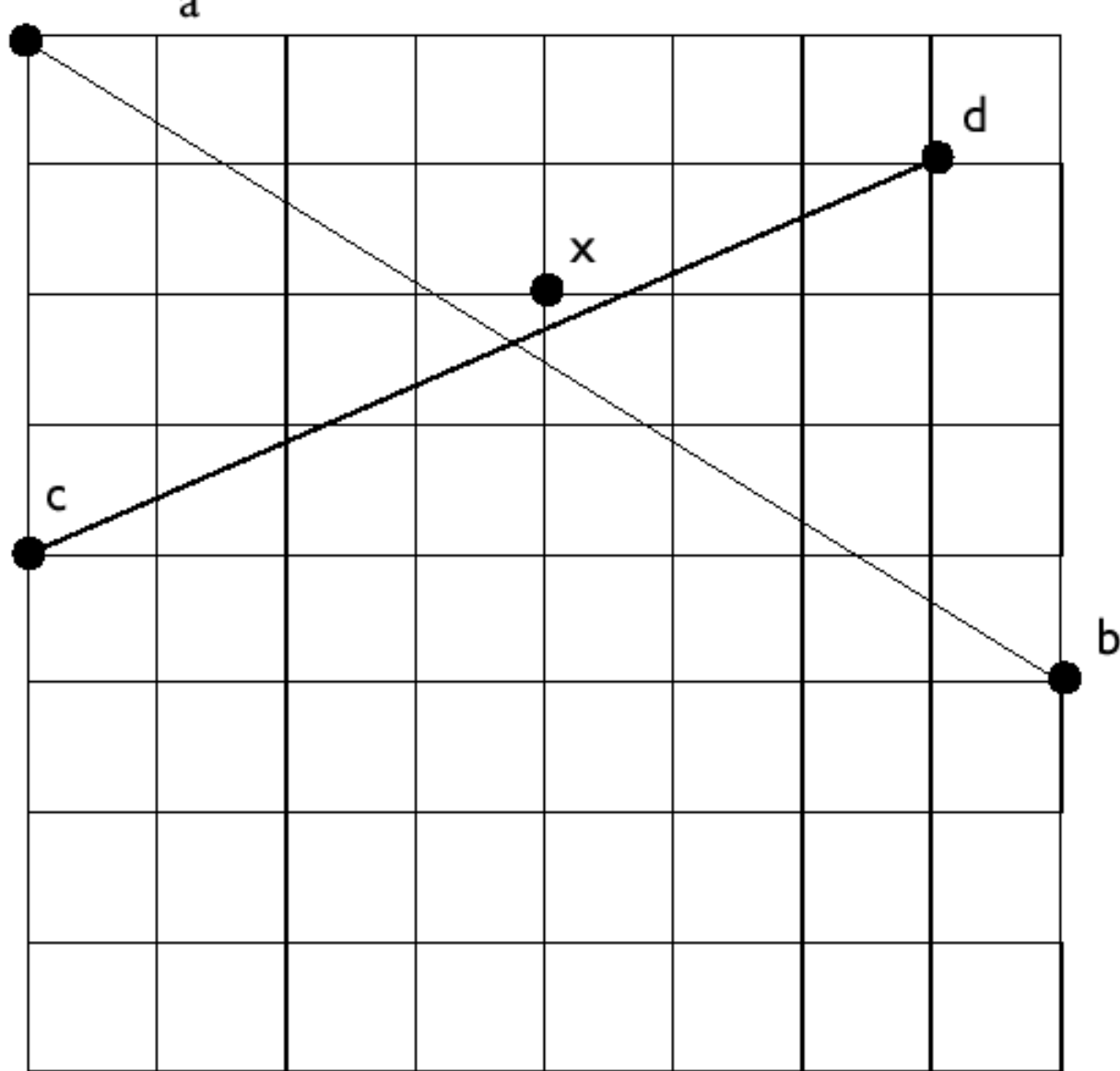


**Parity approach** to area filling -- will likely work for scan lines b and d but may fail for a, c, e, and f.

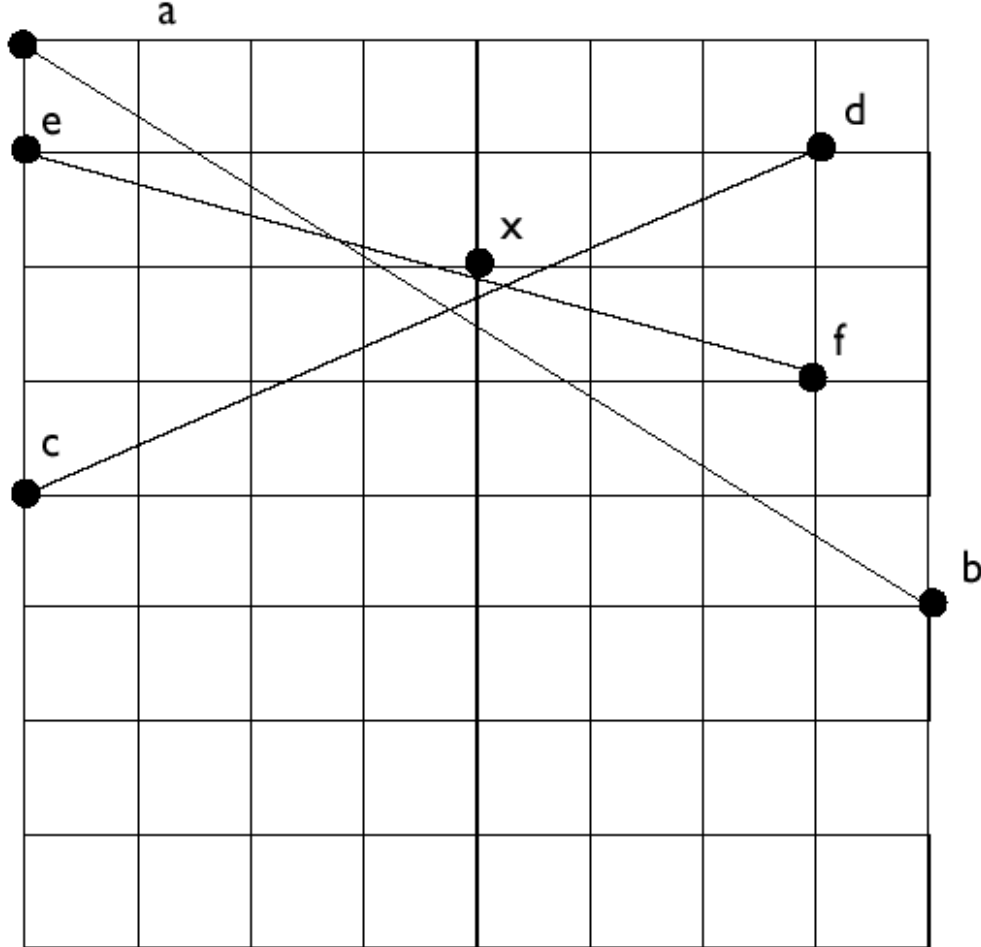
# Representational algorithms

- ✦ GIS is conducted primarily in Euclidean space, but fails to fulfill assumptions of continuity associated with Real numbers that populate Euclidean space.
- ✦ Problem of discretization.
- ✦ There are algorithms that circumvent problems of discretization.
- ✦ Example: intersection not at a point on the grid.
- ✦ What do we do?

Can't add the point to the pointset because it is not a domain grid point.

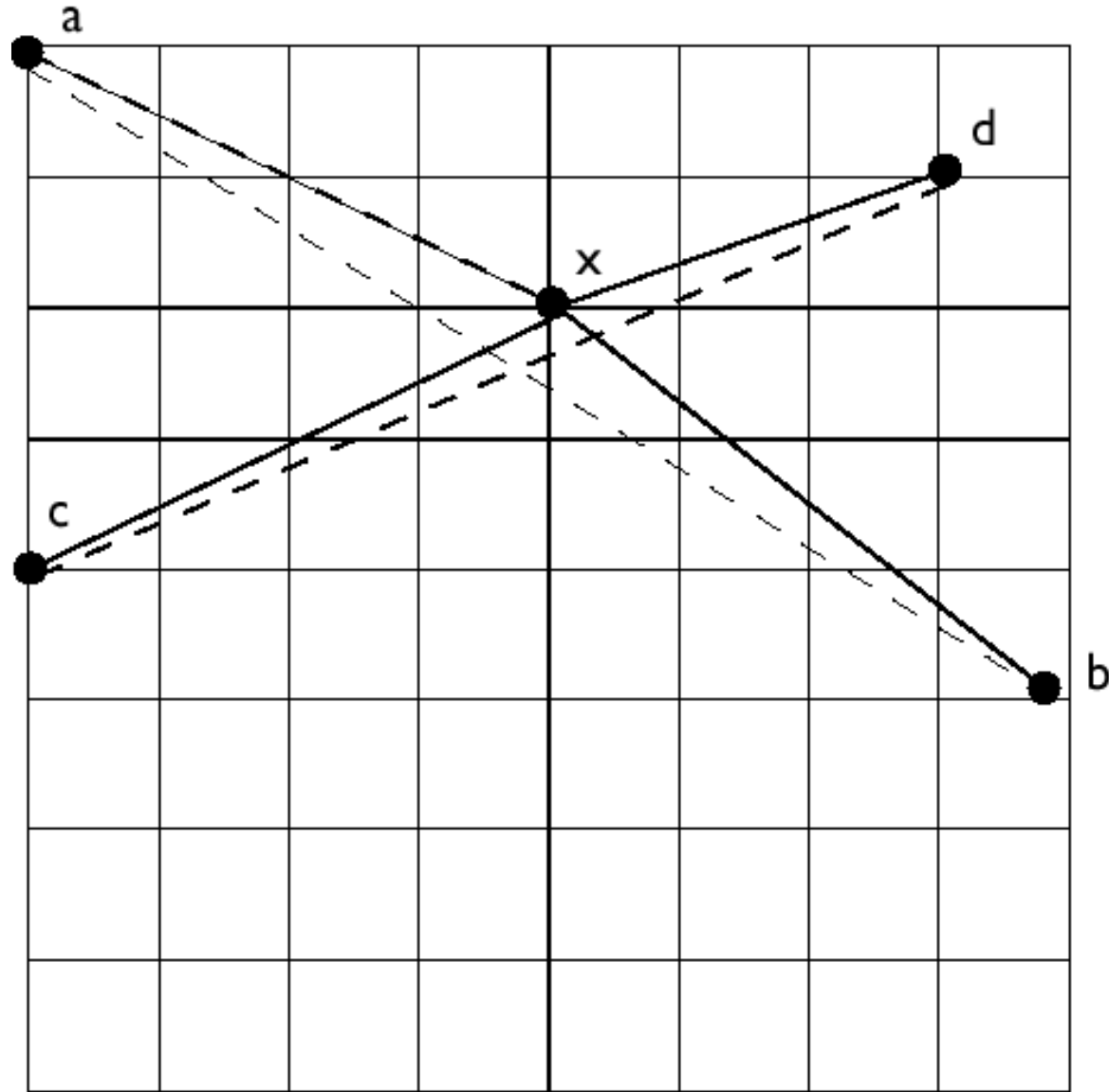


The nearest grid point to the intersection of lines  $ab$  and  $cd$  is  $x$ . The actual intersection point is not on the discretized plane in  $\mathbb{R}^2$ .

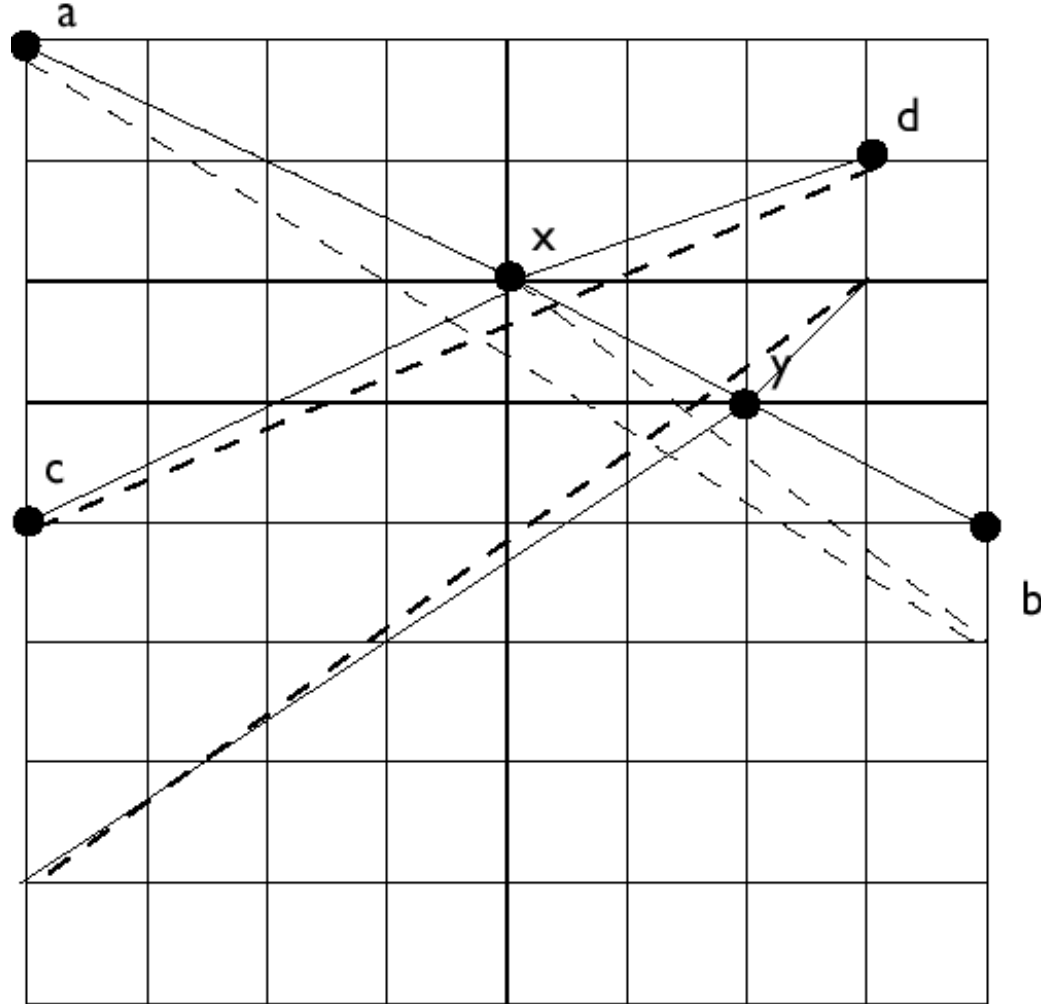


Floating point of intersection. The point of intersection of  $ab$  and  $cd$  is actually below  $ef$  but since we re-allocated it to the set of domain grid points, it is above  $ef$ . This causes problems for topological description and subsequent queries. This has special implications in GIS because we describe entities in *relation* to each other.

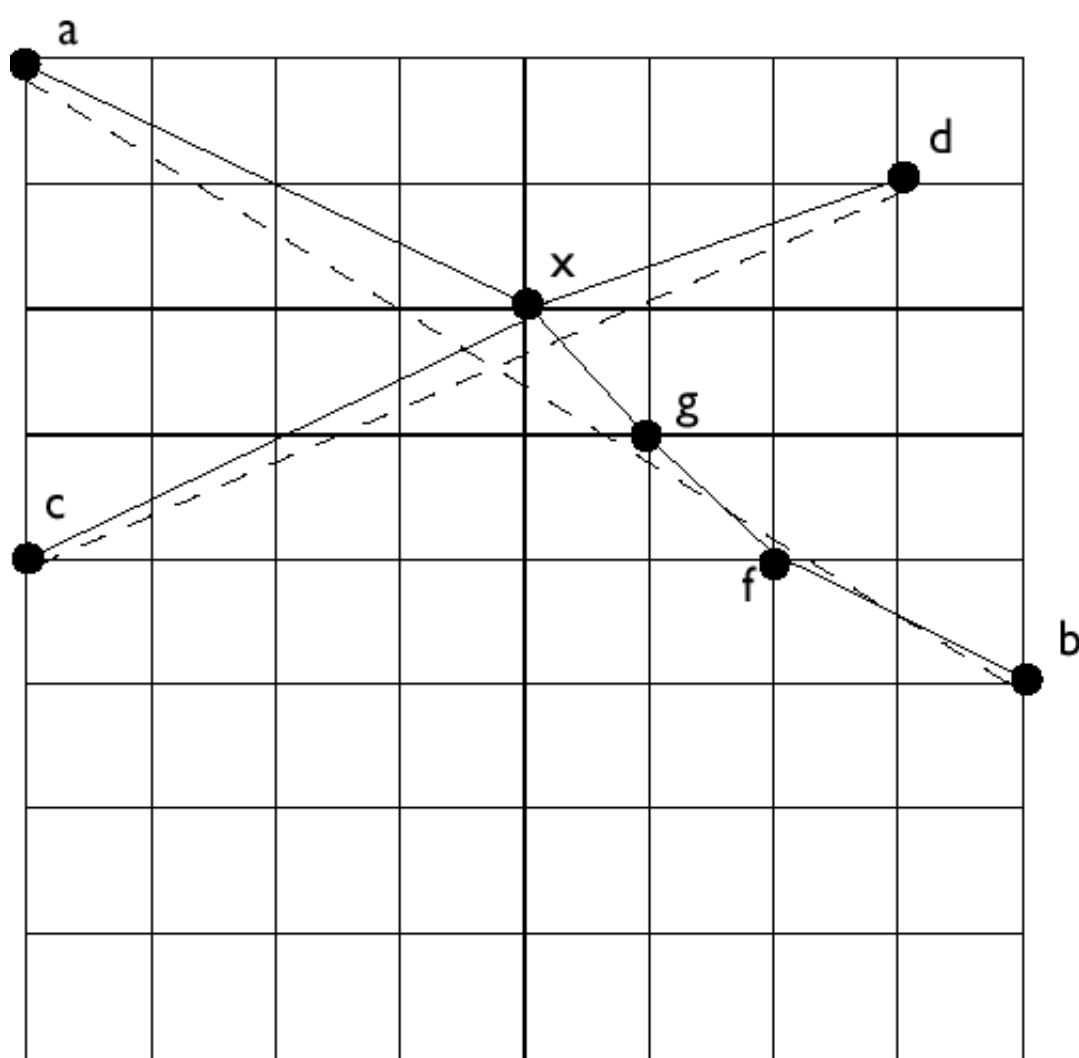




Rounding the point of intersection. The intersecting lines are moved to include the domain grid point  $x$ . The original lines are now segments  $ax$ ,  $xb$ ,  $cx$  and  $xd$ .



More intersections lead to further shifts of line *ab*. In a dynamic situation, more intersections will also cause shifts of subsequent lines that relate to *ab* so that, despite retaining the rounded point *x*, new lines shift way out of position *and* *ab* also shifts farther away from its original position.



The Greene-Yao (1986) algorithm uses the peg board metaphor to envisage the grid domain. There is a peg at every domain point. The elastic band lines between two points can push against but can't pass over any of the pegs.

# Simple vector algorithms for analysis

✦ **Distance from a point to a line.** In GIS, we always mean the **minimum** distance between two entities when we talk about distance.

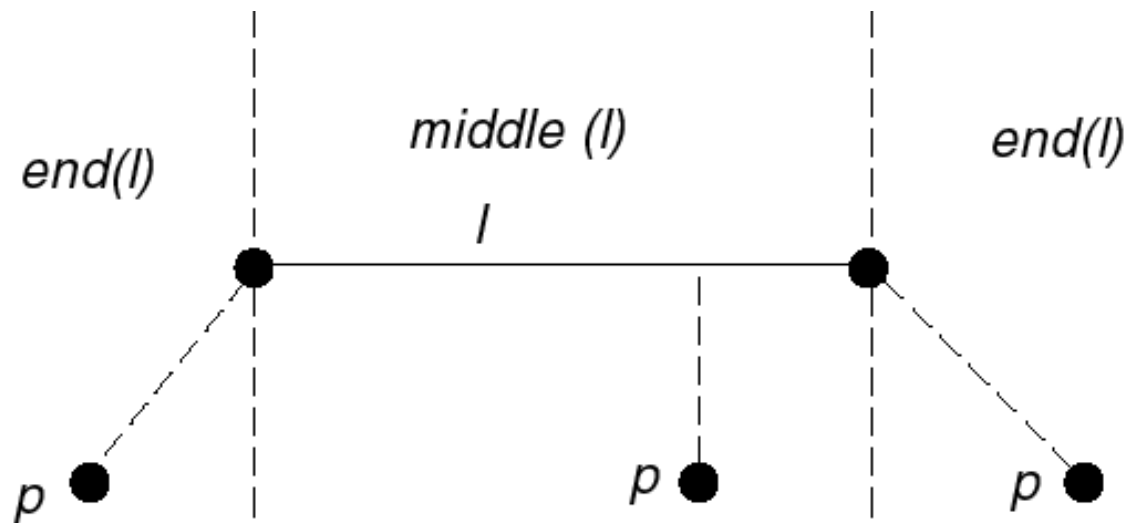
✦ Suppose that the point  $p$  is given by the coordinate pair  $(X, Y)$ ; the line  $l$  is described as

$\{(x, y) \mid ax + by + c = 0\}$  (this is a formal definition and peripheral to this explanation)

✦ then the distance from  $p$  to  $l$  is given by the formula:

$$\mathbf{distance} (p, l) = |aX + bY + c| / \sqrt{a^2 + b^2}$$

✦ the distance is, in effect, the distance from  $p$  to  $l$  measured by the length of the line segment through  $p$  and orthogonal to  $l$ .

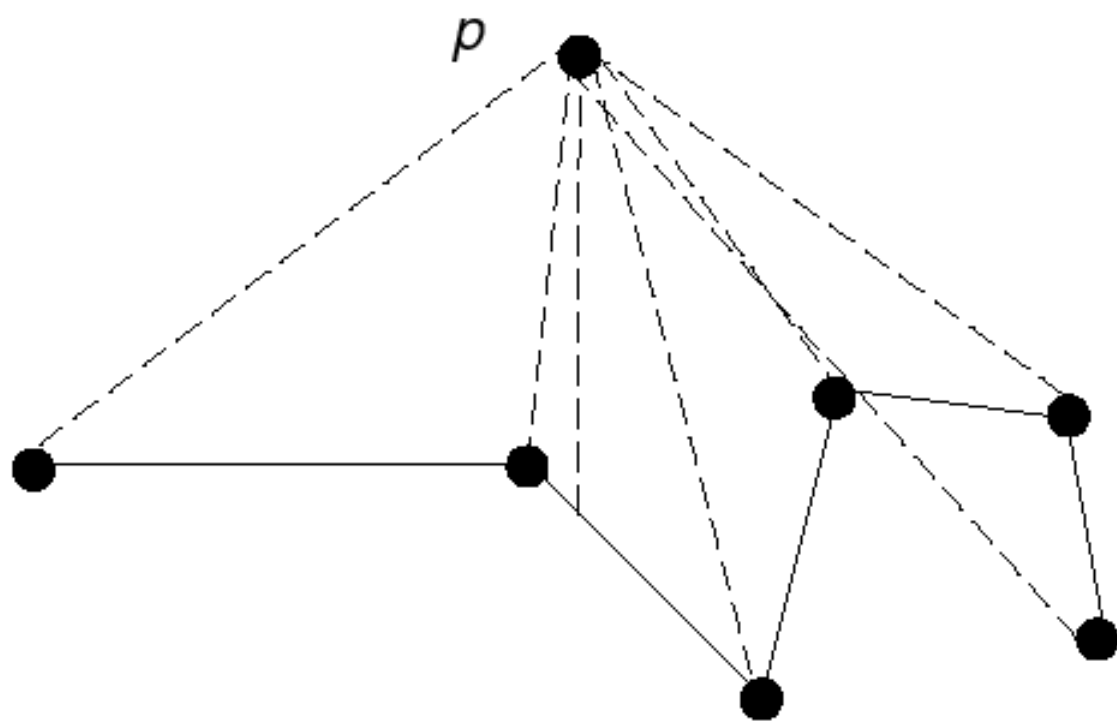


*Distance between a point and a line segment.*

The distance is measured between the point and one of the line's endo points or from the point to the line's middle, if  $p$  is orthoganal to the line.

We consider that the line divides the plane into two different pointsets -- the set of points that are connected: (i) the area corresponding to the middle of the line -- *middle (l)*; and (ii) and those that are disconnected -- *end(l)*.

The calculation depends on whether it is from the point to *middle (l)* or to *end(l)*.



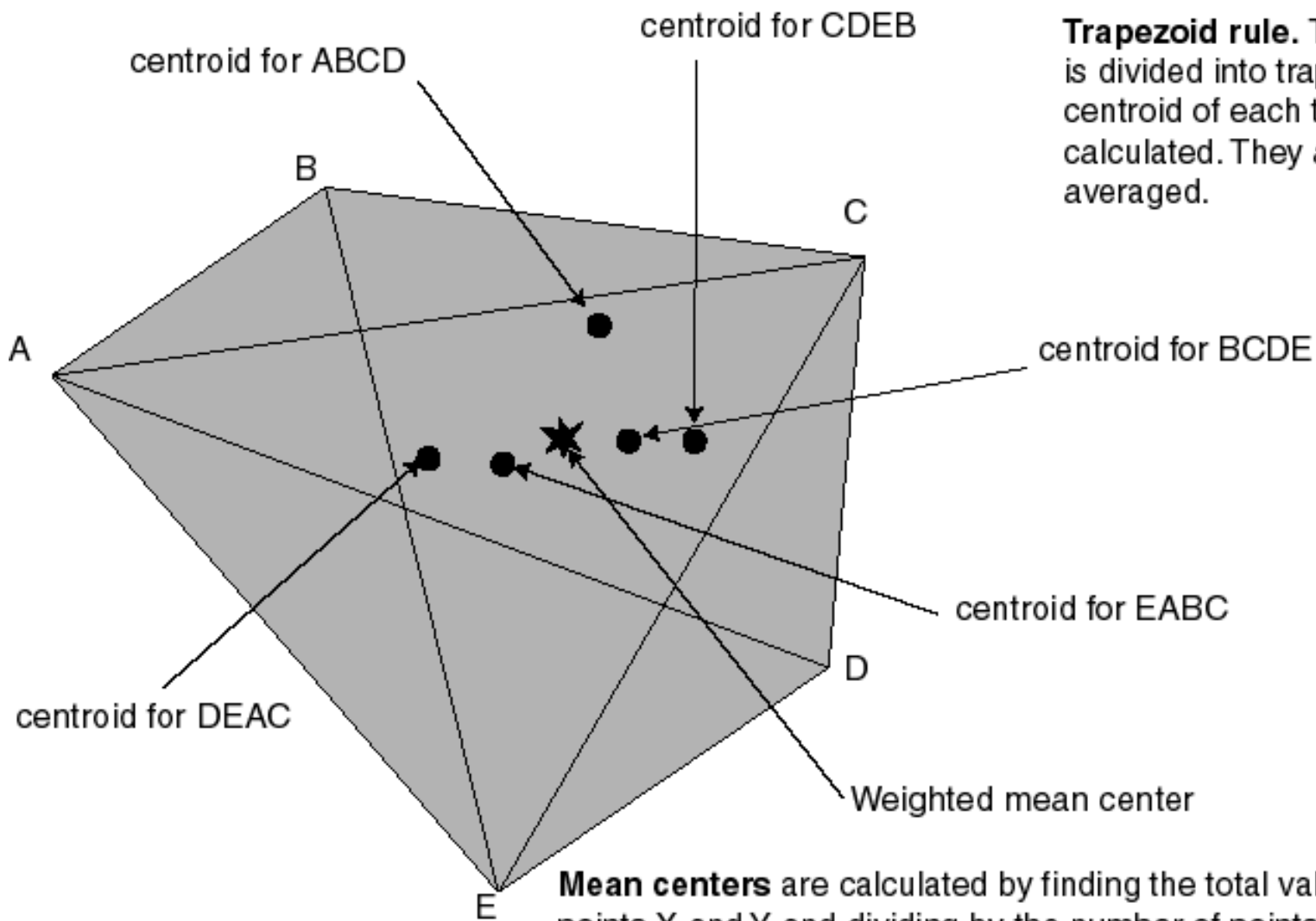
*Distance between a point and a polyline.*

The distance is measured from the point to the polyline and from the point to each of the polyline vertices. The shortest of these distances is the designated distance.

# Polygon to polygon distance

- ✦ Distance between polygons can be calculated as the distance between their closest points .
- ✦ But usually, the distance between polygons is determined by distance from their centroids.
- ✦ Need to know how to calculate a centroid.





**Trapezoid rule.** The polygon is divided into trapezoid. The centroid of each trapezoid is calculated. They are then averaged.

**Mean centers** are calculated by finding the total values for all points X and Y and dividing by the number of points.  
**Weighted means** imply that another variable (than just geographical area) is taken into account. The density of vegetation available for deer grazing, for instance, could be multiplied by each X and Y for which there is data. Total values for X and Y are then tallied and divided by the number of points.

- ✦ The centroid is easily calculated: Just add all the X values of vertices and then all the Y vertices in the coverage and divide each by the number of (x,y) points.
- ✦ Often centroids are **weighted** or moved to the center of gravity of the **variable** rather than the area.

## Distance between centroids.

- ✦ What we are actually concerned with is how to calculate the distance *between* centroids. The distance between centroids then becomes the distance between the two centroid points.