Data Mining Primitives, Languages and System Architecture

Content

- Data mining primitives
- Languages
- System architecture
- Application – Geographical information system (GIS)

Introduction

- Motivation- need to extract useful information and knowledge from a large amount of data (data explosion problem)

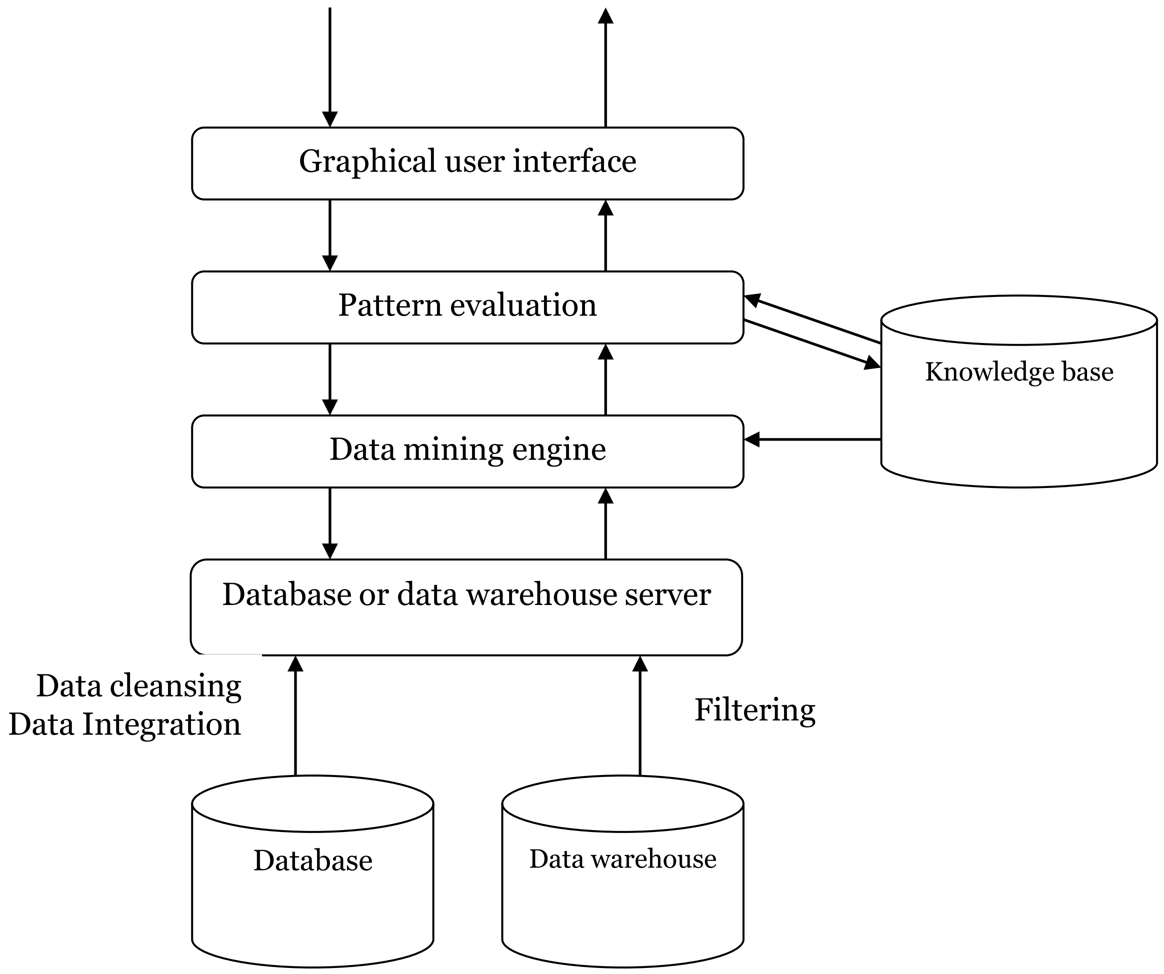
- Data Mining tools perform data analysis and may uncover important data patterns, contributing greatly to business strategies, knowledge bases, and scientific and medical research.

What is Data Mining???

- Data mining refers to *extracting or "mining" knowledge from large amounts of data*. Also referred as Knowledge Discovery in Databases.

- It is a process of discovering interesting knowledge from large amounts of data stored either in databases, data warehouses, or other information repositories.

Architecture of a typical data mining system



- Misconception: Data mining systems can autonomously dig out *all* of the valuable knowledge from a given large database, without human intervention.

- If there was no user intervention then the system would uncover a large set of patterns that may even surpass the size of the database. Hence, user interference is required.

- This user communication with the system is provided by using a set of *data mining primitives*.

Data Mining Primitives

Data mining primitives define a data mining task, which can be specified in the form of a data mining query.

- Task Relevant Data
- Kinds of knowledge to be mined
- Background knowledge
- Interestingness measure
- Presentation and visualization of discovered patterns

Task relevant data

- Data portion to be investigated.
- Attributes of interest (relevant attributes) can be specified.
- Initial data relation
- Minable view

Example

- If a data mining task is to study associations between items frequently purchased at *AllElectronics* by customers in Canada, the task relevant data can be specified by providing the following information:
  - Name of the *database or data warehouse* to be used (e.g., *AllElectronics_db*)
  - Names of the *tables or data cubes* containing relevant data (e.g., *item, customer, purchases* and *items_sold*)
  - *Conditions* for selecting the relevant data (e.g., retrieve data pertaining to purchases made in Canada for the current year)
  - The *relevant attributes or dimensions* (e.g., *name* and *price* from the *item* table and income and age from the customer table)

Kind of knowledge to be mined

- It is important to specify the knowledge to be mined, as this determines the data mining function to be performed.
- Kinds of knowledge include concept description, association, classification, prediction and clustering.
- User can also provide pattern templates. Also called metapatterns or metarules or metaqueries.

Example

A user studying the buying habits of *allelectronics* customers may choose to mine *association rules* of the form:

P (X:customer,W) ^ Q (X,Y) => buys (X,Z)

Meta rules such as the following can be specified:

age (X, "30.....39") ^ income (X, "40k....49K") => buys (X, "VCR")

[2.2%, 60%]

occupation (X, "student ") ^ age (X, "20.....29")=> buys (X, "computer")

[1.4%, 70%]

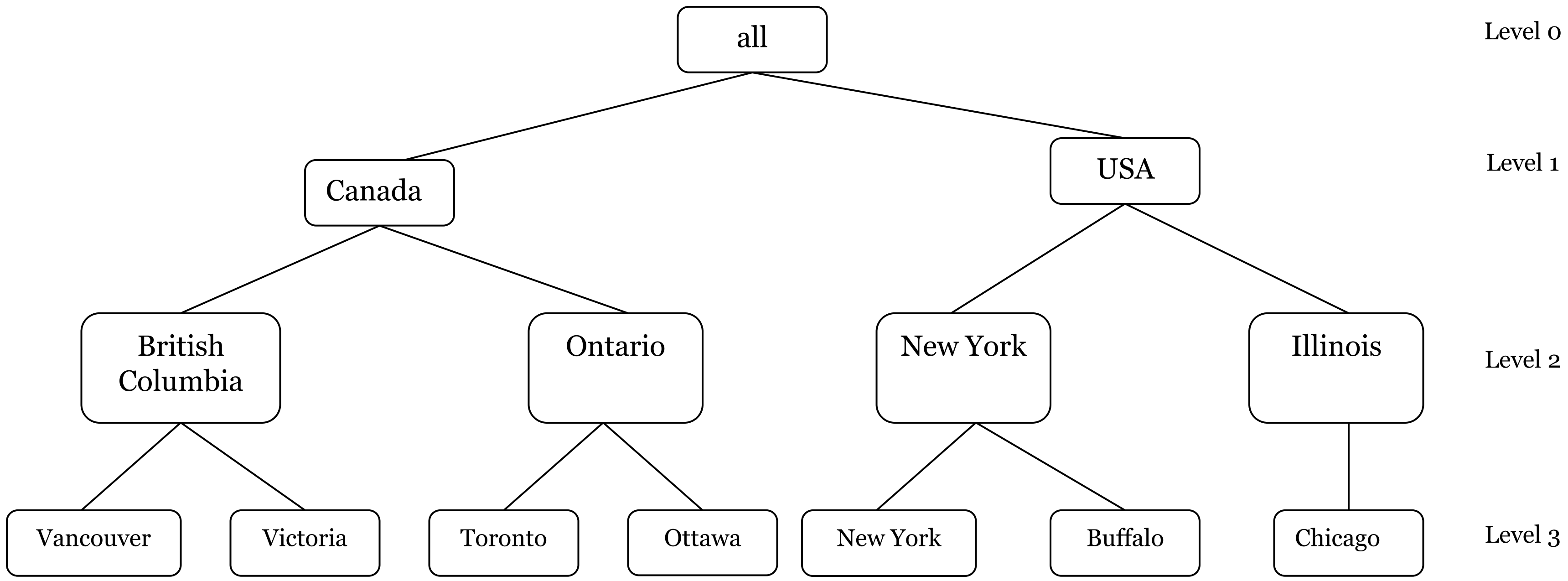
Background knowledge

- It is the information about the domain to be mined
- Concept hierarchy: is a powerful form of background knowledge.
- Four major types of concept hierarchies:
  schema hierarchies
  set-grouping hierarchies
  operation-derived hierarchies
  rule-based hierarchies

Concept hierarchies (1)

- Defines a sequence of mappings from a set of low-level concepts to higher-level (more general) concepts.
- Allows data to be mined at multiple levels of abstraction.
- These allow users to view data from different perspectives, allowing further insight into the relationships.
- Example (location)

Example


all
Canada
USA
British Columbia
Ontario
New York
Illinois
Vancouver
Victoria
Toronto
Ottawa
New York
Buffalo
Chicago
Level 0
Level 1
Level 2
Level 3

Concept hierarchies (2)

- Rolling Up - Generalization of data
  Allows to view data at more meaningful and explicit abstractions.
  Makes it easier to understand
  Compresses the data
  Would require fewer input/output operations
- Drilling Down - Specialization of data
  Concept values replaced by lower level concepts
- There may be more than concept hierarchy for a given attribute or dimension based on different user viewpoints
- Example:
  Regional sales manager may prefer the previous concept hierarchy but marketing manager might prefer to see location with respect to linguistic lines in order to facilitate the distribution of commercial ads.

Schema hierarchies

- Schema hierarchy is the total or partial order among attributes in the database schema.
- May formally express existing semantic relationships between attributes.
- Provides metadata information.
- Example: location hierarchy

  street < *city* < *province/state* < *country*

Set-grouping hierarchies

- Organizes values for a given attribute into groups or sets or range of values.
- Total or partial order can be defined among groups.
- Used to refine or enrich schema-defined hierarchies.
- Typically used for small sets of object relationships.
- Example: Set-grouping hierarchy for age

  {young, middle_aged, senior} $\subset$ *all (age)*

  {20....29} $\subset$ *young*

  {40....59} $\subset$ *middle_aged*

  {60....89} $\subset$ *senior*

Operation-derived hierarchies

- Operation-derived:

  based on operations specified

  operations may include

    - decoding of information-encoded strings
    - information extraction from complex data objects
    - data clustering

  Example: URL or email address

  xyz@cs.iitm.in gives login name < dept. < univ. < country

Rule-based hierarchies

- Rule-based:

  Occurs when either whole or portion of a concept hierarchy is defined as a set of rules and is evaluated dynamically based on current database data and rule definition

- Example: Following rules are used to categorize items as *low_profit, medium_profit* and *high_profit_margin*.

  low_profit_margin(X) <= price(X,P1)^cost(X,P2)^((P1-P2)<50)

  medium_profit_margin(X) <= price(X,P1)^cost(X,P2)^((P1-P2)≥50)^((P1-P2)≤250)

  high_profit_margin(X) <= price(X,P1)^cost(X,P2)^((P1-P2)>250)

Interestingness measure (1)

- Used to confine the number of uninteresting patterns returned by the process.
- Based on the structure of patterns and statistics underlying them.
- Associate a threshold which can be controlled by the user.
- patterns not meeting the threshold are not presented to the user.
- Objective measures of pattern interestingness:
  simplicity
  certainty (confidence)
  utility (support)
  novelty

Interestingness measure (2)

- Simplicity

  a patterns interestingness is based on its overall simplicity for human comprehension.

  Example: Rule length is a simplicity measure

- Certainty (confidence)

  Assesses the validity or trustworthiness of a pattern.

  confidence is a certainty measure

  $$confidence\ (A \Rightarrow B) = \frac{\#\ tuples\ containing\ both\ A\ and\ B}{\#\ tuples\ containing\ A}$$

  A confidence of 85% for the rule buys(X, "computer")=>buys(X,"software") means that 85% of all customers who purchased a computer also bought software

Interestingness measure (3)

- Utility (support)

  usefulness of a pattern

  $$support\ (A{=}{>}B) = \frac{\#\ tuples\ containing\ both\ A\ and\ B}{total\ \#\ \ of\ tuples}$$

  A support of 30% for the previous rule means that 30% of all customers in the computer department purchased both a computer and software.

- Association rules that satisfy both the minimum confidence and support threshold are referred to as **strong association rules.**

- Novelty

  Patterns contributing new information to the given pattern set are called novel patterns (example: Data exception).

  removing redundant patterns is a strategy for detecting novelty.

Presentation and visualization

- For data mining to be effective, data mining systems should be able to display the discovered patterns in multiple forms, such as rules, tables, crosstabs (cross-tabulations), pie or bar charts, decision trees, cubes, or other visual representations.

- User must be able to specify the forms of presentation to be used for displaying the discovered patterns.

Data mining query languages

- Data mining language must be designed to facilitate flexible and effective knowledge discovery.
- Having a query language for data mining may help standardize the development of platforms for data mining systems.
- But designed a language is challenging because data mining covers a wide spectrum of tasks and each task has different requirement.
- Hence, the design of a language requires deep understanding of the limitations and underlying mechanism of the various kinds of tasks.

Data mining query languages (2)

- So…how would you design an efficient query language???
- Based on the primitives discussed earlier.
- DMQL allows mining of different kinds of knowledge from relational databases and data warehouses at multiple levels of abstraction.

DMQL

- Adopts SQL-like syntax
- Hence, can be easily integrated with relational query languages
- Defined in BNF grammar
  [ ] represents 0 or one occurrence
  { } represents 0 or more occurrences
  Words in **sans serif** represent keywords

DMQL-Syntax for task-relevant data specification

- Names of the relevant database or data warehouse, conditions and relevant attributes or dimensions must be specified
- **use database** ‹database_name› or **use data warehouse** ‹data_warehouse_name›
- **from** ‹relation(s)/cube(s)› [**where** condition]
- **in relevance to** ‹attribute_or_dimension_list›
- **order by** ‹order_list›
- **group by** ‹grouping_list›
- **having** ‹condition›

Example

Example 4.11 This example shows how to use DMQL to specify the task-relevant data described in Example 4.1 for the mining of associations between items frequently purchased at *AllElectronics* by Canadian customers, with respect to customer *income* and *age*. In addition, the user specifies that she would like the data to be grouped by date. The data are retrieved from a relational database.

```
use database AllElectronics_db
in relevance to I.name, I.price, C.income, C.age
from customer C, item I, purchases P, items_sold S
where I.item_ID = S.item_ID and S.trans_ID = P.trans_ID and P.cust_ID = C.cust_ID
          and C.address = "Canada"
group by P.date
```

□

Syntax for Kind of Knowledge to be Mined

- Characterization :

  ‹Mine_Knowledge_Specification› **::=**

  mine characteristics [**as** ‹pattern_name›]

  analyze ‹measure(s)›
- Example:

  mine characteristics as customerPurchasing **analyze** count%

- Discrimination:

  ‹Mine_Knowledge_Specification› **::=**

  mine comparison [**as** ‹ pattern_name›]

  for ‹target_class› **where** ‹target_condition›

  {**versus** ‹contrast_class_i **where** ‹contrast_condition_i›}

  analyze ‹measure(s)›
- Example:

  Mine comparison as purchaseGroups

  for bigspenders **where** avg(I.price) >= $100

  versus budgetspenders **where** avg(I.price) < $100

  analyze count

Syntax for Kind of Knowledge to be Mined (2)

- Association:

  ‹Mine_Knowledge_Specification› **::=**

  mine associations [**as** ‹pattern_name›]

  [**matching** ‹metapattern›]

- Example: **mine associations as** buyingHabits

  matching P(X: customer, W) ^ Q(X,Y) => buys (X,Z)

- Classification:

  ‹Mine_Knowledge_Specification› ::=

  mine classification [**as** ‹pattern_name›]

  analyze ‹classifying_attribute_or_dimension›

- Example: **mine classification as** classifyCustomerCreditRating

  analyze credit_rating

Syntax for concept hierarchy specification

- More than one concept per attribute can be specified
- **Use hierarchy** ‹hierarchy_name› **for** ‹attribute_or_dimension›
- Examples:

  Schema concept hierarchy (ordering is important)

  - **define hierarchy** location_hierarchy **on** address **as** [street,city,province_or_state,country]

  Set-Grouping concept hierarchy

  - **define hierarchy** age_hierarchy **for** age **on** customer **as**

    level1: {young, middle_aged, senior} < level0: **all**

    level2: {20, ..., 39} < level1: young

    level2: {40, ..., 59} < level1: middle_aged

    level2: {60, ..., 89} < level1: senior

Syntax for concept hierarchy specification (2)

- operation-derived concept hierarchy
  - **define hierarchy** age_hierarchy **for** age **on** customer **as**
    {age_category(1), ..., age_category(5)} := cluster (default, age, 5) < **all**(age)

- rule-based concept hierarchy
  - **define hierarchy** profit_margin_hierarchy **on** item **as**
    level_1: low_profit_margin < level_0: **all**
    if (price - cost)< $50
    level_1: medium-profit_margin < level_0: **all**
    if ((price - cost) > $50) and ((price - cost) <= $250))
    level_1: high_profit_margin < level_0: **all**
    if (price - cost) > $250

Syntax for interestingness measure specification

- **with** [‹interest_measure_name›] **threshold** = ‹threshold_value›

- Example:
  with support **threshold** = 5%
  with confidence **threshold** = 70%

Syntax for pattern presentation and visualization specification

- **display as** ‹result_form›

- The result form can be rules, tables, cubes, crosstabs, pie or bar charts, decision trees, curves or surfaces.

- To facilitate interactive viewing at different concept levels or different angles, the following syntax is defined:

```
‹Multilevel_Manipulation›  ::=  roll up on ‹attribute_or_dimension›
                               | drill down on ‹attribute_or_dimension›
                               | add ‹attribute_or_dimension›
                               | drop ‹attribute_or_dimension›
```

Architectures of Data Mining System

- With popular and diverse application of data mining, it is expected that a good variety of data mining system will be designed and developed.
- Comprehensive information processing and data analysis will be continuously and systematically surrounded by data warehouse and databases.
- A critical question in design is whether we should integrate data mining systems with database systems.
- This gives rise to four architecture:

  - No coupling
  - Loose Coupling
  - Semi-tight Coupling
  - Tight Coupling

Cont.

- No Coupling: DM system will not utilize any functionality of a DB or DW system

- Loose Coupling: DM system will use some facilities of DB and DW system like storing the data in either of DB or DW systems and using these systems for data retrieval

- Semi-tight Coupling: Besides linking a DM system to a DB/DW systems, efficient implementation of a few DM primitives.

- Tight Coupling: DM system is smoothly integrated with DB/DW systems. Each of these DM, DB/DW is treated as main functional component of information retrieval system.

Application

GIS (Geographical Information System)

What is GIS???

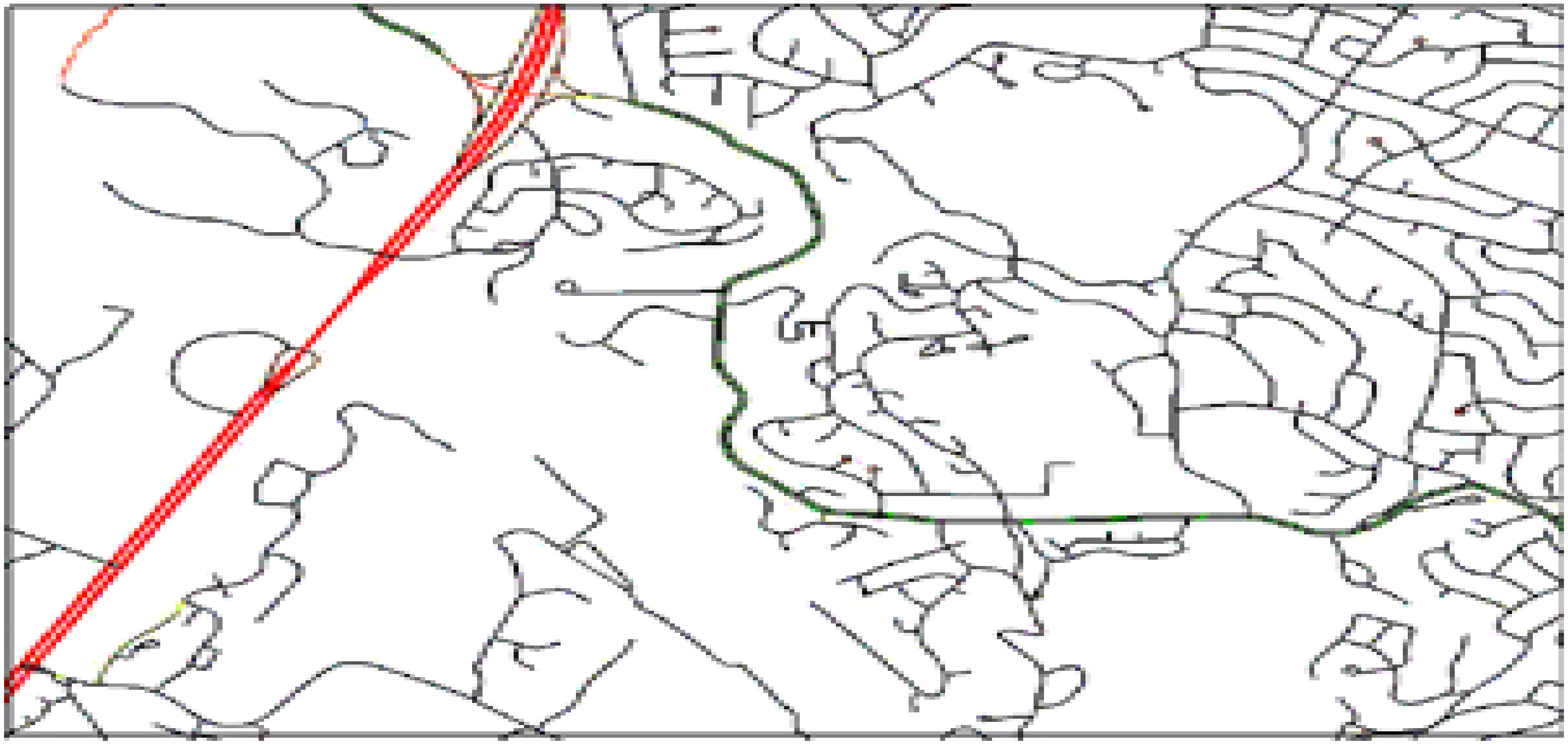
A GIS is a computer system capable of capturing, storing, analyzing, and displaying geographically referenced information;

Example: GIS might be used to find wetlands that need protection from pollution.

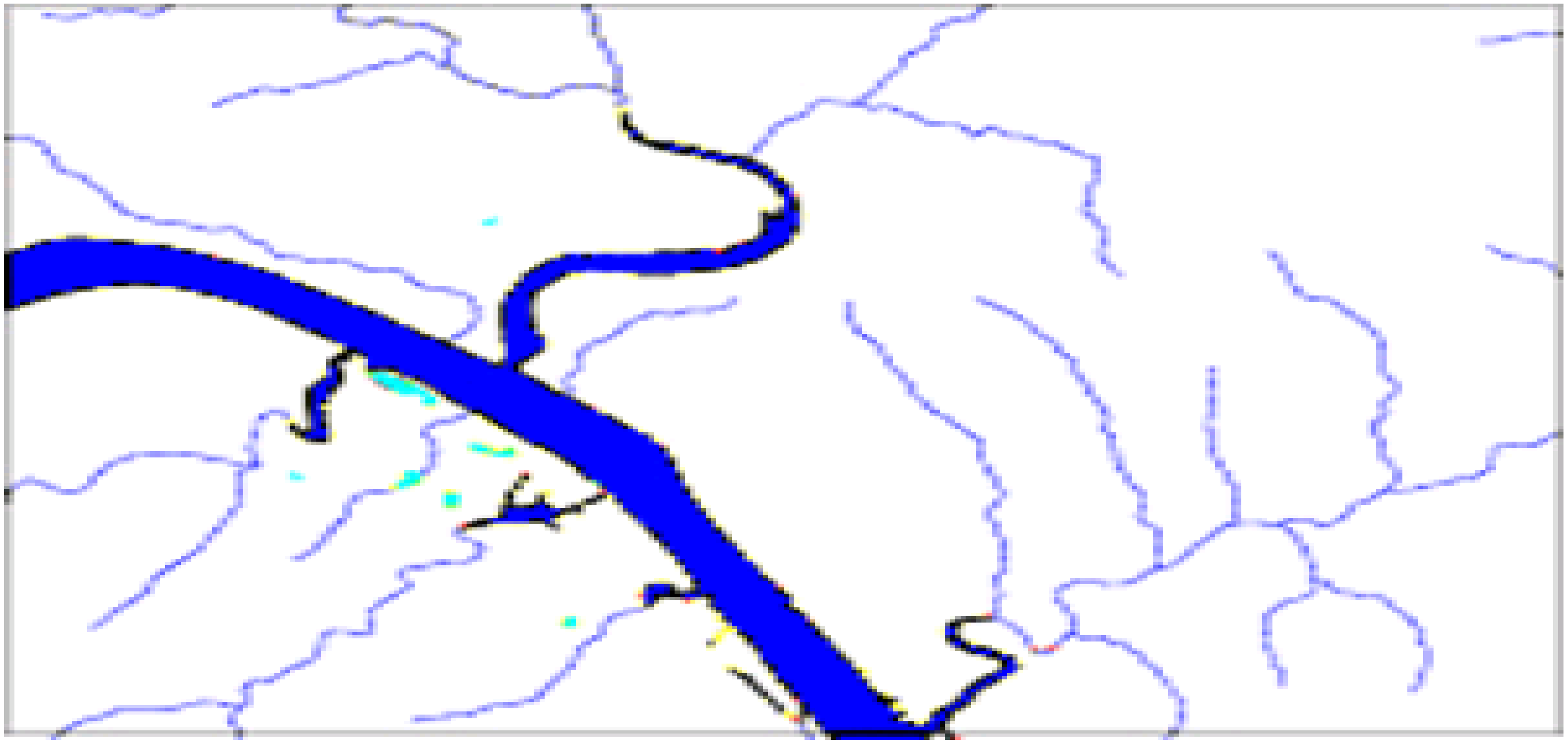
How does a GIS work?

- GIS works by Relating information from different sources
- The power of a GIS comes from the ability to relate different information in a spatial context and to reach a conclusion about this relationship.
- Most of the information we have about our world contains a location reference, placing that information at some point on the globe.

Geological Survey (USGS) Digital Line Graph (DLG) of roads.



Digital Line Graph of rivers.



Data capture

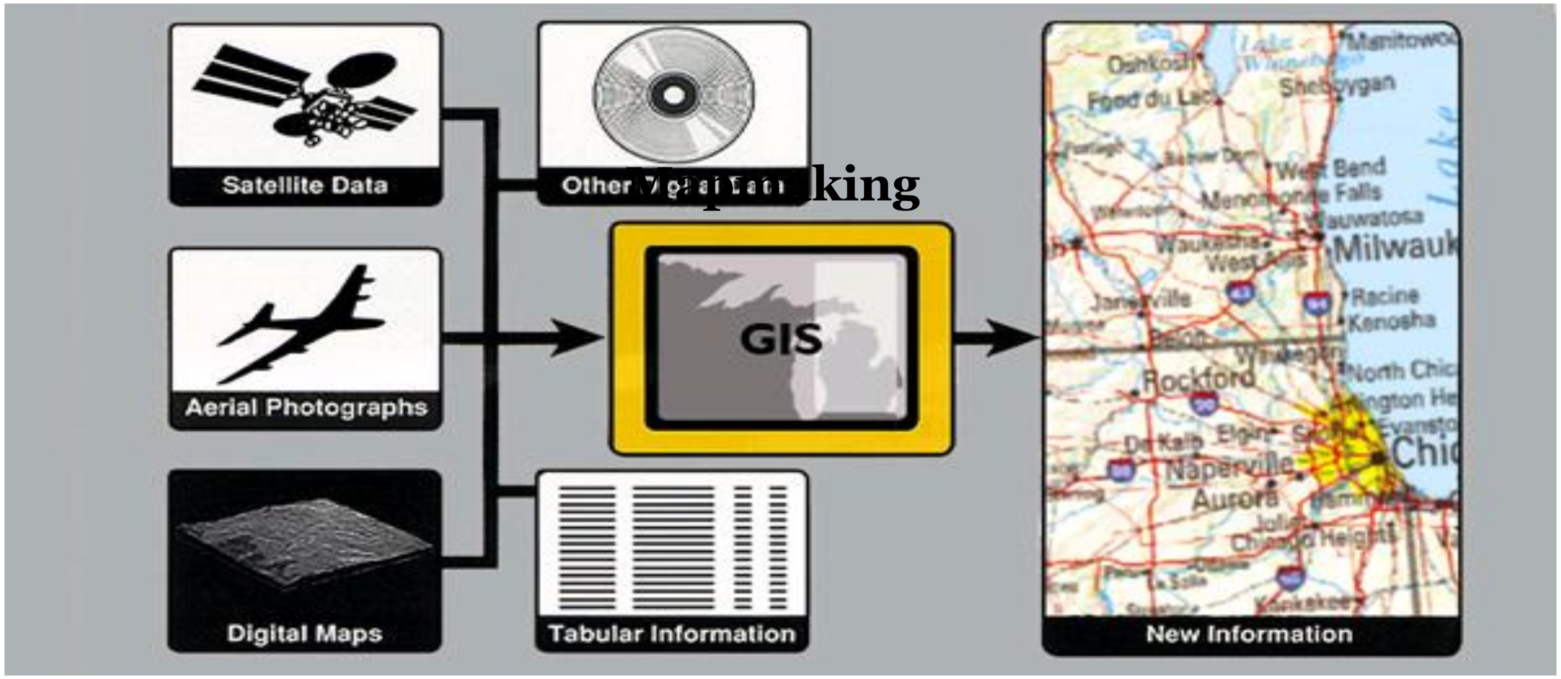
- If the data to be used are not already in digital form
  - Maps can be digitized by hand-tracing with a computer mouse
  - Electronic scanners can also be used

- Co-ordinates for the maps can be collected using Global Positioning System (GPS) receivers

- Putting the information into the system—involves identifying the objects on the map, their absolute location on the Earth's surface, and their spatial relationships .

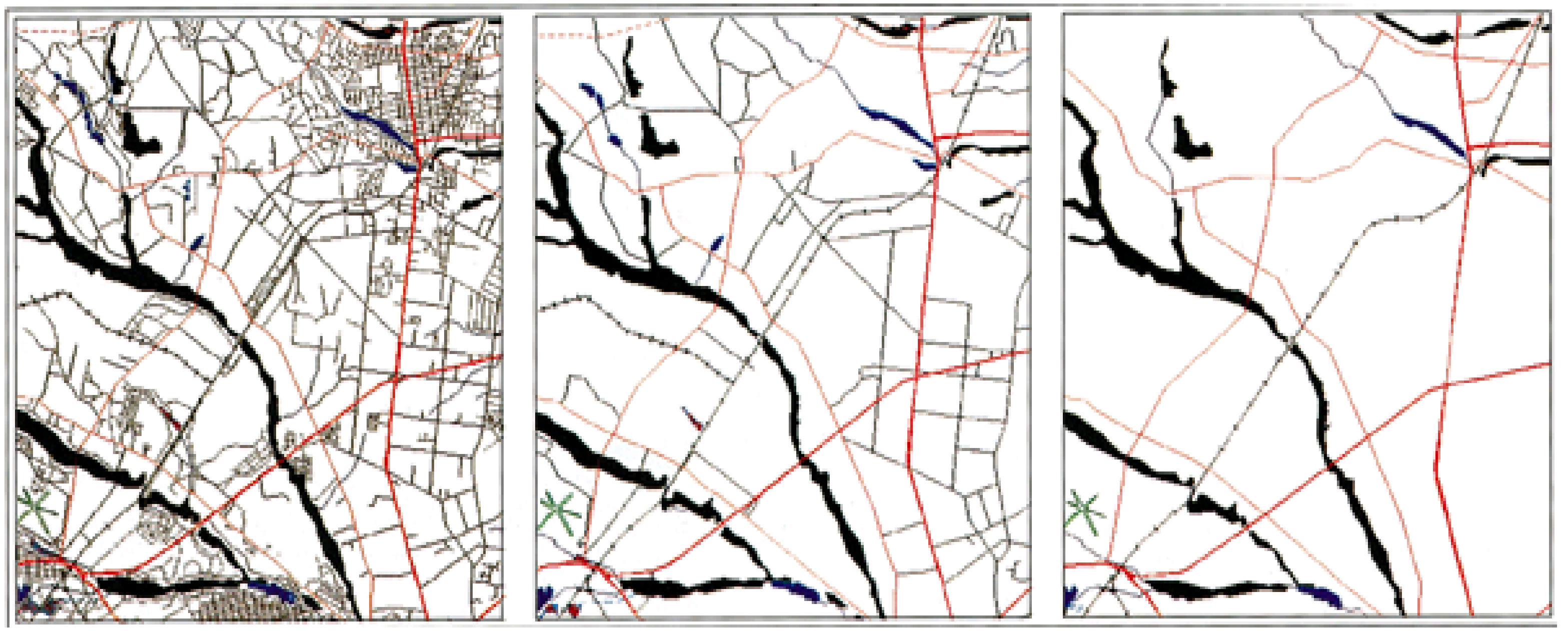
Data integration

- A GIS makes it possible to link, or integrate, information that is difficult to associate through any other means.



Mapmaking

- Researchers are working to incorporate the mapmaking processes of traditional cartographers into GIS technology for the automated production of maps.

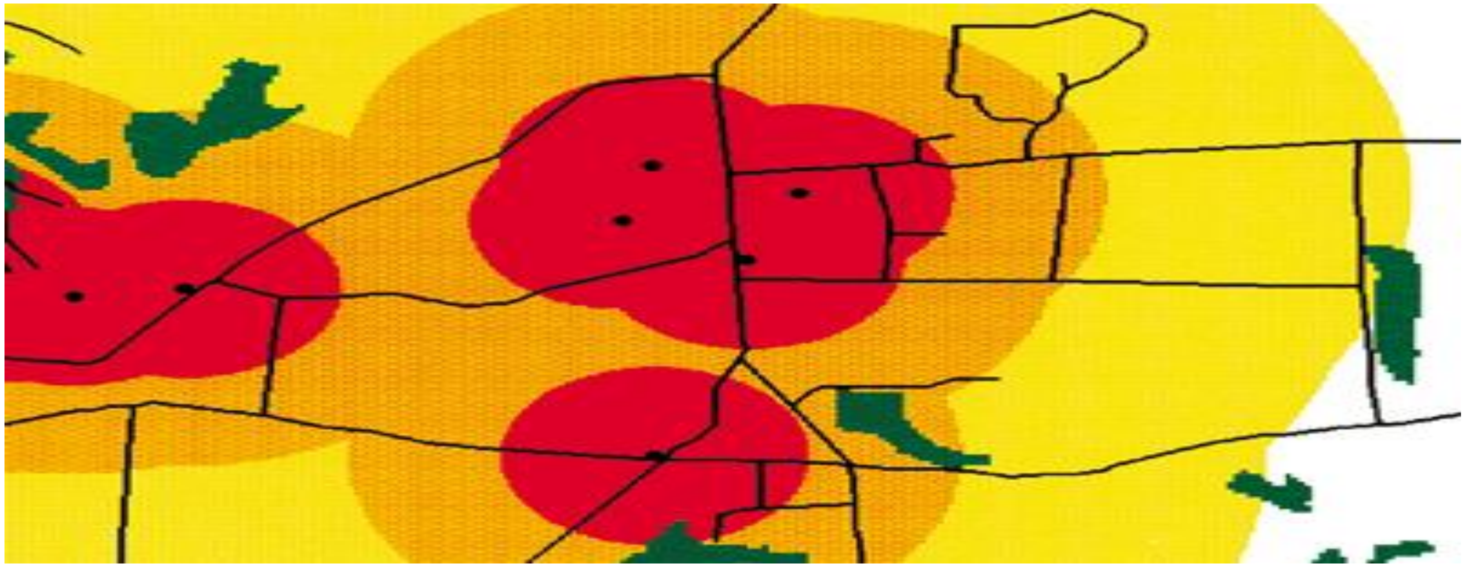


What is special about GIS??

- Information retrieval: What do you know about the swampy area at the end of your street? With a GIS you can "point" at a location, object, or area on the screen and retrieve recorded information about it from off-screen files . Using scanned aerial photographs as a visual guide, you can ask a GIS about the geology or hydrology of the area or even about how close a swamp is to the end of a street. This type of analysis allows you to draw conclusions about the swamp's environmental sensitivity.

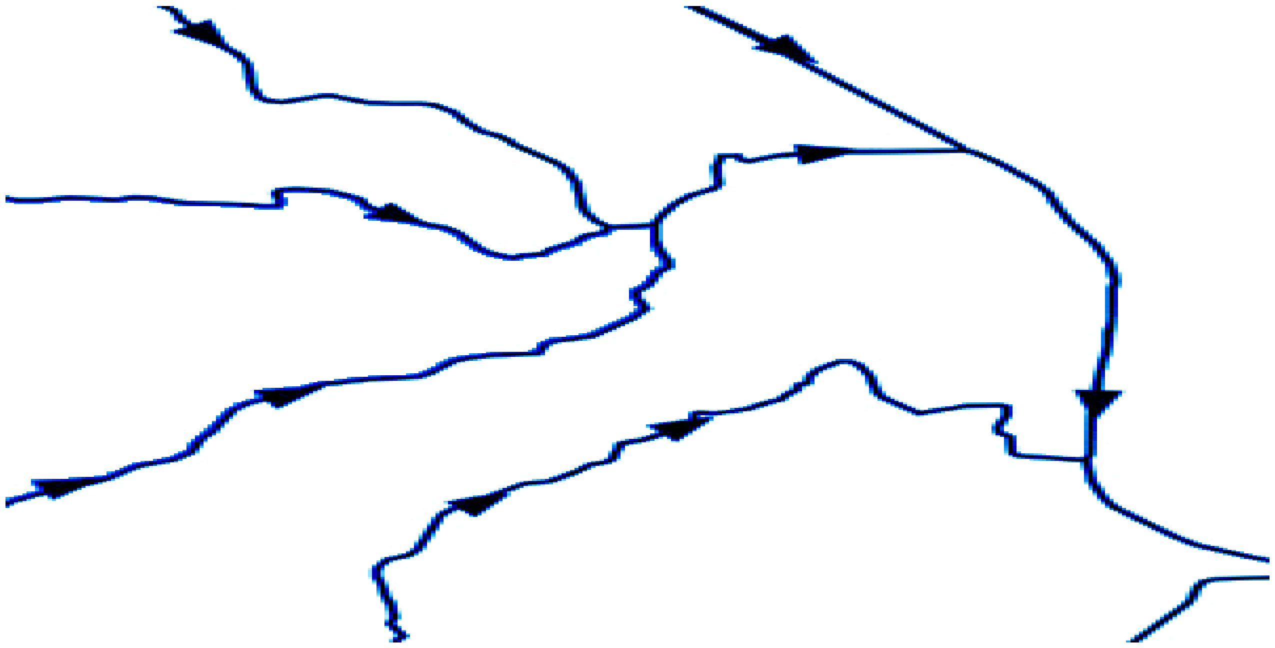
Cont.

- Topological modeling: Have there ever been gas stations or factories that operated next to the swamp? Were any of these uphill from and within 2 miles of the swamp? A GIS can recognize and analyze the spatial relationships among mapped phenomena. Conditions of adjacency (what is next to what), containment (what is enclosed by what), and proximity (how close something is to something else) can be determined with a GIS



Cont.

- Networks: When nutrients from farmland are running off into streams, it is important to know in which direction the streams flow and which streams empty into other streams. This is done by using a linear network. It allows the computer to determine how the nutrients are transported downstream. Additional information on water volume and speed throughout the spatial network can help the GIS determine how long it will take the nutrients to travel downstream



Data Output

- A critical component of a GIS is its ability to produce graphics on the screen or on paper to convey the results of analyses to the people who make decisions about resources.

The future of GIS

- GIS and related technology will help analyze large datasets, allowing a better understanding of terrestrial processes and human activities to improve economic vitality and environmental quality

How is it related to DM?

- In order to represent the data in graphical Format which is most likely represented as a graph cluster analysis is done on the data set.

- Clustering is a data mining concept which is a process of grouping together the data into clusters or classes.