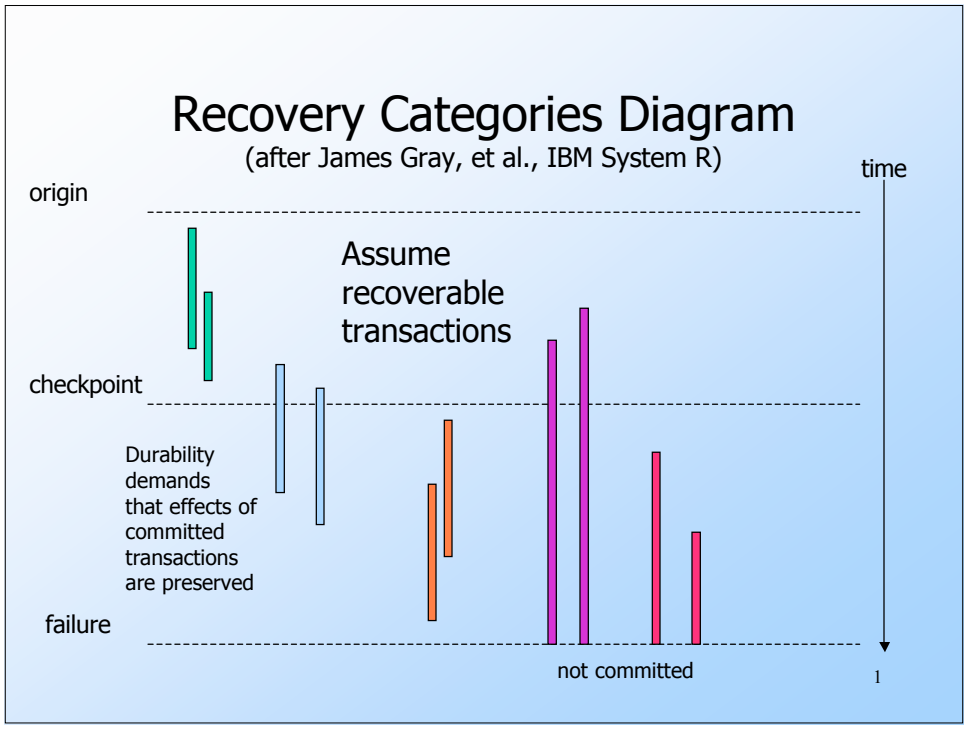
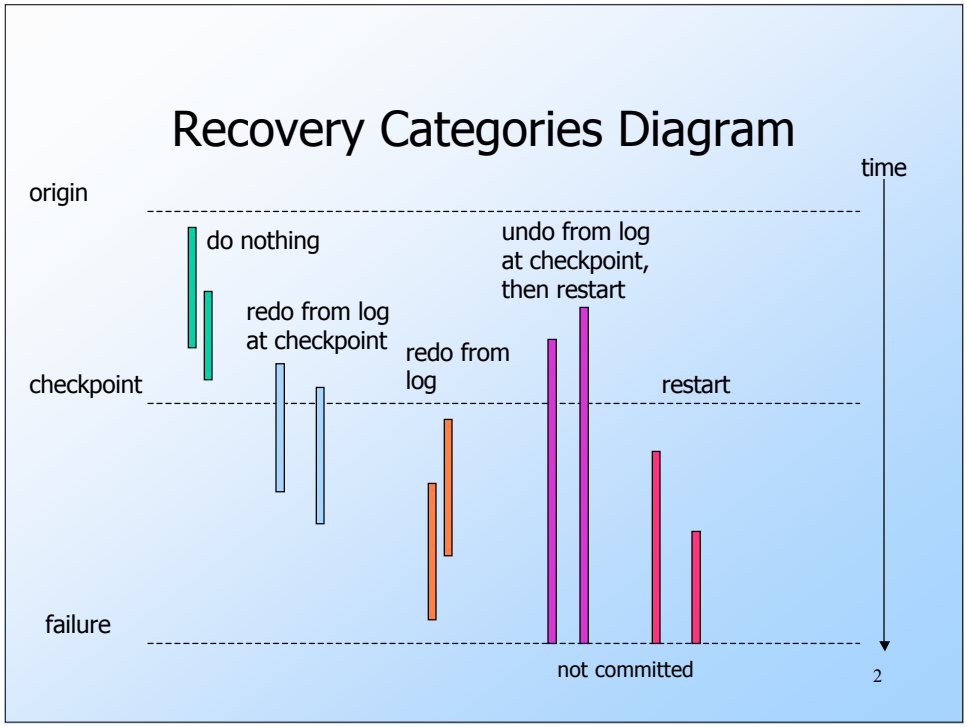
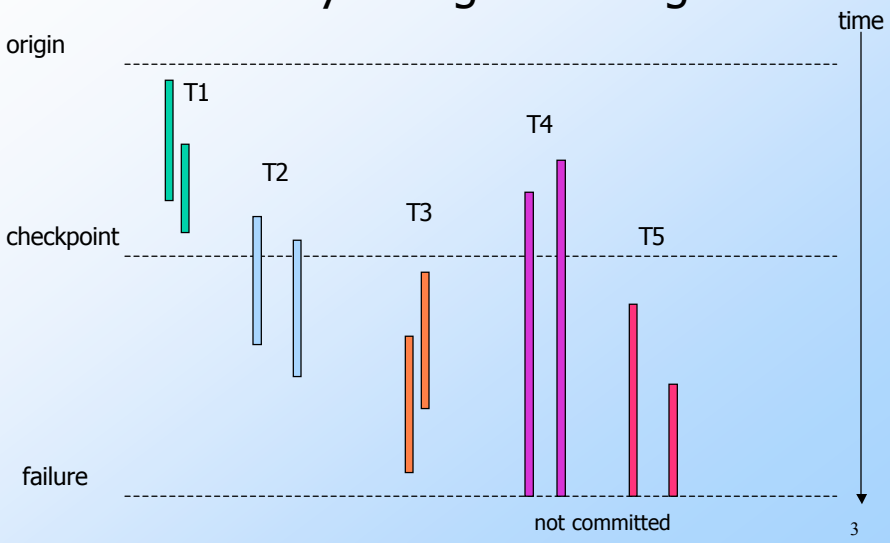

Recovery Categories Diagram
(after James Gray, et al., IBM System R)
time
origin
Assume
recoverable
transactions
checkpoint
Durability
demands
that effects of
committed
transactions
are preserved
failure
not committed
1


Recovery Categories Diagram
time
origin
do nothing
undo from log
at checkpoint,
then restart
redo from log
at checkpoint
redo from
log
checkpoint
restart
failure
not committed
2

Recovery Categories Diagram



3

From Gray, et al.

- "Restart uses the log as follows:
- It reads the most recent checkpoint record and assumes that all transactions active at the time of the checkpoint are of type T4.
- It then **reads the log in the forward direction**, starting from the checkpoint record.
  - If it encounters a BEGIN record, it notes that this is a transaction of type T5.
  - If it encounters the COMMIT record of a T4 transaction, it reclassifies the transaction to T2.
  - Similarly, T5 transactions are reclassified as T3 if a COMMIT record is encountered.
  - When it reaches the end of the log, all T2, T3, T4, and T5 transactions are known.
  - T4 and T5 are the "losers" and T2 and T3 are the "winners".

4

Gray, et al., continued

- Restart **reads the log backward** from the checkpoint, **undoing** all actions of losers.
- It then **reads the log forward** from the checkpoint, **redoing** all actions of winners.
- Once this is done, a new checkpoint is written so that the restart work will not be lost."

5

Note

- The System R approach is but one of several possibilities.
- For an anthology of papers on recovery, see:

  Kumar and Hsu, Recovery mechanisms in database systems, Prentice-Hall, 1998.

6

Spatial, Temporal, and Spatio-Temporal Databases

Robert M. Keller
Harvey Mudd College
April 2004

7

Motivation

- Endowing DBMS with more application capabilities when space or time are involved.
- Can suggest a logical view of data in such applications.
- Can provide flexible querying similar to what query languages do for other data.

8

Contrasts

- Databases (so far) have focused on discrete relations (e.g. the domain values and tuples are enumerable).
- Spatial and temporal databases attempt to integrate properties of space and time with discrete data.

9

Contrasts

- Traditional Query:
  Who is the representative for Claremont, California?
- Spatial Query:
  Who are the representatives living west of Reno, Nevada?
- Temporal Query:
  Which representatives served during the tenure of President Clinton?

10

Example of a Spatial DB Application

- Vehicular Navigation Sytem
  - Receives current position via GPS
  - Can plan a route from current position to destination, identified in a variety of different ways:
    - Enter address
    - Enter place name
    - Point to on map
    - Query database of place types, then select
  - Can display and enunciate (by voice response) directions for each intersection en route
  - Can computing remaining distance and estimated driving time

11

Other Spatial Applications

- Computers: Distributed network of mobile agents
- Medicine: Human body mapped to 3-D space, for medical diagnoses
- Astronomy: Relate spatial distances to astrophysical features
- Urban planning: Relate metrics of usage areas to demographic data (which is temporal)
- Military: Battelfield management
- Politics: (detection of) Gerrymandering

12

Spatial Types

- Point
- Line
- Polyline (series of points connected by lines)
- Curve (e.g. determined by equation)
- Spline (series of curves with smooth transitions
- 2-D shapes, e.g. rectangle, ellipse
- 3-D shapes, e.g. prism, polyhedron, ellipsoid
- Composite shapes
- Shapes from operators (union, intersection, subtraction)

13

Spatial Relationships

- Touching, parallel
- Within (e.g. point, line, or shape within another shape)
- Relative location between items
- Distance between items
- Shortest path including certain items

14

Ties with Other Areas

- Operations Research (Routing algorithms, Flow algorithms)
- Computational Geometry (Algorithms)
- AI: Constraint networks

15

Early Demo Example: Chat
(Warren and Pereira)

- Implemented in Prolog
- Illustrates natural language processing as well as geographic information:
  - prolog (Quintus)
  - :- ensure_loaded(demo(chat)).
  - :- demo(_). or :- hi.

16

Chat Demo

| ?- hi.
Question: Which countries border mexico?
belize, guatemala and united_states.

Question: which countries are bordered by two seas ?
egypt, iran, israel, saudi_arabia and turkey.

Question: what is the total area of countries south of the equator and not in australasia ?
10228 ksqmiles.

Question: what are the countries from which a river flows into the black_sea ?
romania and soviet_union.

Question: which country bordering the mediterranean borders a country that is bordered by a country whose population exceeds the population of india ?
turkey.

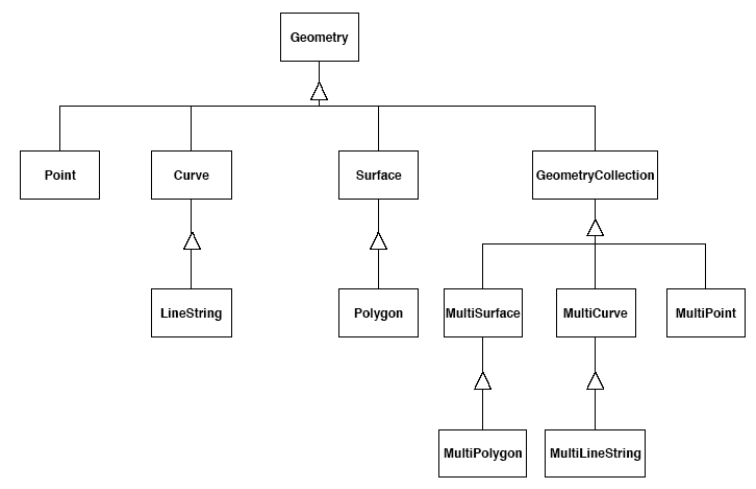
17

Open Geodata Interoperability Specification (OGIS) Standard

- Consists of base-class Geometry and four sub-classes:
  - Point, Curve, Surface and GeometryCollection
- Operations fall into three categories:
  - Apply to all geometry types
    - SpatialReference, Envelope, Export,IsSimple, Boundary
  - Predicates for Topological relationships
    - Equal, Disjoint, Intersect, Touch, Cross, Within, Contains
  - Spatial Data Analysis
    - Distance,Buffer,Union, Intersection, ConvexHull, SymDiff

18

SQL/OGIS Data Type Hierarchy



19

SQL/OGIS Data Examples

Geometry Type	SQL Text Literal Representation	Comment
Point	`'POINT (10 10)'`	a `Point`
LineString	`'LINESTRING ( 10 10, 20 20, 30 40)'`	a `LineString` with 3 points
Polygon	`'POLYGON ((10 10, 10 20, 20 20, 20 15, 10 10))'`	a `Polygon` with 1 exterior ring and 0 interior rings
Multipoint	`'MULTIPOINT (10 10, 20 20)'`	a `MultiPoint` with 2 point
MultiLineString	`'MULTILINESTRING ((10 10, 20 20), (15 15, 30 15))'`	a `MultiLineString` with 2 linestrings
MultiPolygon	`'MULTIPOLYGON ( ((10 10, 10 20, 20 20, 20 15, 10 10)), ((60 60, 70 70, 80 60, 60 60 ) ))'`	a `MultiPolygon` with 2 polygons
GeomCollection	`'GEOMETRYCOLLECTION (POINT (10 10), POINT (30 30), LINESTRING (15 15, 20 20))'`	a `GeometryCollection` consisting of 2 `Point` values and a `LineString` value

SQL/OGIS Functions (Examples)

`Disjoint(g1 Geometry, g2 Geometry) : Integer`	The return type is `Integer`, with a return value of 1 for `TRUE`, 0 for `FALSE`, and –1 for `UNKNOWN` corresponding to a function invocation on `NULL` arguments. `TRUE` if the intersection of `g1` and `g2` is the empty set.
`Touches(g1 Geometry, g2 Geometry) : Integer`	The return type is `Integer`, with a return value of 1 for `TRUE`, 0 for `FALSE`, and –1 for `UNKNOWN` corresponding to a function invocation on `NULL` arguments. `TRUE` if the only points in common between `g1` and `g2` lie in the union of the boundaries of `g1` and `g2`.
`Within(g1 Geometry, g2 Geometry) : Integer`	The return type is `Integer`, with a return value of 1 for `TRUE`, 0 for `FALSE`, and –1 for `UNKNOWN` corresponding to a function invocation on `NULL` arguments. `TRUE` if `g1` is completely contained in `g2`.
`Overlaps(g1 Geometry, g2 Geometry) : Integer`	The return type is `Integer`, with a return value of 1 for `TRUE`, 0 for `FALSE`, and –1 for `UNKNOWN` corresponding to a function invocation on `NULL` arguments. `TRUE` if the intersection of `g1` and `g2` results in a value of the same dimension as `g1` and `g2` that is different from both `g1` and `g2`.

21

SQL/OGIS Query

(from "Spatial Databases -- A Tour")

Query: List the GDP and the distance of a country's capital city to the equator for all countries.

SELECT Co.GDP, **`Distance`**(Point(0,Ci.Shape.y),Ci.Shape) AS "Distance"
FROM Country Co,City Ci
WHERE Co.Name = Ci.Country
AND Ci.Capital ='Y '

Note: The spatial operator **`Touch`**() is used in the WHERE clause to join the Country table with itself. This query is an example of **spatial self join** operation.

Co. Name	Co. GDP	Dist-to-Eq (in Km).
Havana	16.9	2562
Washington, D.C.	8003	4324
Brasilia	1004	1756
Ottawa	658	5005
Mexico City	694.3	2161
Buenos Aires	348.2	3854

SQL/OGIS Query

(from "Spatial Databases -- A Tour")

Query: Find the names of all countries that are neighbors of the United States (USA) in the Country table.

```
SELECT C1.Name AS "Neighbors of USA"
FROM Country C1,Country C2
WHERE Touches(C1.Shape,C2.Shape)=1
AND C2.Name ='USA '
```

Note: Spatial operator **`Touches`**() is used in the WHERE clause to join the Country table with itself. This query is an example of **spatial self join** operation.

23

Major Issues

- Euclidean geometry is continuous, whereas computers (and databases) are discrete.
- Simple questions, such as "Is point P on line L?" may be difficult to answer.
- The answer may depend on the approach to discretization.

24

An Approach: "Realms"

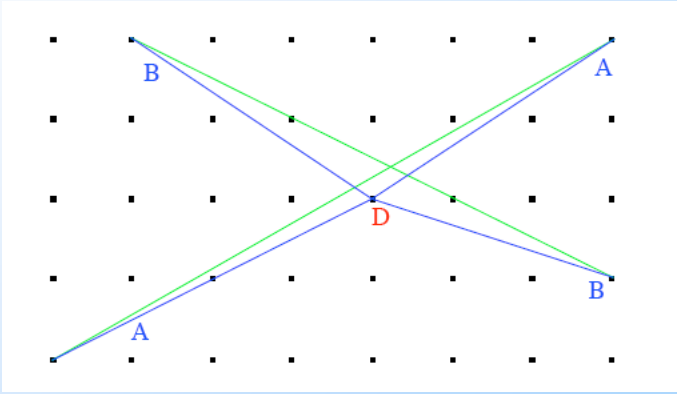
(Ralf Hartmut Güting, Tutorial Spatial Database Systems)

- *Realm*
- (intuitive notion): Complete description of the geometry (all points and lines) of an application.
- *Realm*
- (formally): A finite set of points and line segments defined over a grid such that:
  - (i) each point or end point of a segment is a grid point
  - (ii) each end point of a segment is also a point of the realm
  - (iii) **no realm point lies *within* a segment**
  - (iv) any two distinct segments do neither intersect nor overlap

25

"Bending truth" to fit the Realm model

(Ralf Hartmut Güting, Tutorial Spatial Database Systems)



can lead to some anomalies.

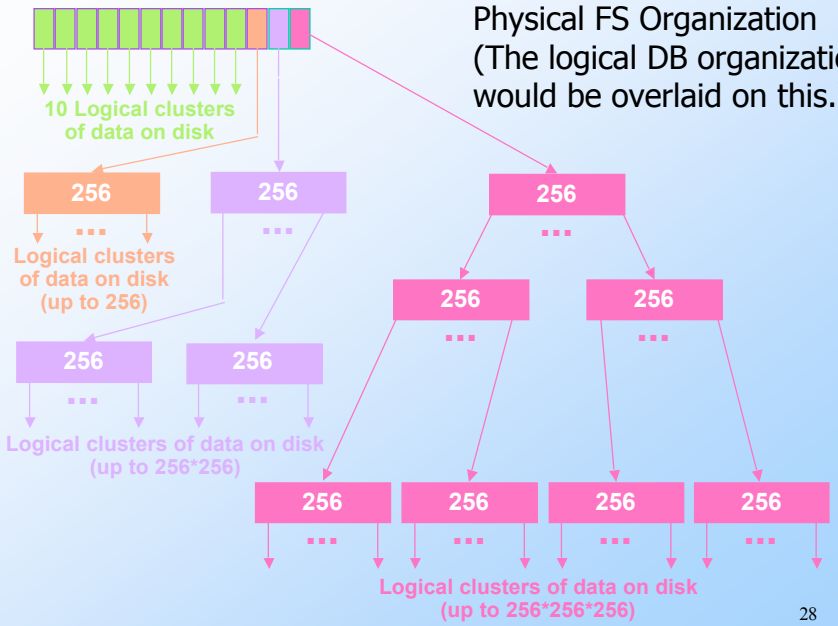
26

Disk File Organization: General (prior to refining for spatial DBs)

- Database data is typically stored on disk
- Meta-data (data describing the data is there too)
- Data stored on disk in blocks (or "sectors")
- Each block has a physical **address**
- Some blocks used for indexing other blocks
- Purpose of indexing: organize data by particular attribute (e.g. numerical attribute ⇒ index can be searched to find range of blocks of interest without looking at *all* blocks)

27

Example: UNIX Physical FS Organization (The logical DB organization would be overlaid on this.)



28

Logical → Physical Indexing

- Example: Index on an integer attribute
- The index could be a table giving ranges of values and pointing to blocks where those values can be found:
  - 0-5: block 15392
  - 6: blocks 1421, 21537, 126
  - 7-100: block 17
  - 101-999: block 32174
- Given a query that requires data values in range 200-300, we would know to confine the search to block 32174
- The index itself would be stored on some block or blocks on the disk.

29

Using Indexes for Joining, etc.

- For operations such as join on indexed relations, can compute the set of blocks to be examined based on commonality of indexed values.
- This is much faster than examining the entire content of each relation.

30

Disk Organization for Spatial DBs

- Quad trees, oct trees, KD trees
- R trees
- Grid files
- Space-filling curves

31

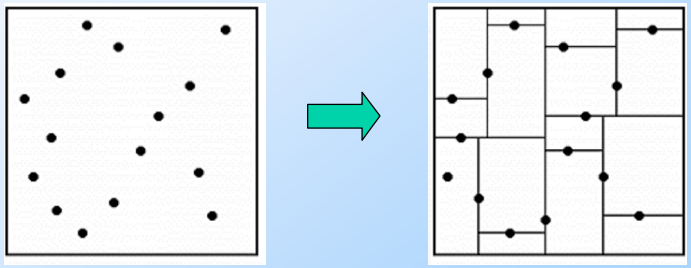
Quad Trees, Oct Trees

- Extension of "trie" concept
- Common in graphics, physical simulation

32

KD Trees (K = # dimensions)

- Extension of binary search tree
- Recursively bisect set of points (based on median) until each subdivision is one point.
- Useful in nearest-neighbor search and others.



33

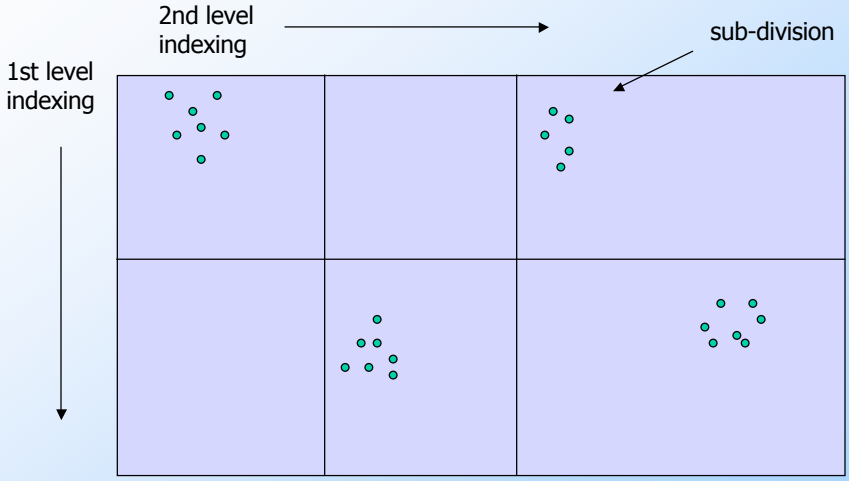
Grid Files

(Jürg Nievergelt, Hans Hinterberger, Kenneth C. Sevcik)

- Extends integer indexing idea to multiple dimensions.
- Therefore could be used for spatial data.
- Divides space non-uniformly, depending on **occupation density.**
- **Buckets** oriented toward ***physical*** implementation.
- Several **sub-divisions** may map to one bucket.

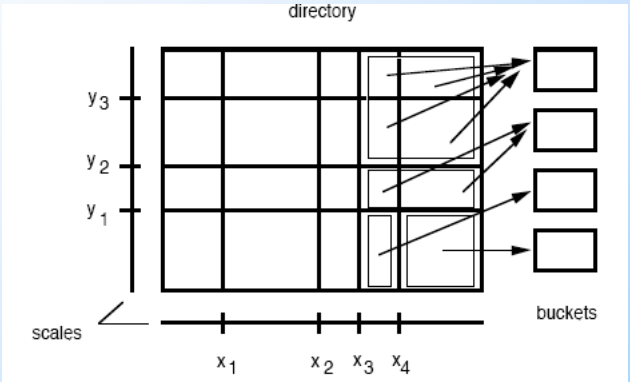
34

Grid File Example: 2-space



35

Buckets vs. Sub-Divisions



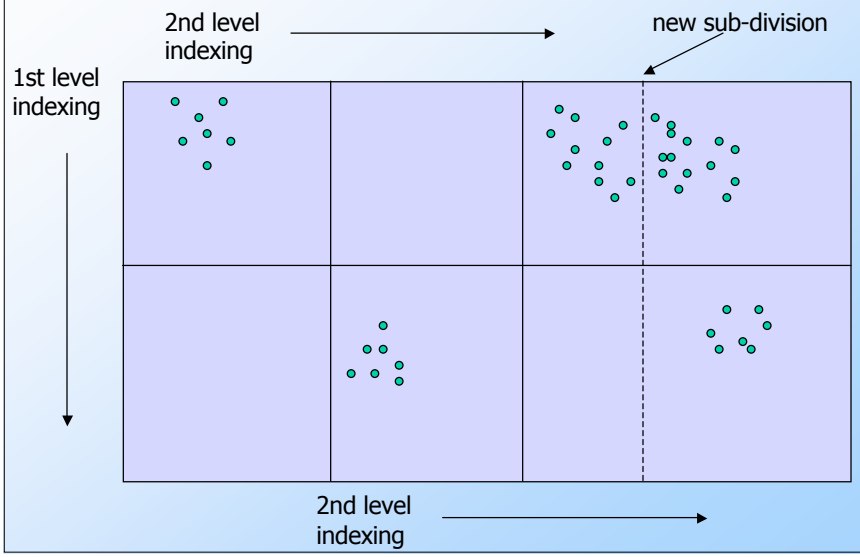
36

Sub-Dividing Grid File

- Buckets depend on how many points (records/tuples) will fit in a block, not to do with the spatial distribution of records.
- When a bucket gets too large:
  - The bucket is split
  - The points are redistributed in indices

37

Grid File Example: Expansion

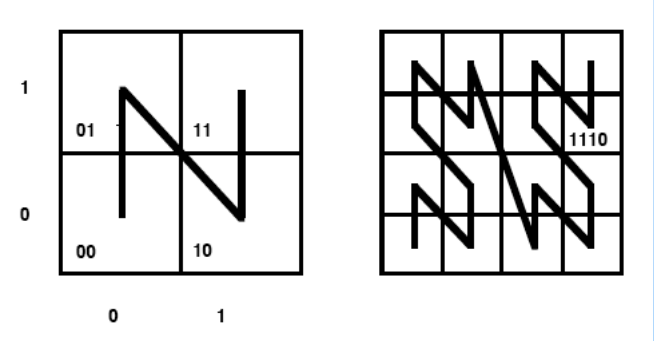


38

Space-Filling Curves

An application for a Mathematical Curiosity

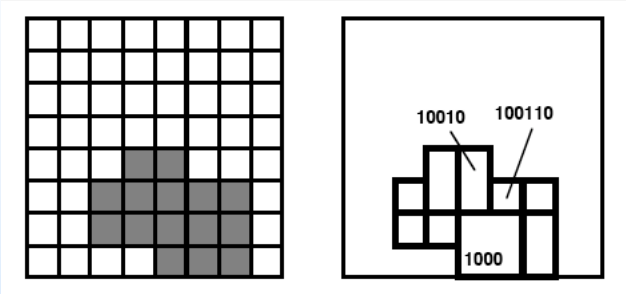
(Ralf Hartmut Güting, Tutorial Spatial Database Systems)



"Z" model of space organization and addressing

39

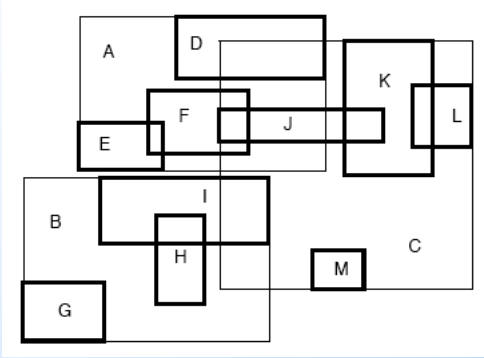
Any Shape Representable



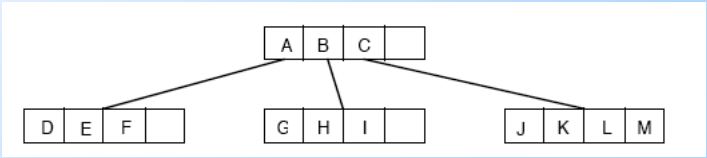
as a string of Z indices

40

R-Trees (Region Trees)

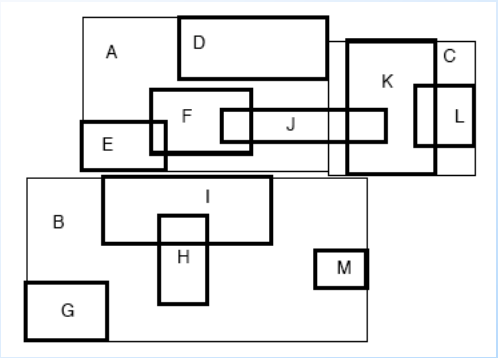


Keep single object in one sub-tree.



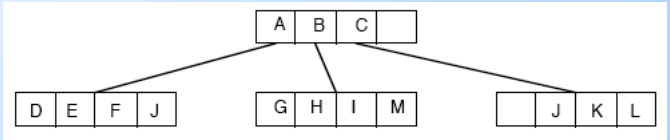
41

R+-Trees



Keep leaf regions disjoint.

Split similar to B-trees, as necessary.



42

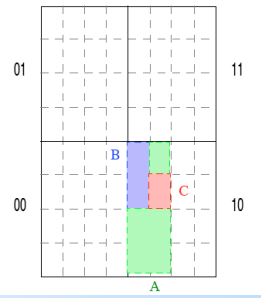
Geometric Operations

- Tree structure can play a role in operations such as intersection, union, difference.
- Generalized Search Trees (GiST; Hellerstein, Naughton & Pfeffer '95) generalize B trees, R trees, etc. and can implement numerous operations

43

Filter&Refine Example

- Suppose using Z representation
- To find whether two objects intersect, compare Z indices



0110, 10, 10010, 100110, 10110, ..

0111, 100, 1010, 1011, 1101

44