18BGE66S - Fundamentals of GIS

[Syllabus Unit III:- Data Base Management System: Structure, Functions and Organizational aspects – RDBMS - GIS software: Data Storage –Analysis –Buffering –Overlay.]

A database is a collection of related data which represents some aspect of the real world. A database system is designed to be built and populated with data for a certain task.

DBMS

Database Management System (DBMS) is a software for storing and retrieving users' data while considering appropriate security measures. It consists of a group of programs which manipulate the database. The DBMS accepts the request for data from an application and instructs the operating system to provide the specific data. In large systems, a DBMS helps users and other third-party software to store and retrieve data.

DBMS allows users to create their own databases as per their requirement. The term "DBMS" includes the user of the database and other application programs. It provides an interface between the data and the software application.

History of DBMS

Important landmarks from the history:

1960 - Charles Bachman designed first DBMS system

1970 - Codd introduced IBM'S Information Management System (IMS)

1976- Peter Chen coined and defined the Entity-relationship model also know as the ER model

1980 - Relational Model becomes a widely accepted database component

1985- Object-oriented DBMS develops.

1990s- Incorporation of object-orientation in relational DBMS.

1991- Microsoft ships MS access, a personal DBMS and that displaces all other personal DBMS products.

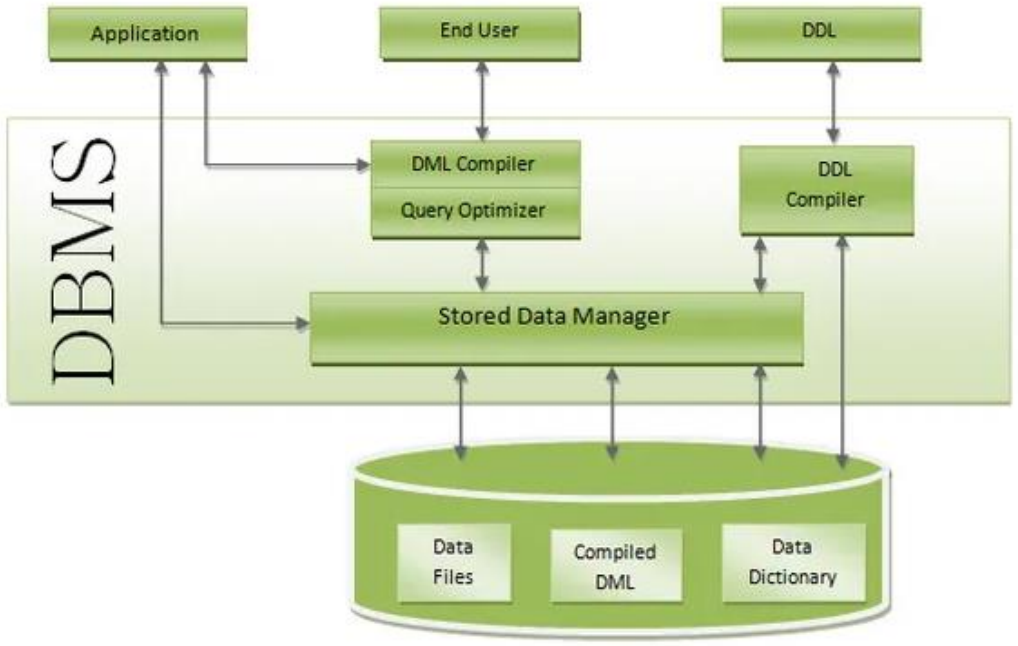
1995: First Internet database applications

1997: XML applied to database processing. Many vendors begin to integrate XML into DBMS products.

Structure of DBMS

- **Applications: –** It can be considered as a user-friendly web page where the user enters the requests. Here he simply enters the details that he needs and presses buttons to get the data.
- **End User: –** They are the real users of the database. They can be developers, designers, administrators, or the actual users of the database.
- **DDL: –** Data Definition Language (DDL) is a query fired to create database, schema, tables, mappings, etc in the database. These are the commands used to create objects like tables, indexes in the database for the first time. In other words, they create the structure of the database.
- **DDL Compiler: –** This part of the database is responsible for processing the DDL commands. That means this compiler actually breaks down the command into machine-understandable codes. It is also responsible for storing the metadata information like table name, space used by it, number of columns in it, mapping information, etc.

- **DML Compiler: –** When the user inserts, deletes, updates or retrieves the record from the database, he will be sending requests which he understands by pressing some buttons. But for the database to work/understand the request, it should be broken down to object code. This is done by this compiler. One can imagine this as when a person is asked some question, how this is broken down into waves to reach the brain!
- **Query Optimizer: –** When a user fires some requests, he is least bothered how it will be fired on the database. He is not all aware of the database or its way of performance. But whatever be the request, it should be efficient enough to fetch, insert, update, or delete the data from the database. The query optimizer decides the best way to execute the user request which is received from the DML compiler. It is similar to selecting the best nerve to carry the waves to the brain!
- **Stored Data Manager: –** This is also known as Database Control System. It is one of the main central systems of the database. It is responsible for various tasks
  - It converts the requests received from query optimizer to machine-understandable form. It makes actual requests inside the database. It is like fetching the exact part of the brain to answer.
  - It helps to maintain consistency and integrity by applying the constraints. That means it does not allow inserting/updating / deleting any data if it has child entry. Similarly, it does not allow entering any duplicate value into database tables.
  - It controls concurrent access. If there are multiple users accessing the database at the same time, it makes sure, all of them see correct data. It guarantees that there is no data loss or data mismatch happens between the transactions of multiple users.
  - It helps to back up the database and recovers data whenever required. Since it is a huge database and when there is any unexpected exploit of the transaction, and reverting the changes is not easy. It maintains the backup of all data so that it can be recovered.
- **Data Files: –** It has the real data stored in it. It can be stored as magnetic tapes, magnetic disks, or optical disks.
- **Compiled DML: –** Some of the processed DML statements (insert, update, delete) are stored in it so that if there are similar requests, it will be re-used.
- **Data Dictionary: –** It contains all the information about the database. As the name suggests, it is the dictionary of all the data items. It contains a description of all the tables, view, materialized views, constraints, indexes, triggers, etc.



Structure of Database Management System (DBMS)

DBMS Functions

1. Data Dictionary Management. ...
2. Data Storage Management. ...
3. Data Transformation and Presentation. ...
4. Security Management. ...
5. Multiuser Access Control. ...
6. Backup and Recovery Management. ...
7. Data Integrity Management. ...
8. Database Access Languages and Application Programming Interfaces.
9. Database Communication Interfaces
10. Transaction Management

Characteristics of Database Management System

- Provides security and removes redundancy
- Self-describing nature of a database system
- Insulation between programs and data abstraction
- Support of multiple views of the data
- Sharing of data and multiuser transaction processing
- DBMS allows entities and relations among them to form tables.
- It follows the ACID concept (Atomicity, Consistency, Isolation, and Durability).

- DBMS supports multi-user environment that allows users to access and manipulate data in parallel.

Popular DBMS Software

- MySQL
- Microsoft Access
- Oracle
- PostgreSQL
- dBASE
- FoxPro
- SQLite
- IBM DB2
- LibreOffice Base
- MariaDB
- Microsoft SQL Server etc.

Advantages of DBMS

- DBMS offers a variety of techniques to store & retrieve data
- DBMS serves as an efficient handler to balance the needs of multiple applications using the same data
- Uniform administration procedures for data
- Application programmers never exposed to details of data representation and storage.
- A DBMS uses various powerful functions to store and retrieve data efficiently.
- Offers Data Integrity and Security
- The DBMS implies integrity constraints to get a high level of protection against prohibited access to data.
- A DBMS schedules concurrent access to the data in such a manner that only one user can access the same data at a time
- Reduced Application Development Time

Disadvantage of DBMS

DBMS may offer plenty of advantages but, it has certain flaws-

- Cost of Hardware and Software of a DBMS is quite high which increases the budget of your organization.
- Most database management systems are often complex systems, so the training for users to use the DBMS is required.
- In some organizations, all data is integrated into a single database which can be damaged because of electric failure or database is corrupted on the storage media
- Use of the same program at a time by many users sometimes lead to the loss of some data.

- DBMS can't perform sophisticated calculations

RDBMS

Stands for "Relational Database Management System." An RDBMS is a DBMS designed specifically for relational databases. Therefore, RDBMSes are a subset of DBMSes.
A relational database refers to a database that stores data in a structured format, using rows and columns. This makes it easy to locate and access specific values within the database. It is "relational" because the values within each table are related to each other. Tables may also be related to other tables. The relational structure makes it possible to run queries across multiple tables at once.

While a relational database describes the type of database an RDMBS manages, the RDBMS refers to the database program itself. It is the software that executes queries on the data, including adding, updating, and searching for values. An RDBMS may also provide a visual representation of the data. For example, it may display data in a tables like a spreadsheet, allowing you to view and even edit individual values in the table. Some RDMBS programs allow you to create forms that can streamline entering, editing, and deleting data.

Most well known DBMS applications fall into the RDBMS category. Examples include Oracle Database, MySQL, Microsoft SQL Server, and IBM DB2. Some of these programs support non-relational databases, but they are primarily used for relational database management.
Examples of non-relational databases include Apache HBase, IBM Domino, and Oracle NoSQL Database. These type of databases are managed by other DMBS programs that support NoSQL, which do not fall into the RDBMS category.

GIS Software

GIS software lets you produce maps and other graphic displays of geographic information for analysis and presentation. With these capabilities a GIS is a valuable tool to visualize spatial data or to build decision support systems for use in your organization.

A GIS stores data on geographical features and their characteristics. The features are typically classified as points, lines, or areas, or as raster images. On a map city data could be stored as points, road data could be stored as lines, and boundaries could be stored as areas, while aerial photos or scanned maps could be stored as raster images.

Geographic Information Systems store information using spatial indices that make it possible to identify the features located in any arbitrary region of a map. For example, a GIS can quickly identify and map all of the locations within a specified radius of a point, or all of the streets that run through a territory.

In addition to the above capabilities, Maptitude implements a professional-strength relational database, a feature critical for GIS software. Attribute data may be freely joined to and detached from geographic layers and tables. Relational data manipulation is integrated with robust and powerful geoprocessing for spatial queries, polygon overlay, and other location-based analyzes. This is supported seamlessly so that data are moved easily to and from relational tables and geographic databases. In addition, the Maptitude fixed-format binary table supports 32,767 fields and 1 billion records, and has unlimited character field widths.

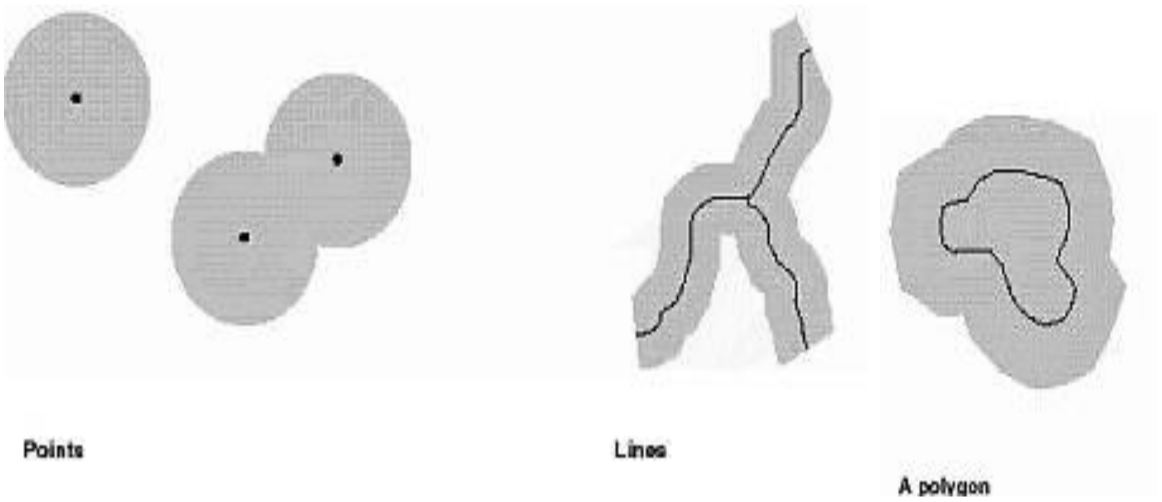
BUFFERING

The creation a zone of interest around an entity. It is an important function used to determine spatial proximity or nearness of various features by defining a distance zone around map features. Buffer can be generated for points, lines and polygons.

Based on the concept of proximity, buffering separates a map into two areas.

- One area that is within a specified distance of selected map features and the other area that is beyond.
- The area that is with in the specified distance is called the buffer zone.
- Selected map features for buffering may be points, lines, areas.
- Buffering around points creates circular buffer zones extending outward or from the polygon boundaries.
- It is used to identify a zone of interest around an entity, or set of entities.

Example: Connectivity (Vector)



Proximity Operation: Buffer Types

A GIS operation in which areas that are within a specified distance of selected map features are separated from areas that are beyond. The area that is within the specified distance is called the buffer zone. Selected map features for buffering may be points, lines, areas. Buffering around points creates circular buffer zones. Lines creates a series of elongated buffer zones. Buffering around polygons creates buffer zones extending outward or from the polygon boundaries.

Buffering as already stated, is used to identify a zone of interest around an entity, or set of entities. If a point is buffered a circular zone is created,. Buffering lines and areas creates new areas. Creating buffer zones around point features is the easiest operation , a circle of the required radius simply drawn around each point creating buffer zones around line and area features is more complicated.

GIS do this by placing a circle of the required radius at one end of the line or area boundry to be buffered.

This circle tangential to the line makes is used to define the boundry of the buffer zone. Only the most basic set of buffer operations as there are many variations on this theme. For Ex, buffer zone may be of fixed or varying width according to feature attributes. When analyzing a road network, wide buffer zones could be attached to motorways and narrower buffer zone to minor roads to reflect traffic densities.

Example: Connectivity (Vector)
Proximity Operation - Buffers & Setbacks

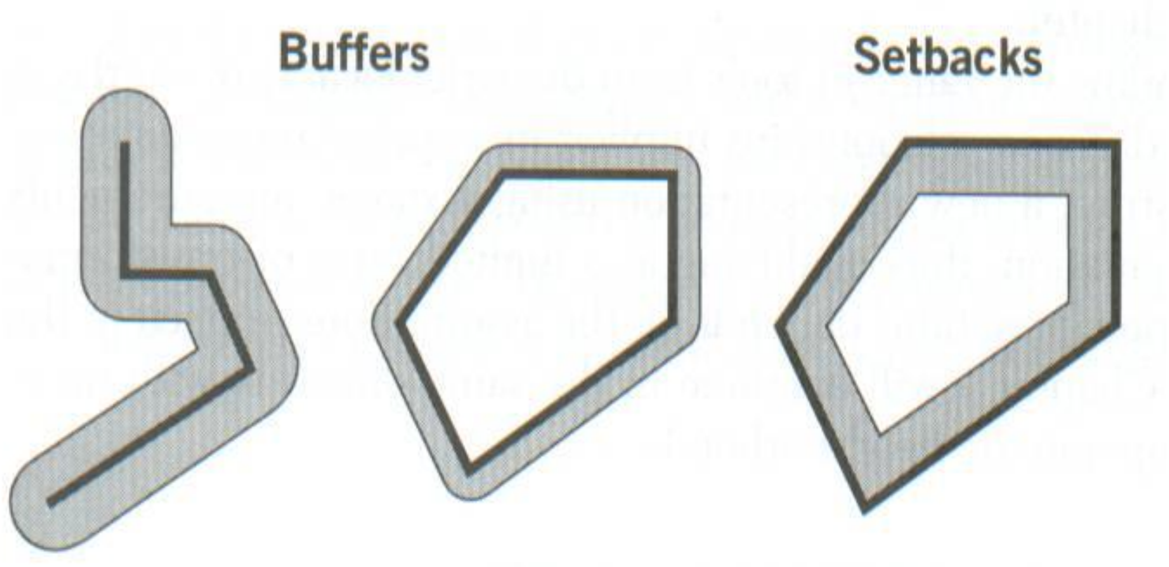


Diagram of simple buffers and a setback.
NOTE: Buffers go outward from lines or areas; setbacks run inside of areas (not lines).

Map Overlay Methods

The ability to integrate data from two sources using map overlay is perhaps the key GIS analysis function. Using GIS it is possible to take two different thematic map layers of the same area and overlay them on one top of the others to form a new layer. The techniques of GIS map overlay may be linked to sieve mapping, the overlaying of tracings of paper maps on a light table. Map overlay has its origins in the work of McHarg (1969), who used large and complex manual map overlays in landscape architecture. Map overlay has many applications. At once it can be used for the visual comparison of data layers.

As with many other operations and analysis in GIS there are differences in the way map overlays are performed between the raster and vector worlds. In vector based systems map overlay is time-consuming, complex and computationally expensive. In raster based systems it is just the opposite – quick, straightforward and efficient.

Map overlay

Map overlay combines the geometry and attributes of two feature maps to create the output.(i.e.,) overlay where new spatial data sets are created involving the merging of data from two or more input data layers to create a new output data layer. The number of map features of the output map is not the sum of map features on the input and overlay maps but is usually much larger than the sum. Each map feature on the output contains a combination of attributes from the input and overlay maps. Two methods of overlay

1) Vector overlay
2) Raster overlay

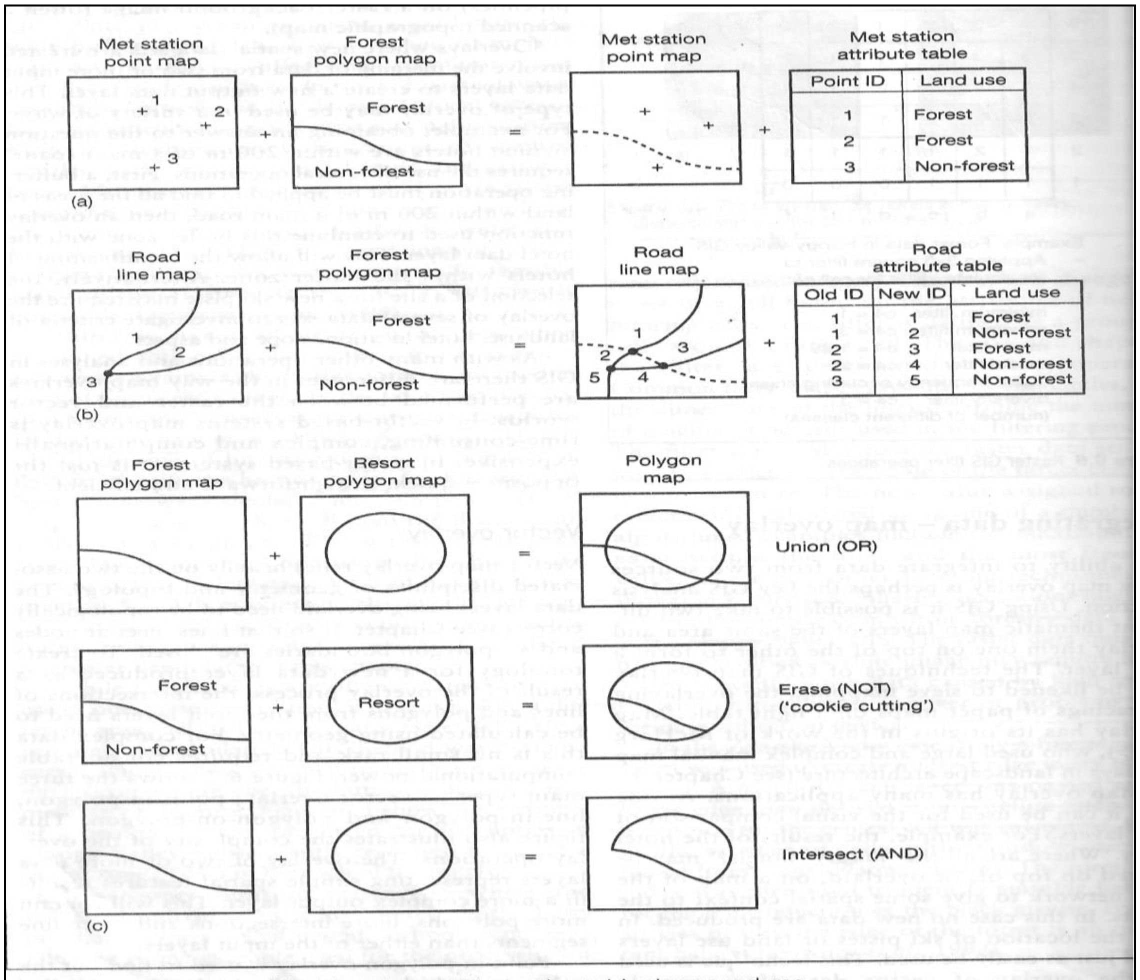
Vector overlay

Vector map overlay relies heavily on the two associated disciplines of geometry and topology .The data layers being overlaid need to be topologically correct so that lines meet at nodes and all polygon boundaries are closed .To create topology for a new data layer produced as a result of the overlay process, intersections of lines and polygons from the input layers need to be calculated using geometry. In vector based systems overlay is time consuming, complex and computationally expensive. For complex data this is no small task and requires considerable computational power.

Types of vector overlay

There are mainly three types of vector overlay

Point-in-polygon
Line-in-polygon
Polygon-on-polygon



Point ID	Land use
1	Forest
2	Forest
3	Non-forest

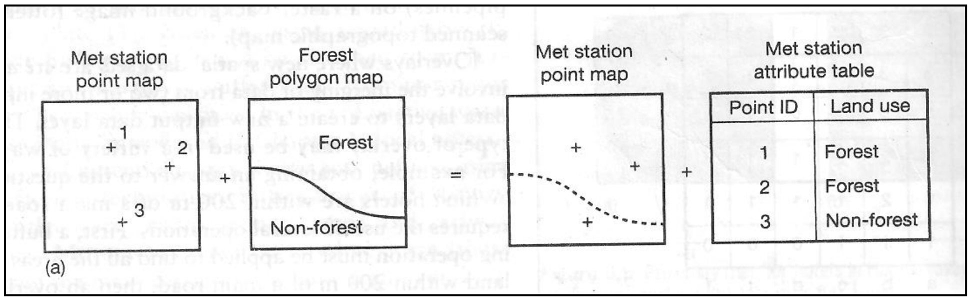
Old ID	New ID	Land use
1	1	Forest
1	2	Non-forest
2	3	Forest
2	4	Non-forest
3	5	Non-forest

Figure 6.7 Vector overlays: (a) point-in-polygon; (b) line-in-polygon; (c) polygon-on-polygon

(Fig -1)

Point-in-polygon

In a point-in-polygon operation, the same point features in the input are included in the output but each point is assigned with attributes of the polygon with in which it falls. In the happy valley GIS meteorological stations are represented as points and land use as polygons. Using point-in-polygon overlay on these vector data layers it is possible to find out in which land use polygon each meteorological station is located. Fig 2 illustrates this overlay process. On the out put map a new set of rain gauge points is created with additional attributes describing land use.

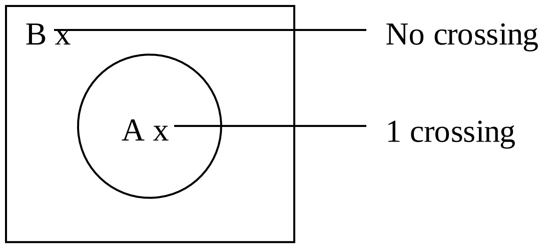


Point ID	Land use
1	Forest
2	Forest
3	Non-forest

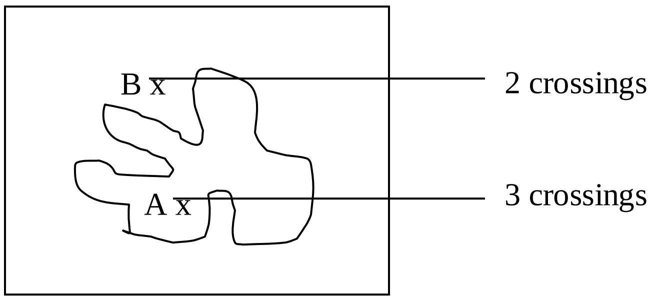
(Fig-2)

Point-in-polygon analysis

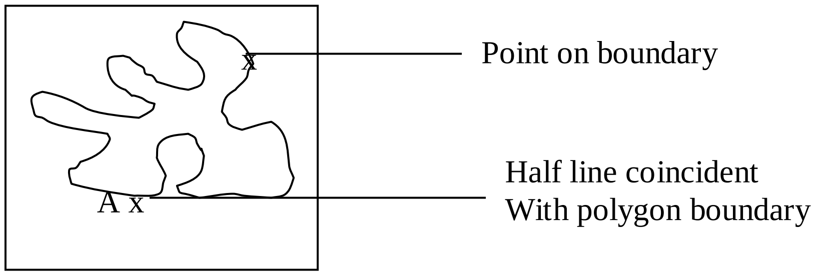
1) The most common technique used in GIS is the '**half line**' or **Jordon method** (Laurini and Thompson, 1992).Consider a circular polygon with a point at its centre, and another point outside the circular polygon. If a line is extended from the central point to the edge of the data layer it will cross the polygon boundary once. A line extended from the point outside the circle will not cross the polygon boundary.



2) Now consider a more complex polygon with crenellated edges. A line extended from the central point will always cross the polygon boundary an odd number of times, while the line extended from point outside the polygon will cross the boundary an even number of times.



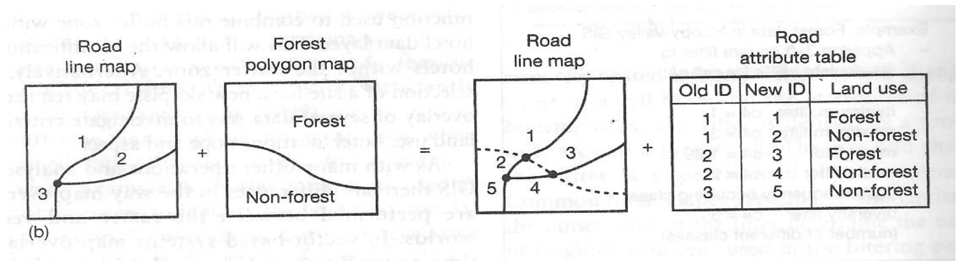
3) A difficulty with this method arises when points lie exactly on the polygon boundary. Are these considered in or out? Lauri and Thomson suggest that where the point has been produced using a mouse, it can be moved slightly to facilitate the in or out decision. Another problem situation is where the 'half line' is coinciding with the polygon boundary, making the counting of crossings impossible.



Line-in-polygon

In a line-in-polygon operation, the output contains the same line features as in the input but each of them is dissected by the polygon boundaries on the overlay map.

This overlay is more complicated. Imagine that we need to know where road pass through forest area to plan a scenic forest drive. To do this we need to overlay the road data on a data layer containing forest polygons. The output map will contain roads split into smaller segments representing 'roads in forest areas' and 'roads outside forest areas'.



Road
attribute table

Old ID	New ID	Land use
1	1	Forest
1	2	Non-forest
2	3	Forest
2	4	Non-forest
3	5	Non-forest

(Fig- 3)

Polygon-on-polygon

The most common overlay operation is polygon-on-polygon, involving two polygons maps. The output combines the polygon boundaries from the input and overlay maps to create a new set of polygons. This overlay could be used to examine the areas of forestry in the happy valley resort. Two input data layers are required – a forest data layer containing forest polygons, and resort boundary layer. Three different outputs could be obtained.

1) The output data layer could contain all the polygons from both of the input maps. This corresponds to the Boolean OR operation, or in mathematical terms, UNION.

2) The forest data layer could contain the whole of the resort area, and forest within this. The boundary of the resort would be used as the edge of the output map, and forest areas would be cut away if they fall outside it. This operation is referred to as 'cookie cutting' and it is equivalent to the mathematical IDENTITY operation.

3) The output data layer could contain the areas that meet both forest and within happy valley .This is the mathematical INTERSECT operation, and the out put map shows where the two input layers intersect. An output map would produced showing the whole of any forest polygons that are entirely with in the resort boundary , and 'cut 'away forest polygons which cross the resort boundary.

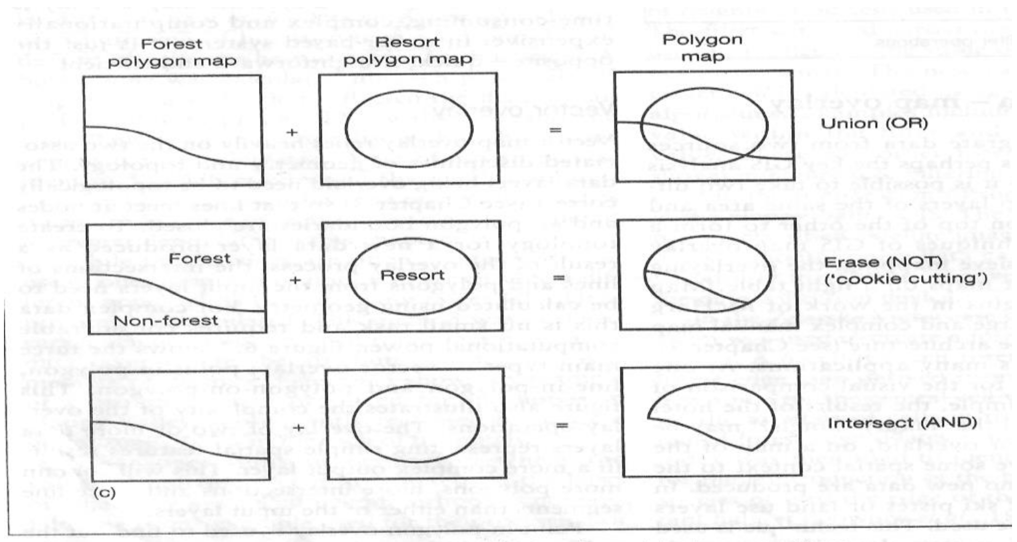
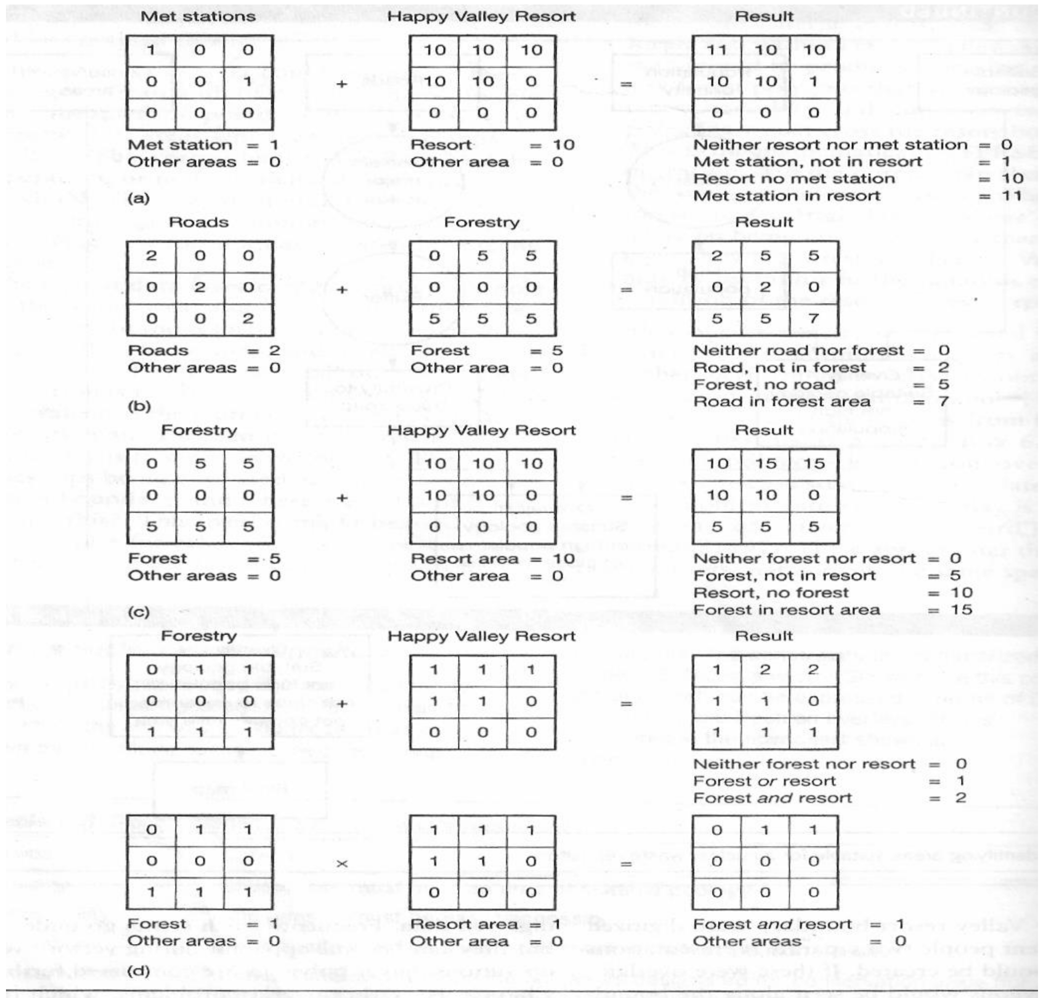


Figure 6.7 Vector overlays: (a) point-in-polygon; (b) line-in-polygon; (c) polygon-on-polygon

One problem with vector overlay is the possible generation of **sliver polygons.** These appear after the overlay of two data sets that contain the same spatial entities. If the happy valley resort boundary were digitized by two different people two separate representation of the area would be created. If these were overlaid, long thin polygons would be seen along the boundary instead of a single line. These sliver polygons arise from inconsistencies and inaccuracies in the digitized data.

Raster overlay

In the raster data structure everything is represented by cells- a point is represented by a single cell, a line by a string of cells and area by group of cells. The methods of performing overlays are different from those in vector GIS.Raster map overlay introduces the idea of map algebra or 'mapematics' (berry,1993) .A mathematical operations are performed on individual cell values from two or more input layers to produce an output value. Thus, the most important consideration in raster overlay is the appropriate coding of point, line and area features in the input data layers. Imagine that four of the happy valley data layers have been rasterized; the location of meteorological stations, the road network, the land use layer and the resort boundary.



ire 6.10 Raster overlays: (a) point-in-polygon (using add); (b) line-in-polygon (using add); (c) polygon-on-polygon
gon-on-polygon (Boolean alternatives)

To find out which meteorological stations are contained within the happy valley resort an equivalent to the vector **point-in-polygon overlay** is required. To do this one approach would be to add the two data layers.(figure 4a) The output map would contain cells with the following values:

- 0 for cells outside the resort boundary and without a meteorological station;
- 1 for cells containing meteorological stations but outside the resort boundary ;
- 10 for cells inside the resort boundary but without a meteorological station;
- 11 for cells inside resort boundary and containing a meteorological station.

In an operation to the vector **line-in-polygon method** (figure 4b), the sections of roads that pass through forest areas could be obtained. This would require the roads data layer, and a reclassified version of the land use map that contained only forest areas. Again the two maps would be added.

- 0 for cells with neither roads nor forest present ;
- 2 for cells with roads, but outside forest areas ;
- 5 for cells with forest present, but roads absent ;
- 7 for cells with both forest and roads present.

If the value ‘2’ for a road were added to land use codes, the new value for a cell could be the same as that for another land use type (for example a road value 2 + land use value 2(water) = 4(which is the same as the value here for agricultural land)

Polygon-on-polygon analysis is conducted in just the same way. (figure 4c). Again the coding of input layers is the key to understanding the output produced by raster overlay. For example, adding the forest layer and resort boundary would produce the output layer with the following codes:

- 0 for cells outside the resort boundary and with forest absent ;
- 5 for cells outside the resort boundary and with forest present ;
- 10 for cells inside the resort boundary and with forest absent ;
- 15 for cells inside the resort boundary and with forest present

The coding of raster images used in overlay is very important, and frequently users employ Boolean images (using only codes 1 and 0)(figure).The algebraic manipulation of images in raster GIS is a powerful and flexible way combining data and organizing analysis.

There are two issues affecting raster overlay

- Resolution
- Scales of measurement

Resolution is determined by the size of cell used. (Ex) SPOT satellite data, are collected at a resolution of 10m. For some analysis, you may wish to overlay a SPOT image with 40m resolution. The result will be an output grid with a resolution of 10m, which is greater than the resolution at which the second data set was collected. Since you cannot disaggregate data with any degree of certainty. A better approach to the overlay of these two data sets would be to aggregate cells in the SPOT image to match the resolution of the second layer

The second issue is that of **scales of measurement**. Care is needed when overlaying raster data layers to determine whether the operation makes real sense according to the scales of measurement

Subject Name: Fundamentals of GIS
Subject Code : 18MAG34E
Faculty : K Sumesh