Maps as Numbers: Data Models

vertices

nodes

decode

S – Start node

E – End node

arcs

9/10/2019

Geo327G/386G: GIS & GPS Applications in Earth Sciences
Jackson School of Geosciences, University of Texas at Austin
The Task

- An accurate, registered, digital map that can be queried and analyzed

Translate:

Real World Locations, Paper Map ➔ Computer Files

Spatial Data Models, Topology

Entity Info. ➔ Queriable Database Files

Relational or Object-Oriented Databases

Relate Spatial Coordinates to Entity Info.

“Spatial DBMS” software = GIS software!
Data Models

How is reality abstracted and codified?

- Reality
  - Wells produce from rocks that contain oil and gas

- Conceptual Models
  - Wells are points, rock units are polygons (both are objects)
  - Well A penetrates Fm. 1; produces oil. Well B penetrates Fm. 3; produces gas. Fm 3 overlies Fm. 1.

- Logical Models
  - Store well locations with a particular file structure, relationships in a dBase table. Associate table with location.
Characterized all features or phenomenon as:

- **Discrete objects**; e.g. wells, roads, rock bodies, etc.
  - **Object-based models**

- **Continuous phenomena**; e.g. gravity, magnetic intensity, topography, temperature, snowfall, soil pH, etc. = “fields” of values
  - **Field-based models**

- Organize objects and fields by a common theme; e.g. geology, hydrography, transportation
  - **Thematic layers**
Logical Models

- **VECTOR MODEL**
  - Discrete objects are represented by points and vectors, continuous fields by irregular tessellations of triangles (TINs)

- **RASTER MODEL**
  - Discrete objects and continuous fields are represented by an array of square cells (pixels)
How should discrete objects be coded?

Raster Model

Vector Model
Vector Model

Logical Models

AREAS
(Polygons)
consisting of...

LINES
(Arcs)
consisting of...

POINTS
consisting of...

COORDINATES

\((x, y)\)

\((1, 5)\)

\((5, 1)\)

\((7, 2)\)

\((5, 7)\)

\((3, 8)\)
Continuous Phenomena As Surfaces

- **Raster Topography**
  - Regular tessellations, e.g. DEM, DTM

- **Vector Topography**
  - Irregular tessellations, e.g. T.I.N.
Simple Vector Data Structure

Vector Line

P1  P2  P3  P4  P5

Table of Points

<table>
<thead>
<tr>
<th>ID</th>
<th>X</th>
<th>Y</th>
<th>Node</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>503200</td>
<td>3200522</td>
<td>From</td>
</tr>
<tr>
<td>P2</td>
<td>503250</td>
<td>3200522</td>
<td></td>
</tr>
<tr>
<td>P3</td>
<td>503300</td>
<td>3200460</td>
<td></td>
</tr>
<tr>
<td>P4</td>
<td>503245</td>
<td>3200410</td>
<td></td>
</tr>
<tr>
<td>P5</td>
<td>503350</td>
<td>3200410</td>
<td>To</td>
</tr>
</tbody>
</table>

(in UTM coordinates)
Simple Raster Data Structure:

Raster Line

Equivalent Binary “Flat File”
(Plus: Raster dimension, resolution and location)

```
1100000
0010000
0001000
0010000
0111111
```
Vector Models (Raster Next Time...)

- “Graphical”
- Topologic/georelational
- T.I.N.
- Network
“Graphical” Vector Model

- Lines have arbitrary beginning and end, like spaghetti on a plate

- Common lines between adjacent polygons duplicated

- Can lead to “slivers” of unassigned area = “sliver polygons”
“Graphical” Vector Model

- Shortcomings for maps:
  - No real world coordinates required
  - No identification of individual objects; no way to attach attributes
  - Details of relationships among object (e.g. what’s adjacent) not stored, but needed for spatial analysis
Graphical Vector Structure

- Contains no explicit information about adjacency, containment or contiguity i.e.
  - Which polygons are adjacent?
  - Which polygons are contained within other polygons?
  - Which lines are connected? Where are they connected? Where do lines begin and end?

= “Spaghetti Data Model“
Topological Vector Model

- Store pts. as x,y geographic coordinates
- Store lines as arcs of connected pts.
- Store polygons as closed paths

Also explicitly store ....

- Where lines start and end (connectivity)
- Which polygons are to the right and left side of a common line (adjacency)
Topology

- The geometric relationship(s) between entities (e.g. points, lines, areas); where is one thing with respect to another?
Topological Properties

- Spatial characteristics that are unchanged by transformations like scaling, rotation and translation are topologic
- **Non-topological**: x, y coordinates, area, distance, orientation
- **Topological**:
  - **Contiguity** – what’s adjacent
  - **Connectivity** – what’s connected
  - **Containment** – what’s inside or outside of a region
Topological Properties

- **Contiguity:**
  - Adjacency

- **Connectivity:**
  - What’s connected

- **Containment:**
  - What’s inside or outside of a region

- Unchanged by translation, scaling, rotation
Maintaining Topology: Planar Enforcement

- One and only one feature at every x, y location
  - Lines cross at nodes; polygons space-filling, exhaustive, mutually exclusive (no overlaps or gaps)
  - Sum of the area of all individual polygons equals the area of extent of all polygons
  - Common boundaries stored only once
- A PLANAR GRAPH meets these conditions
- Allows spatial queries for adjacency, containment and rapid what-is-where
- All raster data is of this sort
Non-Planar vs. Planar Graphs

- **Spaghetti**

- **Topologic**

<table>
<thead>
<tr>
<th>Polygons</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Survey A</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Survey B</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Survey C</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>None</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

after Bonham-Carter, 1994
Lines: Graphic vs. Topologic

- **Graphic (Spaghetti)**
  - Vertices
  - Overshoot ("dangle")
  - Table of \((x,y)\) coordinates

- **Topologic (with meatballs)**
  - Vertices
  - Nodes
  - Arcs
  - Table of \((x,y)\) coordinates
  - Table of arcs with IDs, starting and ending nodes
**Lines: Arc-Node Topology**

**Vertex Table**

<table>
<thead>
<tr>
<th>ID</th>
<th>x</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>3</td>
<td>5</td>
</tr>
</tbody>
</table>

**Node Table**

<table>
<thead>
<tr>
<th>ID</th>
<th>x</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Arc Table**

<table>
<thead>
<tr>
<th>ID</th>
<th>FID</th>
<th>F Node</th>
<th>T Node</th>
<th>Vertices</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100</td>
<td>1</td>
<td>2</td>
<td>1, 2</td>
</tr>
<tr>
<td>2</td>
<td>102</td>
<td>3</td>
<td>2</td>
<td>3, 4, 5, 6, 7</td>
</tr>
<tr>
<td>3</td>
<td>103</td>
<td>3</td>
<td>4</td>
<td>null</td>
</tr>
</tbody>
</table>

F = “Start” node (F: “From” node)
T = “End” node or (T: “To” node)
Polygons: Polygon-Arc Topology

Arc Table

<table>
<thead>
<tr>
<th>Arc ID</th>
<th>L. Poly</th>
<th>R. Poly</th>
<th>F Node</th>
<th>T Node</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>World</td>
<td>P1</td>
<td>N1</td>
<td>N2</td>
</tr>
<tr>
<td>A2</td>
<td>P1</td>
<td>P2</td>
<td>N2</td>
<td>N1</td>
</tr>
<tr>
<td>A3</td>
<td>P2</td>
<td>World</td>
<td>N2</td>
<td>N1</td>
</tr>
</tbody>
</table>

Polygon Table

<table>
<thead>
<tr>
<th>Poly ID</th>
<th>FID</th>
<th>Arcs.</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>100</td>
<td>A1, A2</td>
</tr>
<tr>
<td>P2</td>
<td>102</td>
<td>A2, A3</td>
</tr>
</tbody>
</table>

Arc Coordinates Table

<table>
<thead>
<tr>
<th>Arc</th>
<th>Start</th>
<th>Vertices</th>
<th>End</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>N1</td>
<td>v7, ..., v11, ...</td>
<td>N2</td>
</tr>
<tr>
<td>A2</td>
<td>N2</td>
<td>..., v8</td>
<td>N1</td>
</tr>
<tr>
<td>A3</td>
<td>N2</td>
<td>v1, v2, ..., v6</td>
<td>N1</td>
</tr>
</tbody>
</table>
Why Bother With Topology?

- Provides a way of error trapping and geometry validation after data entry
  - All lines must meet at nodes, all polygons must close, polygons can’t overlap, all lines in a network must join
- Permits spatial queries, precise measurements
What Kind of Queries Does Topology Permit?

- **Connectivity**
  - What is shortest path between features or locations? (networks, flow)
  - Find all fault trace intersections

- **Contiguity**
  - What’s adjacent: e.g. Show all granite/limestone contacts
  - Combine all contiguous units with a specific attribute (e.g. lithology) into a single unit

- **Containment (= “Area Definition”)**
  - What proportion of an area is underlain by a specific rock type?
  - What is spatial density of specific feature(s)?
Logical Models

Vector Models

- Graphical ✓
- Topologic/"georelational" ✓
- T.I.N.
- Network
Triangulated Irregular Network - TIN

- Topological 3-D model for representing continuous surfaces using a tessellation of triangles
Triangular Irregular Network

- Network ("tessellation") of edge-sharing triangles made from irregularly spaced points with x, y and z values
- Density of triangles varies with density of data points (e.g. spacing of contours) - c.f. raster with uniform data density – advantages for file size
- Triangle sides are constructed by connecting adjacent points so that the minimum angle of each triangle is maximized (see “Delaunay Triangulation” for details); a “fat” triangle, not a “sliver” triangle.
- Can render faces, calculate slope, aspect, surface shade, hidden-line removal, etc.
- Practical limit for computation on desktop is ~ 10-15 million nodes
How Are Triangle Created?
Find the Delaunay Triangulation

- Find the set of circumcircles such that no point lies within a circumcircle
  - Circumcircle is the circle that passes through all 3 corners of a triangle
  - For 4 or more points on the same circumcircle (e.g. a rectangle) the D. Triangulation is not unique
  - For a set of points on a line, the D. Triangulation is degenerate (no triangle)
- D. Triangulation avoids sliver triangles – better represents average slopes and aspects
TIN Topology

After Zeiler, Modeling our World, p. 165
TIN for Seiad Valley, CA

- Triangle edges symbolized
- Faces symbolized for elevation & aspect
3-D TIN Scenes of Seiad Valley fault
3-D TINS, Grand Canyon
Vector Models

- Graphical ✓
- Topologic/”georelational” ✓
- T.I.N. ✓

- Network - not discussed, see Help files