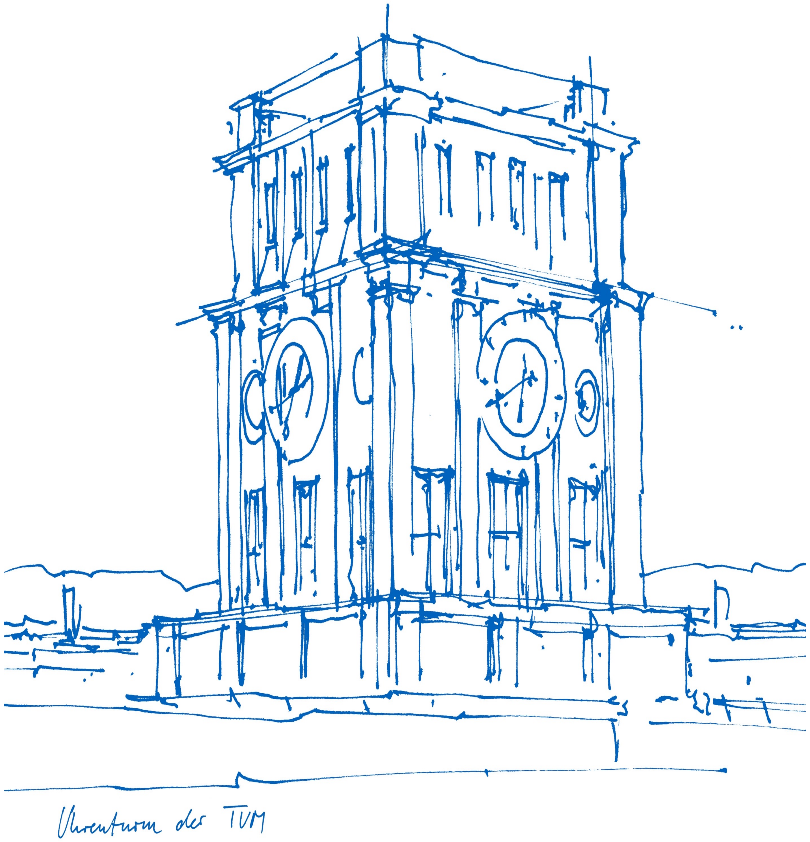
Geoinformation

Prof. Dr.-Ing. Liqiu Meng

Chair of Cartography and Visual Analytics

WS 24/25



List of Lectures

L1: Introduction

L2: Spatiotemporal representations and databases

L3-L4: Spatial data analysis

L5: Cartographic techniques

L2: Spatiotemporal representations and databases

I. Spatial representations

II. Temporal representations

III. Data structures

IV. Index methods for spatial data

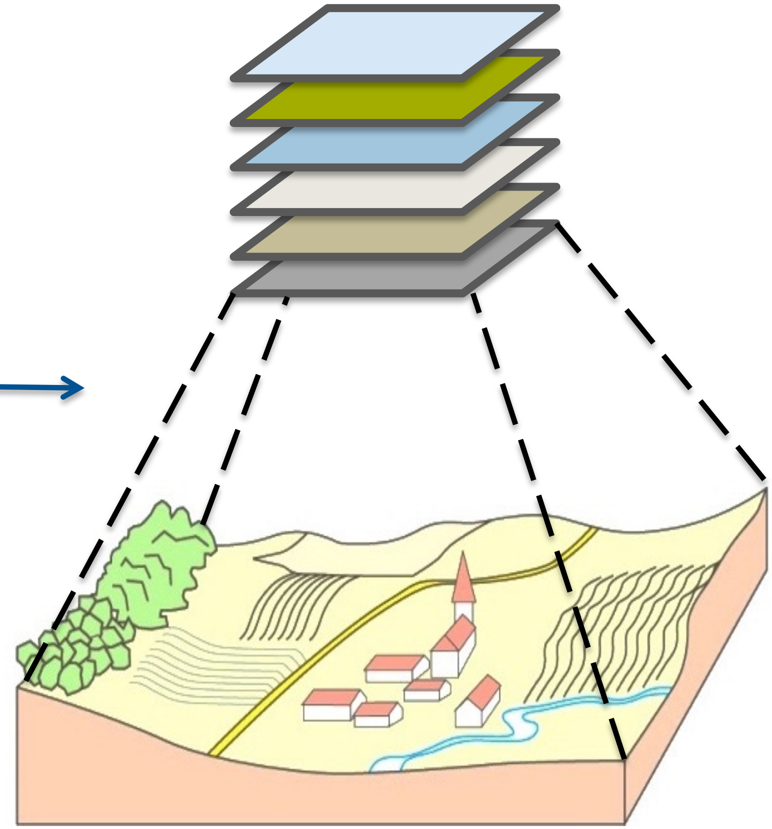
I. **Spatial representations**

II. Temporal representations

III. Data structures

IV. Index methods for spatial data

Theme layers or temporal sequences



Data modelling
the process of describing the external world in a data model

Spatial data model
It represents the geographic space.

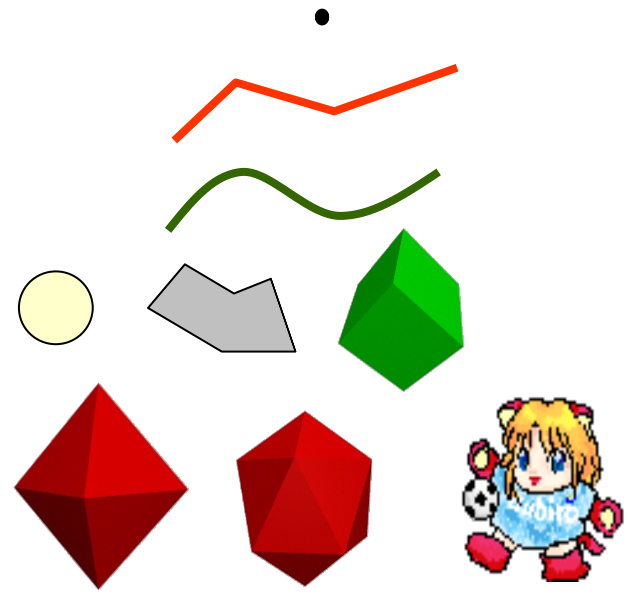
It defines categories, classes, attributes of and relationships among geo-objects.

It describes how the information is structured in a database and made accessible.

I.1 Entity modeling

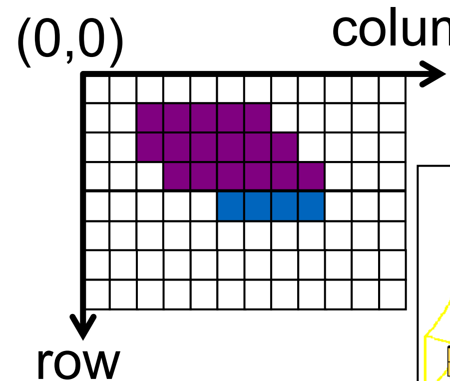
– **Vector data** model
based on points, lines, polygons, polyhedron, compound entities

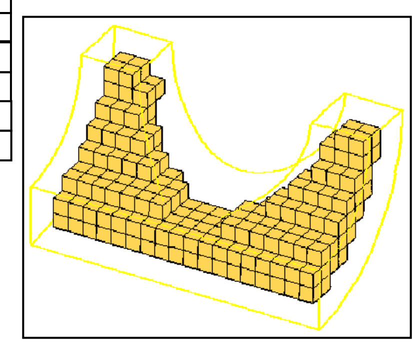
The values are stored in the entities.



– **Raster data** model
based on a matrix composed of uniform elements (pixels or voxels) of a given resolution.

The values are stored in the elements.



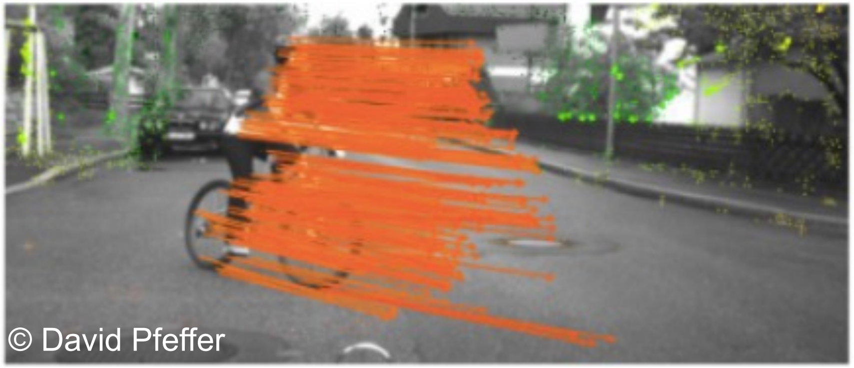


Stixel (superpixel)

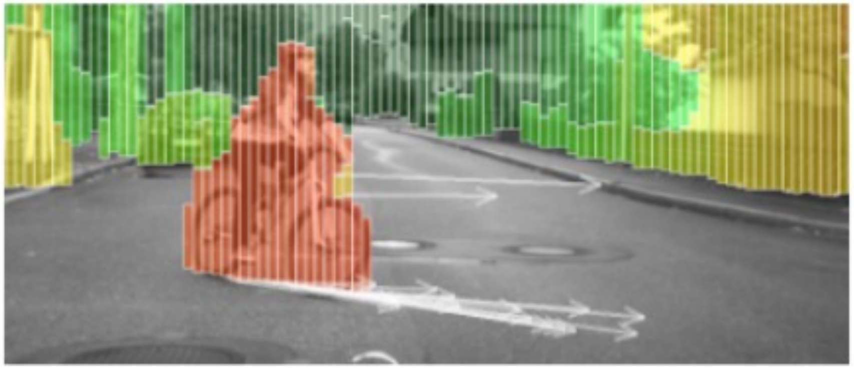
Sticklike bounding box formed by neighboring image points. Useful for motion detection in autonomous driving. A million of 3D points could be aggregated to hundreds of Stixels in varying distances.

Image pixels of different depths

Stixels of different depths



6D-vision point features of a cyclist

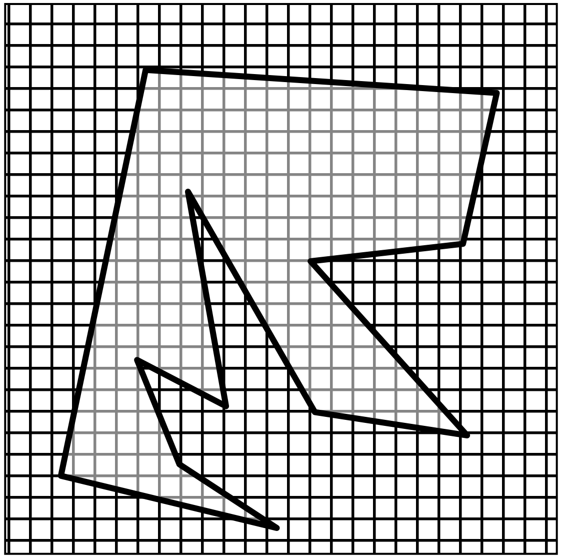


Stixels of different depths

I.2 Field modeling

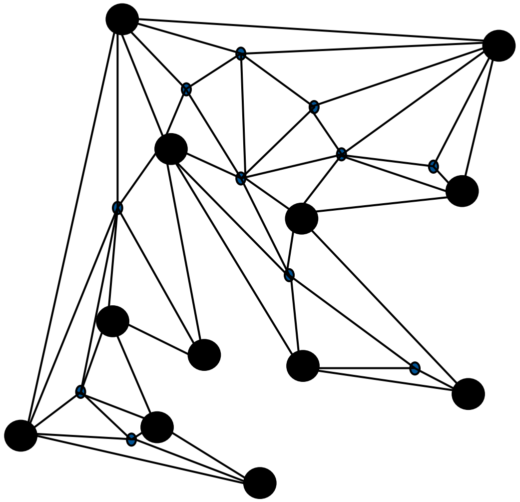
It aims to simulate fields with discrete spatial values.

It is less precise than entity modeling.

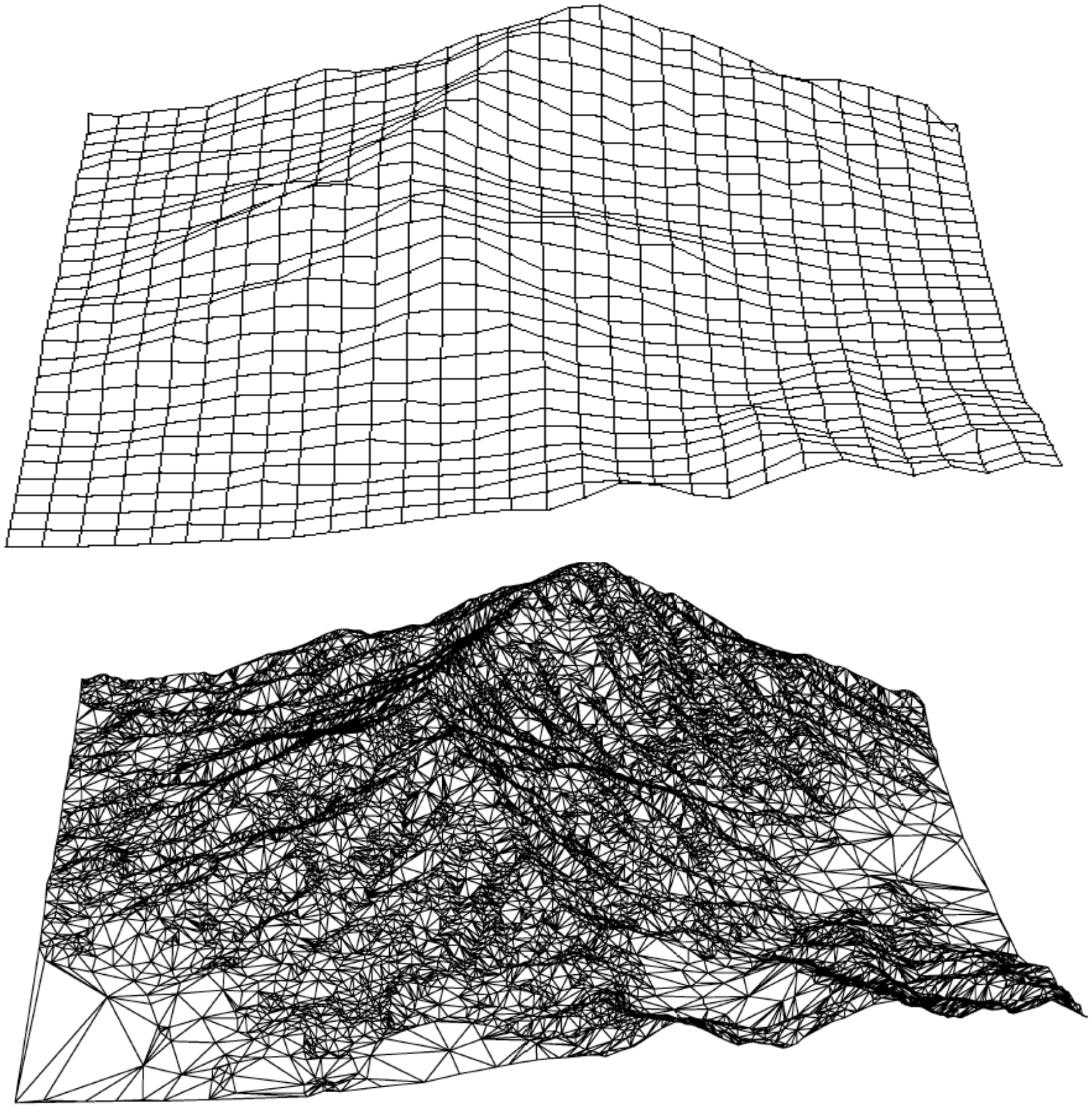


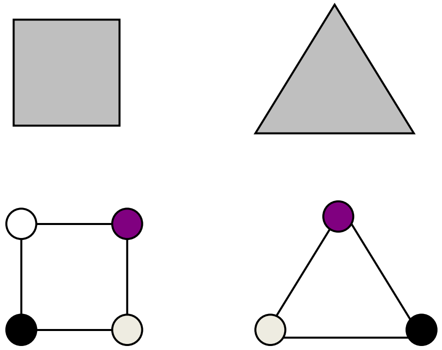
Finite element models

- regular grids
- Triangulated Irregular Networks (TIN)



The "finite element" means a limited number of individual values of a field can be observed or interpolated.





The values can be associated with nodes or cells

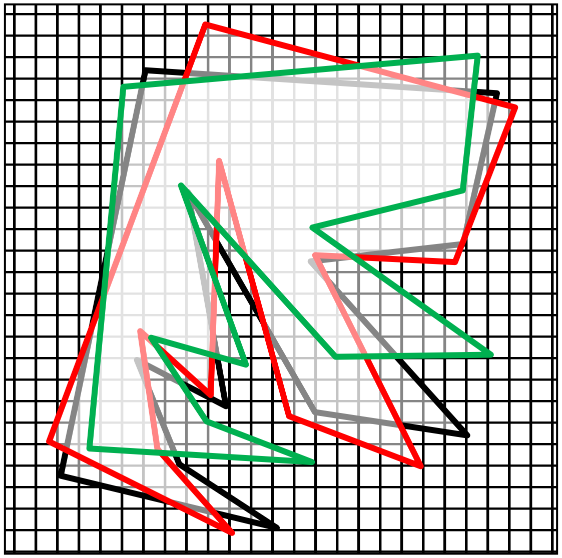
I.3 Process modeling

It aims to simulate phenomena with discrete temporal values.

It is less precise than field modeling.

Finite difference models (mostly with regular grids)

The “finite difference” means a limited number of time intervals for which individual states of a phenomena can be observed or interpolated.



I.4 Modelling of spatial relations

Topological relations

Formalizing spatial relations relies much on human intuitions. Many cognitively important spatial relations are predominantly topological.

Metric relations

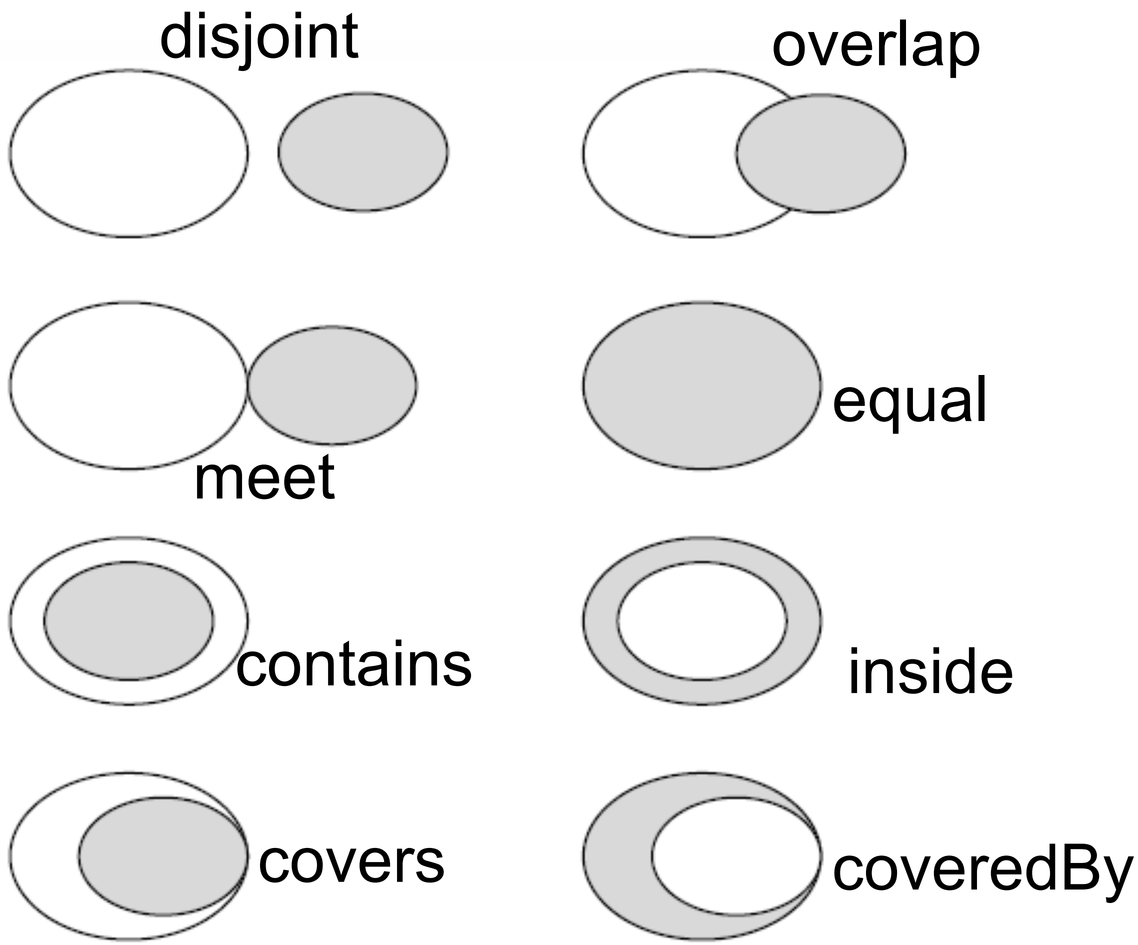
Metric information such as size, shape, distance, or direction often refines, rather than defines, spatial relations.

- Distances
  metric measures e.g. Euclidean and Taxicab distance;
  topological terms, e.g. ‘near’, and ‘far’

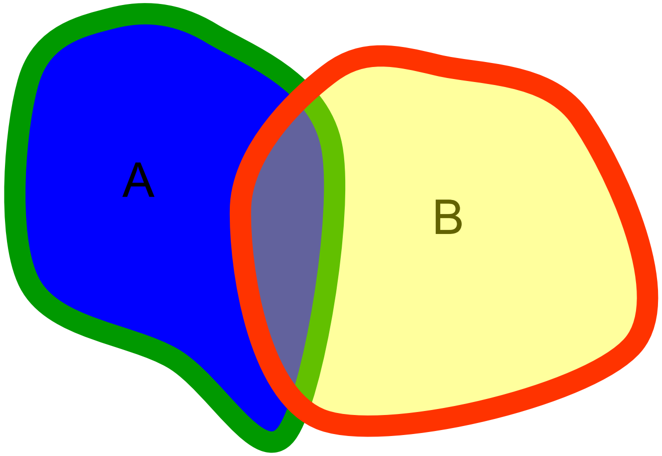
- Directions
  metric measures, e.g. degrees from a reference direction;
  topological terms in either four or eight directions

- Reference frames
  based on cardinal directions for outdoor spaces;
  viewer-centred for bodily spaces

8 topological relations between two polygons



9 relations between two polygons with thick boundaries



Each entity is composed of an interior, a boundary, and an exterior.

interior (A^o)
boundary (δA)
exterior (A^-)

interior (B^o)
boundary (δB)
exterior (B^-)

3^2 combinations

$$\begin{pmatrix} A^\circ \cap B^\circ & A^\circ \cap \partial B & A^\circ \cap B^- \\ \partial A \cap B^\circ & \partial A \cap \partial B & \partial A \cap B^- \\ A^- \cap B^\circ & A^- \cap \partial B & A^- \cap B^- \end{pmatrix}$$

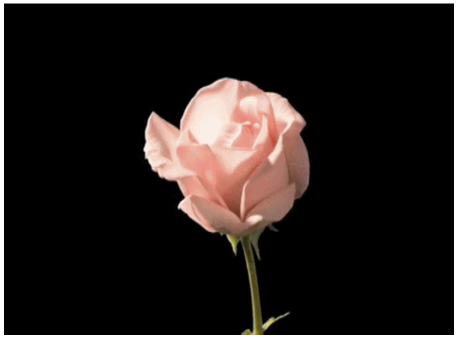
I. Spatial representations

II. **Temporal representations**

III. Data structures

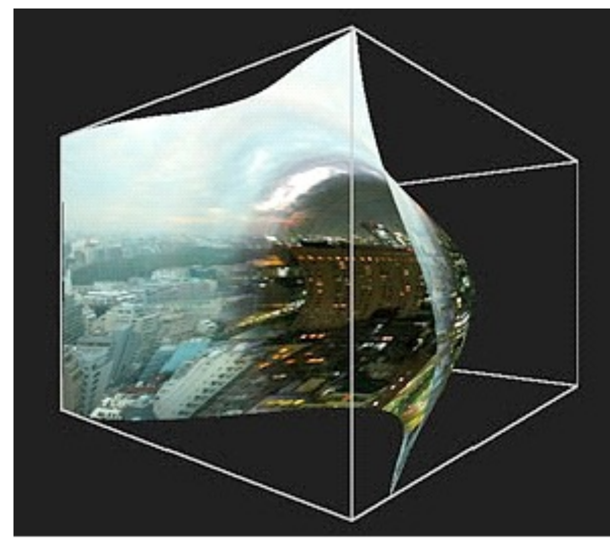
IV. Index methods for spatial data

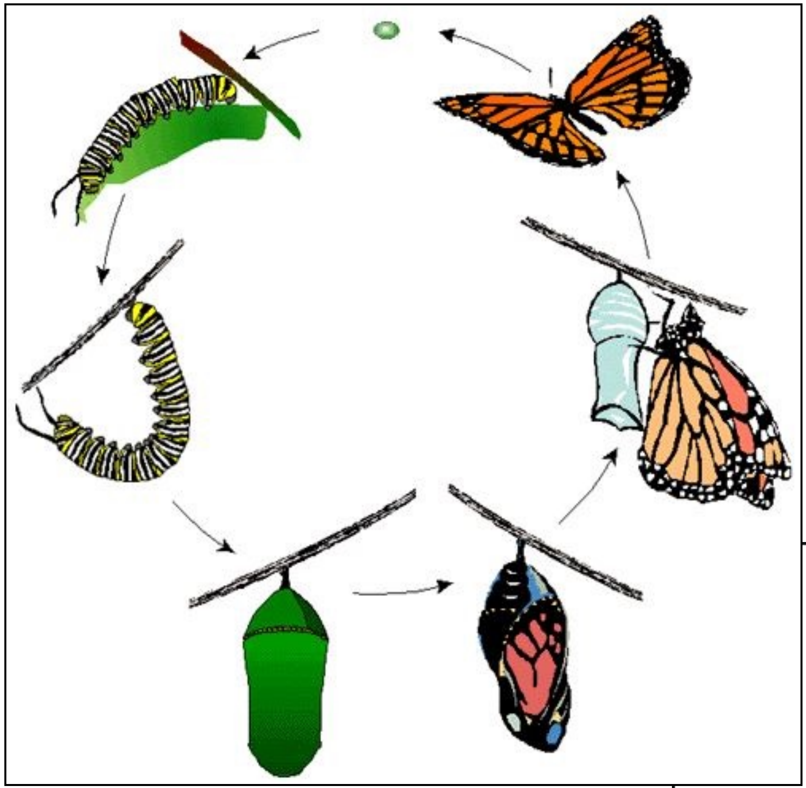
Time is reflected in changes



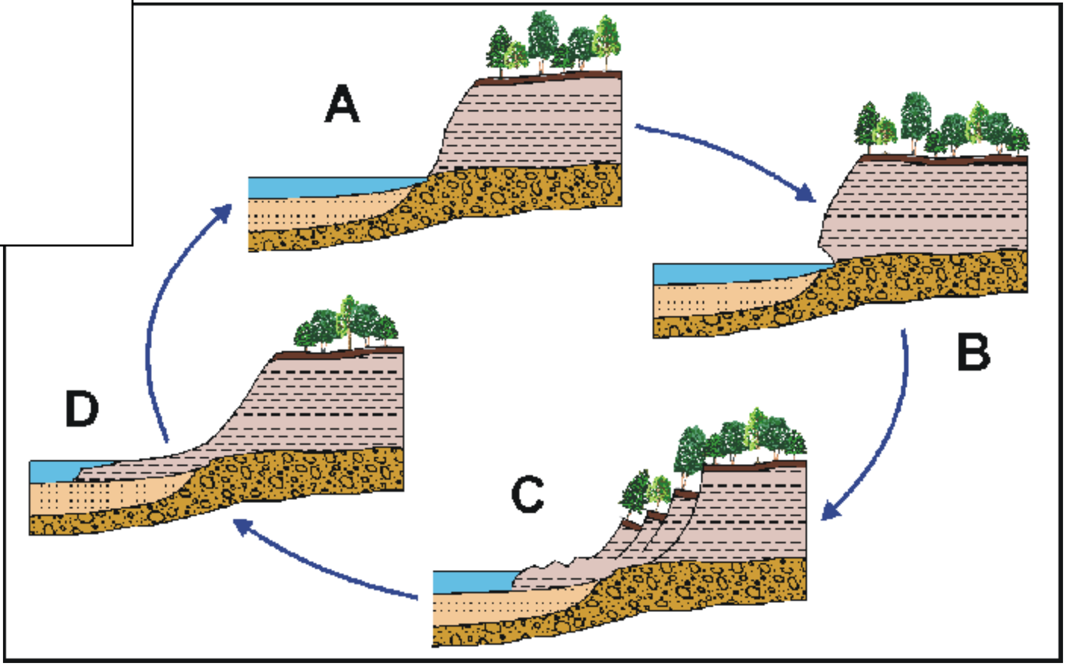








Generally predictable life cycles

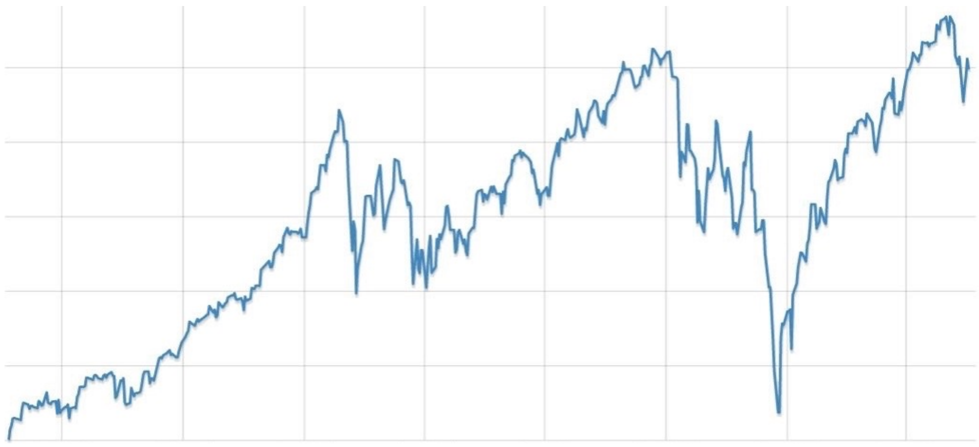


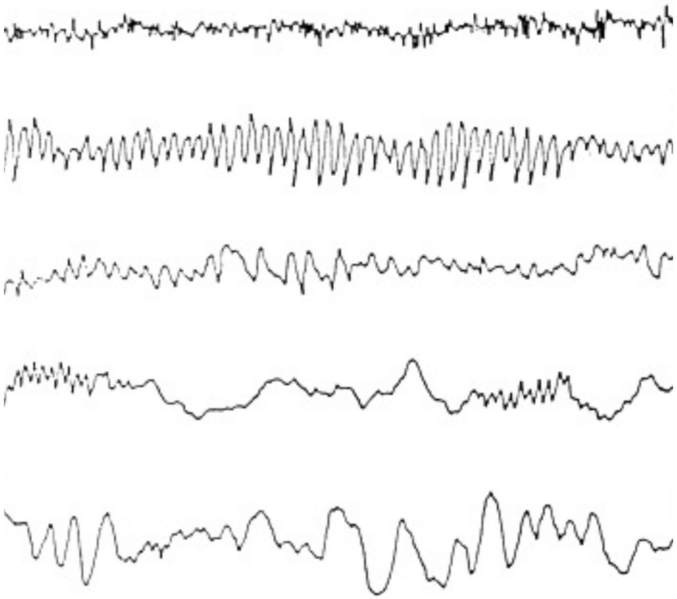
Patterns of changes

random, monotonic, periodic move, deteriorate,
expand / shrink, deform, split, merge

Attributes describing changes

- Rate: quick / slow, sudden / gradual
- Frequency: always, often, sporadic, once
- Duration: short / long
- Amplitude: large / small





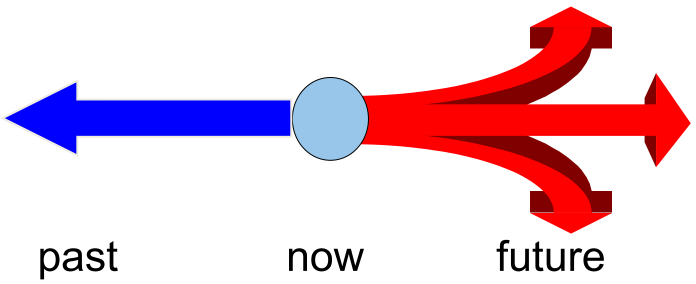
Temporal units

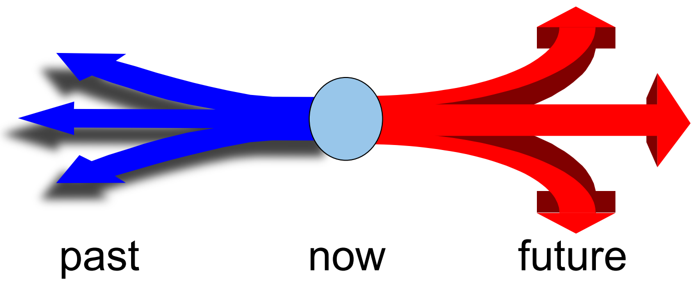
seconds, minutes, days, seasons…

The smallest unit is called a *chronon*.

Branching

Multiple realities over time. This allows comparison among simulation results or between simulation results and observed data (e.g. weather forecast, acting time of an archaeological site).





II.1 Modelling of temporal relations

Relation	Symbol	X	Y
X before Y	<		
X equals Y	=		
X meets Y	m		
X overlaps Y	o		
X during Y	d		
X starts Y	s		
X ends Y	e		

Association between temporal elements (e.g. temporal distance, topological relations)

Combination of different temporal distributions (temporal overlays that act as Boolean operators)

Transformation between temporal scales (generalization and extrapolation)

Application scenarios:

- Cross-document event ordering in natural language processing (tense, since, then, during, meanwhile, before, after, long ago, at once etc.)
- Coordination of production lines or services
- Tracking for Corona infections
- Detection of causality relations
- To-do list and deadlines

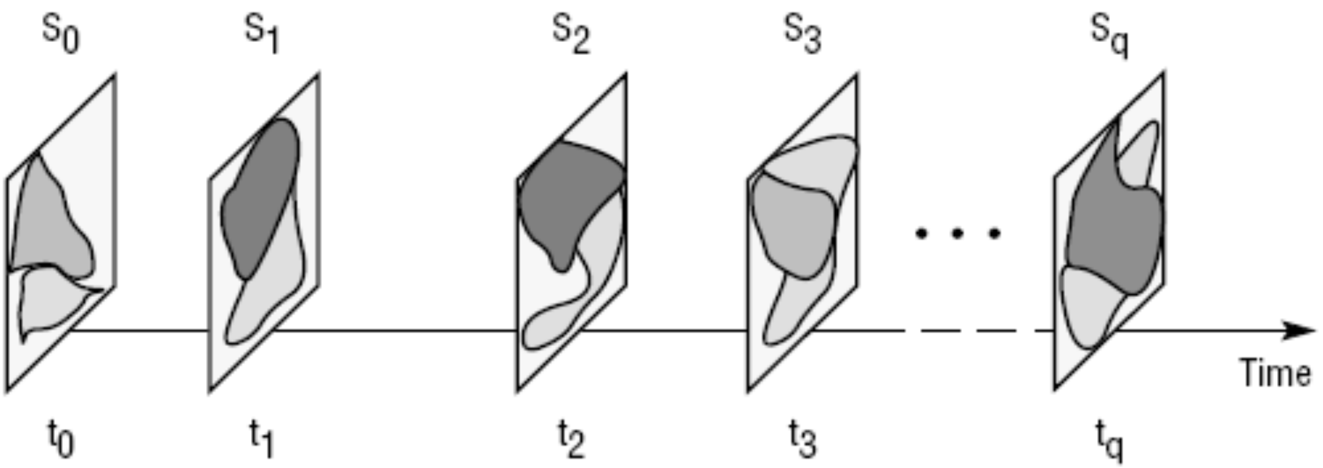
II.2 Modelling of spatio-temporal relations

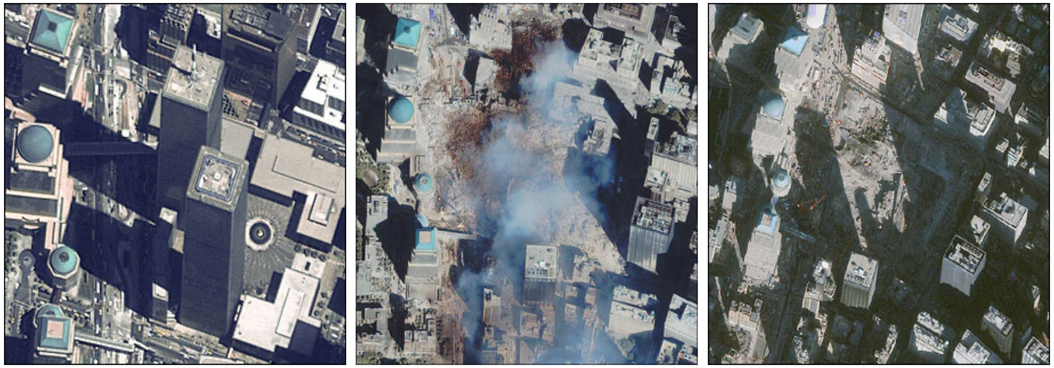
II.2.1 Location-based representations

1. a temporal series of spatial ‘snapshots’

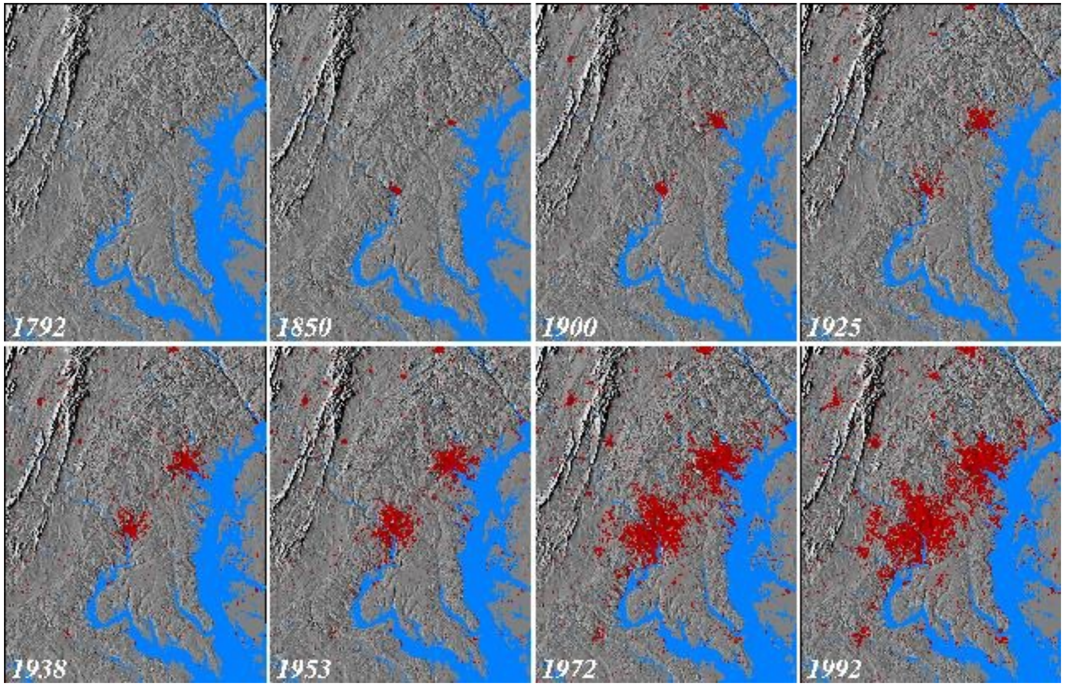
   It is straightforward The temporal interval can vary.

   Challenge: large redundancy, long retrieval time, missing events in the interval





Snapshots of the 9.11 terror attack



Snapshots of population growth

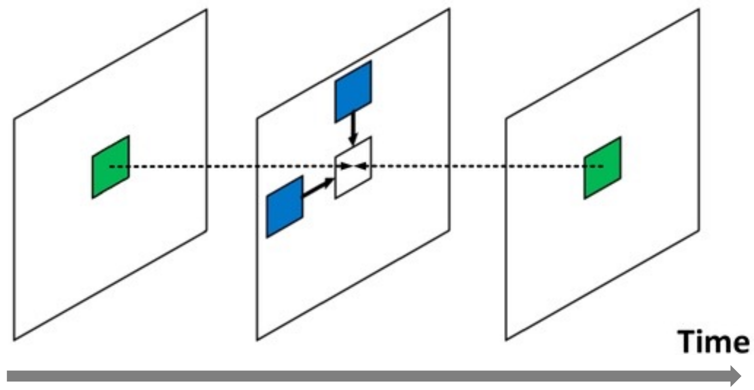
2. temporal grid that stores individual changes

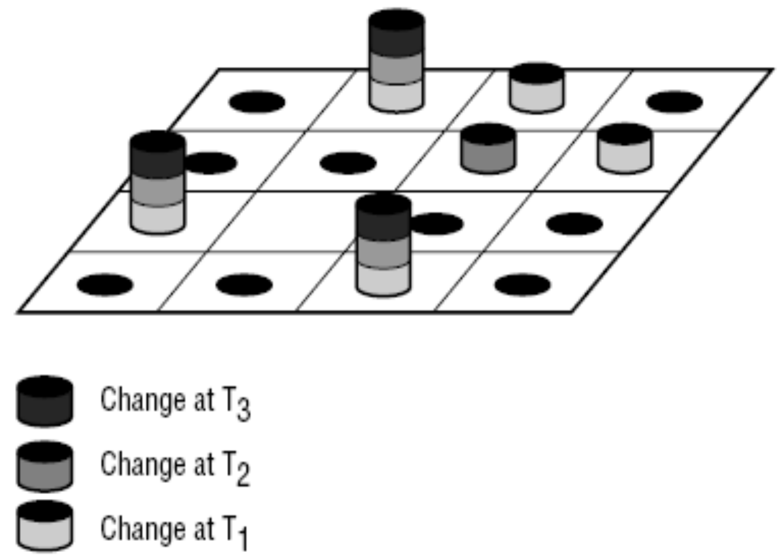
It has less redundancy. A change is expressed by the new value and the time at which the change occurred.

Challenge:

tedious grid-wise updating,

difficult to determine a suitable grid size



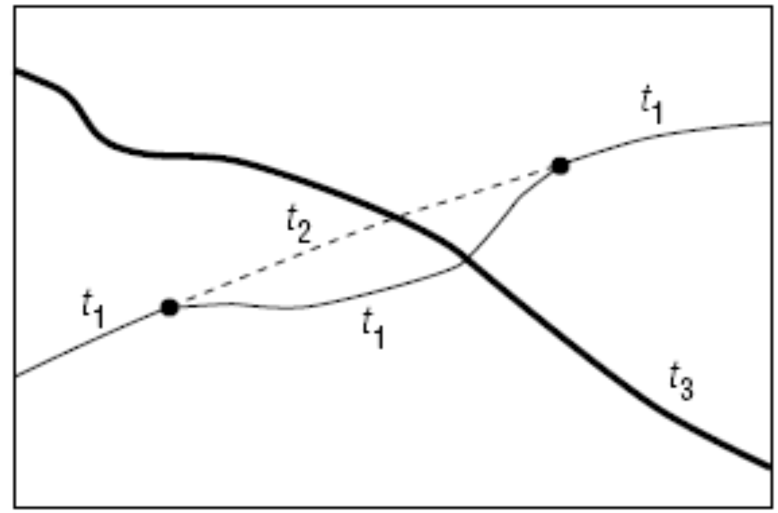


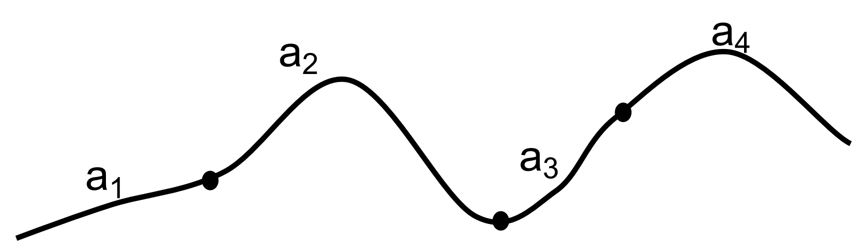
II.2.2 Entity-based representations

1. Geometric changes are incrementally recorded and maintained.

   Challenge:

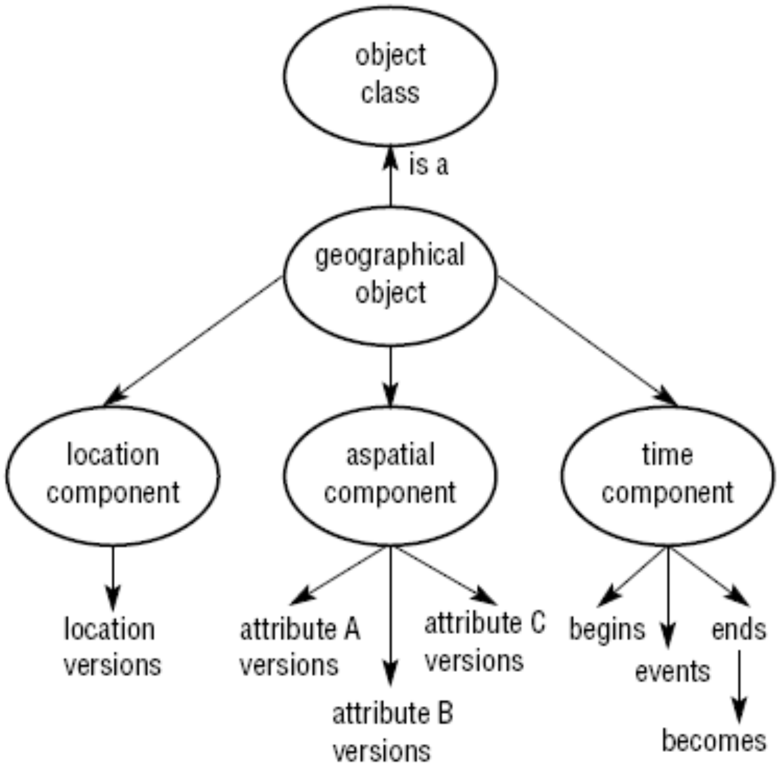
   difficult to maintain the identity of individual entities when split or merge occurs too often and when semantic changes happen too often to parts of an entity.

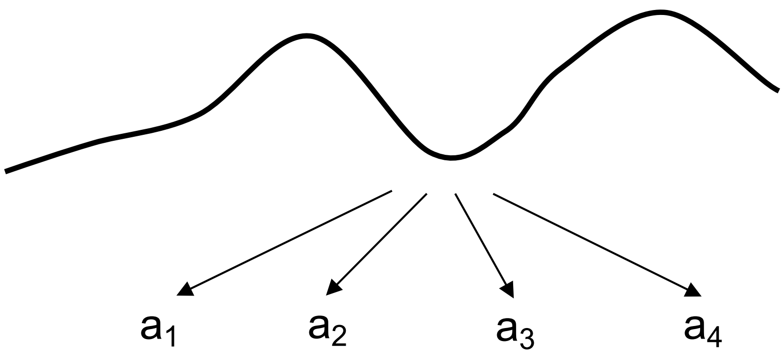




2. Storing the spatial, temporal and aspatial components of an entity as an integral object. Rules for determining how to split or merge objects are stored as part of the entity definitions.

Challenge:
difficult to determine general rules for splitting or merging



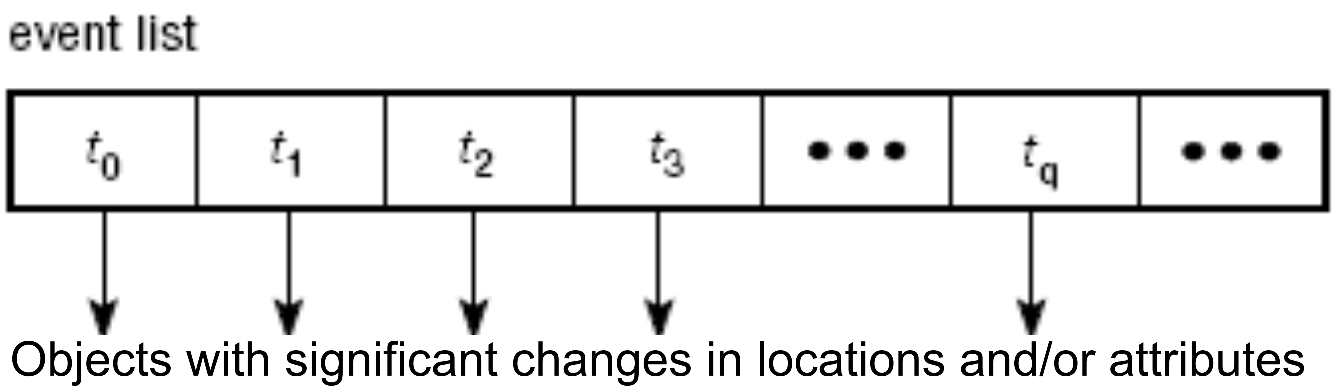


II.2.3 Time-based representations

The time associated with each change is stored in an order from a starting date (t_0) to some other later date (t_n).

Each location along the timeline can be associated with a particular set of events.

Challenge: how to find possible relations between the events



Example: 2024 Top 10 events

I. Spatial representations

II. Temporal representations

III. **Data structures**

IV. Index methods for spatial data

A data structure is a logical way of storing data for later access.

Different data structures can be applied for different tasks.

Data structures are sometimes chosen to fit the algorithms.

Some algorithms are embedded in data structures.

III.1 Array

- Array is a list of items that have the same data type and occupy a contiguous area of storage.
- Items in a multi-dimensional array are accessed by multiple orders.
- Arrays have fixed or variable sizes.

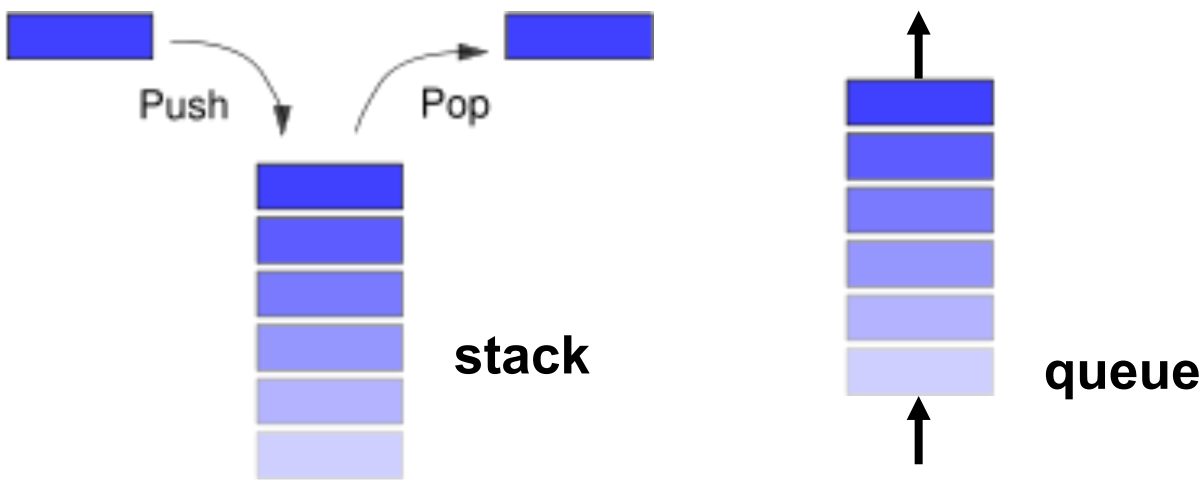
$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \qquad A_{1,1} = 1,\ A_{1,2} = 2,\ \ldots,\ A_{3,2} = 8,\ A_{3,3} = 9$$

Arrangement of a 2D array using RC-convention:

row-based access order: 1,2,3,4,5,6,7,8,9
column-based access order: 1,4,7,2,5,8,3,6,9

III.2 Stack and queue

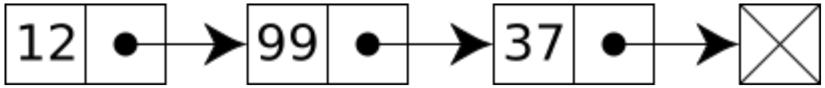
- A stack is a list of items accessed following the principle of Last In First Out (LIFO). It inserts / removes an item to / from the top of the stack.
- A queue is a list of items accessed following the principle of First-In-First-Out (FIFO). It inserts an item to the rear position but removes an item from the top.



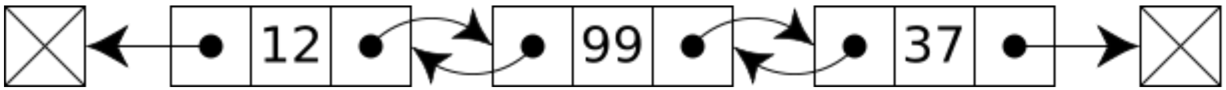
III.3 Linked list

– A linked list consists of a sequence of items, each containing arbitrary data fields and one or two "links" pointing to the next and/or previous items.

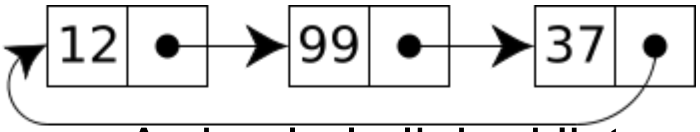
– Linked lists permit insertion and removal of items at any point in the list.



A singly-linked list



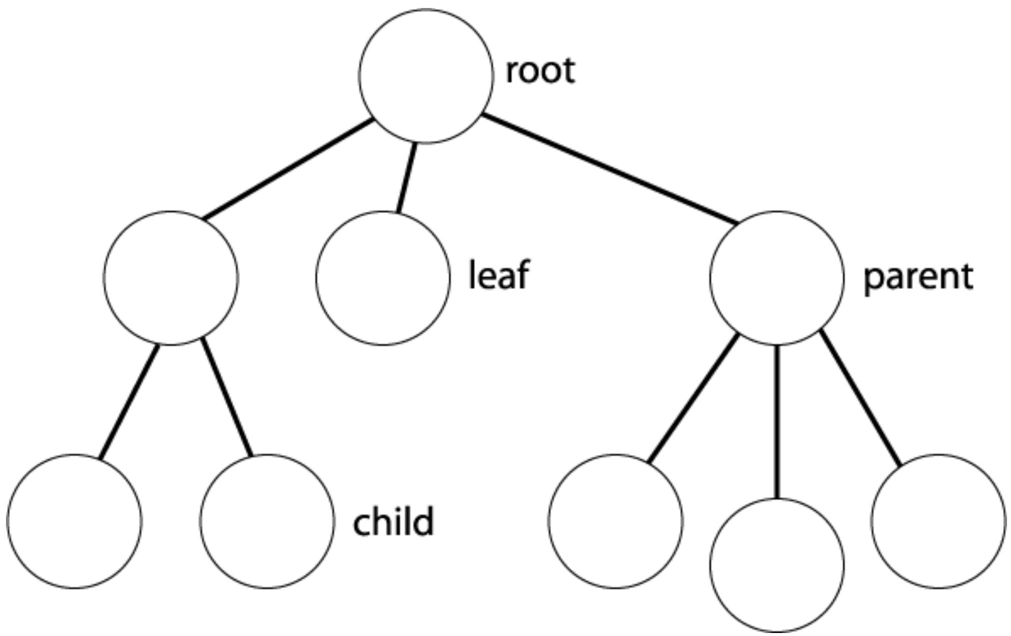
A doubly-linked list



A circularly-linked list

III.4 Tree (hierarchy)

- A tree is a set of linked nodes. A node may contain values, pointers or a tree of its own.
- The **depth** of a node is the length of the path to its root. The number of nodes at the same level is called **width**.
- A tree structure can be accessed by depth or width.

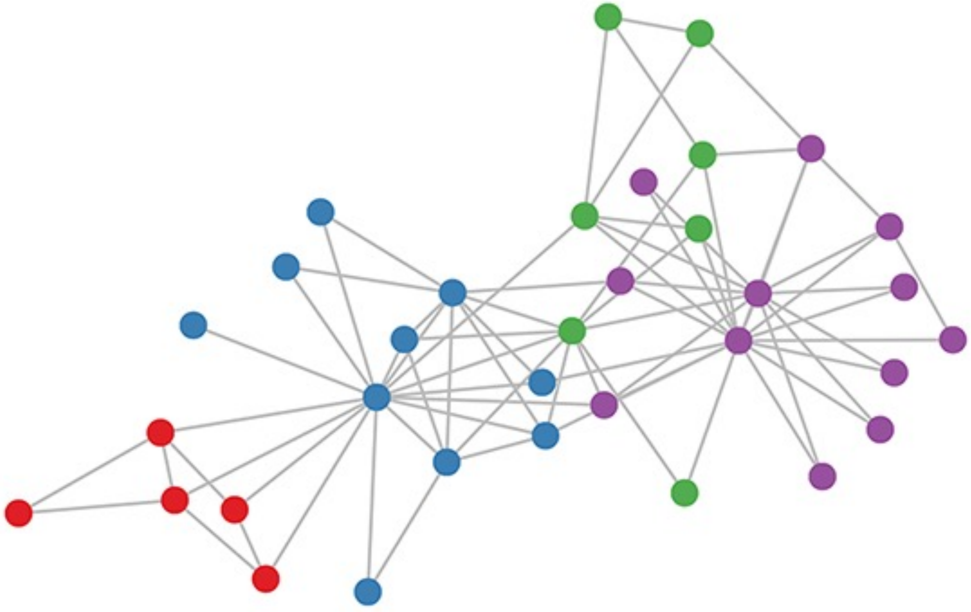


III.5 Graph (network)

A graph is a set of nodes (vertices) *V* connected by edges *E* in one or both directions *G=(V,E)*.

Each vertex and edge can have one or more attributes.

A graph is typically accessed by *V* or *E* and their relationships.



I. Spatial representations

II. Temporal representations

III. Data structures

IV. Index methods for spatial data

An index is a way of rearranging the data structure for more efficient retrieval. A dataset can have indexes for its attributes.

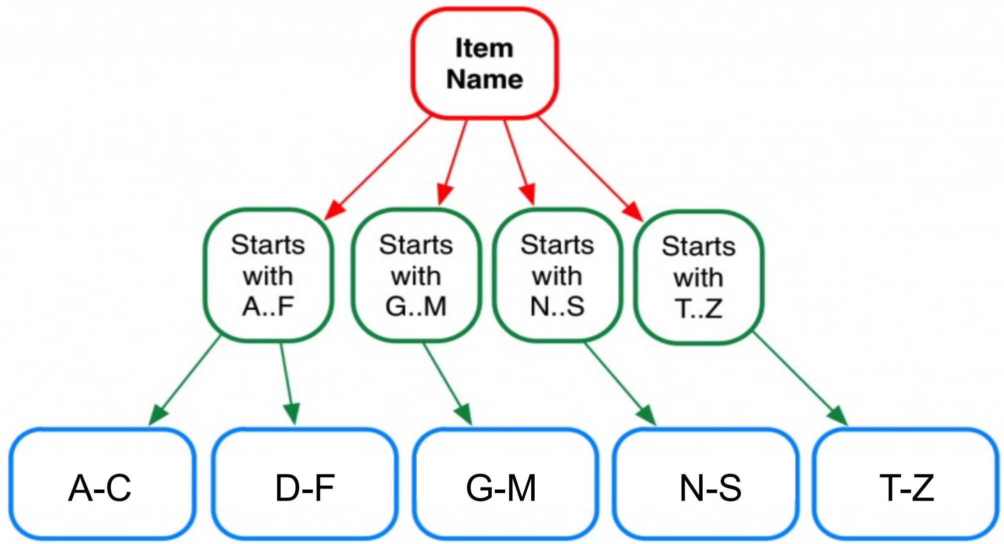
The searching time can be reduced e.g. from O(n) to O(logn).

Index (family name)

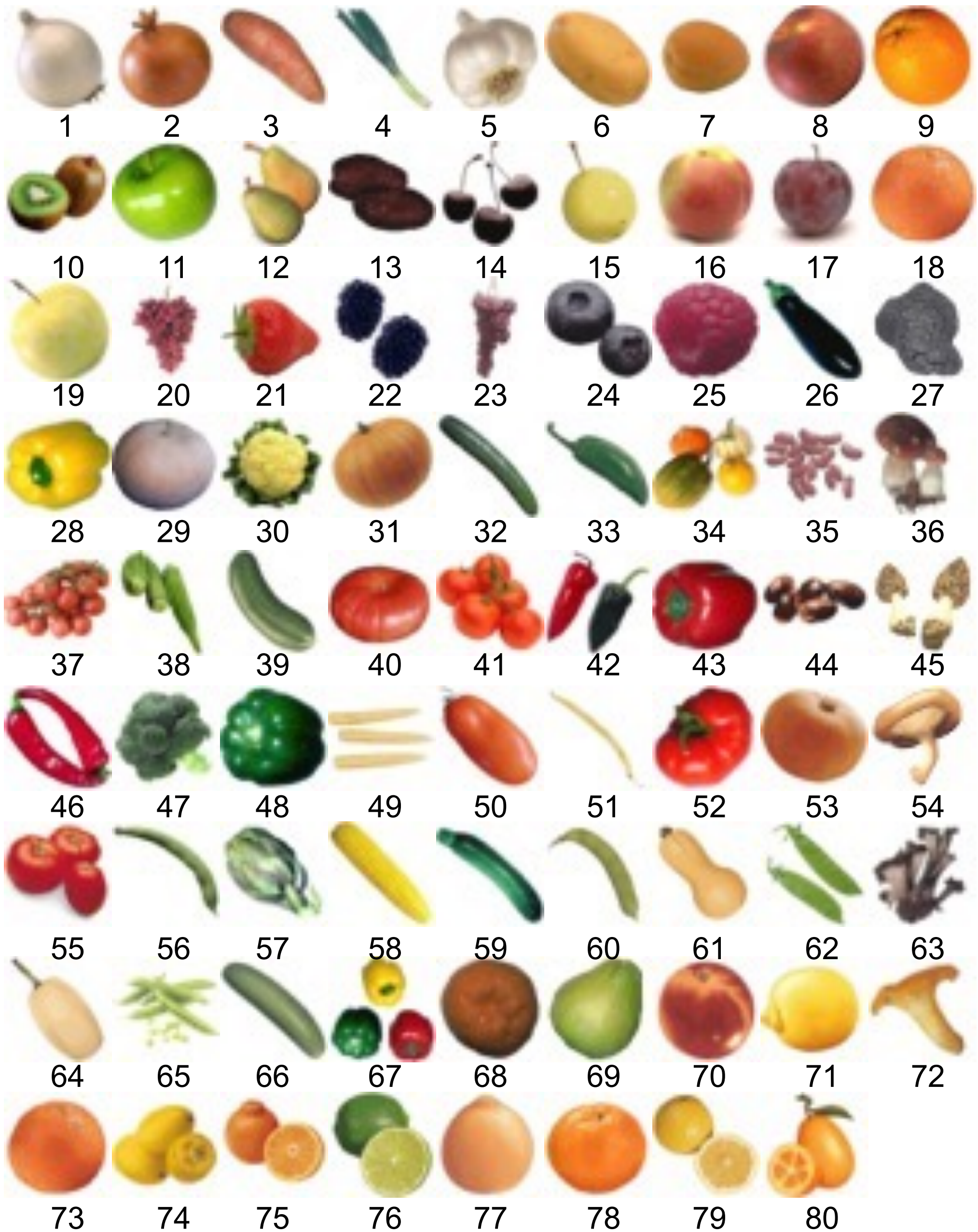
Johnson
…
Karlsson
Kopf
…
Zhang

Data set

First name	Family name	Birth date	Hometown
Ben	Johnson	1960-12-12	New York
Ulla	Karlsson	1956-12-01	Stockholm
Zhilin	Zhang	1967-04-23	Hong Kong
Matteus	Kopf	1943-08-03	Berlin



Grocery scale for fruits and vegetables

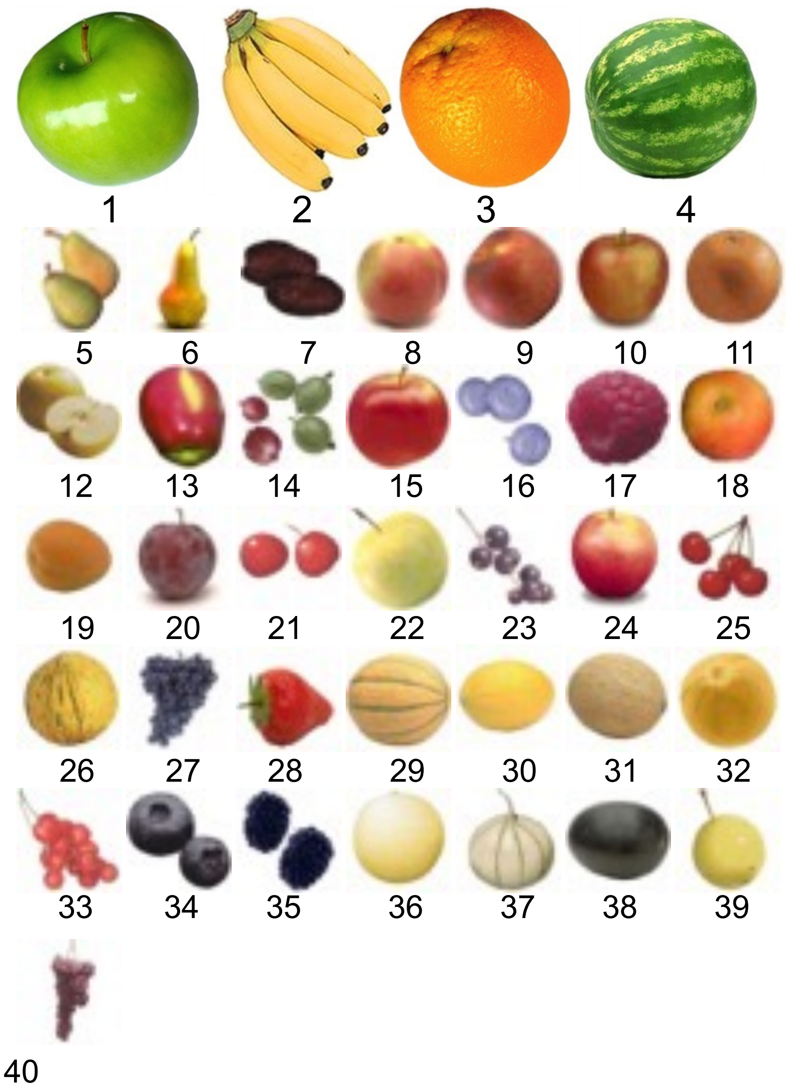


fruits

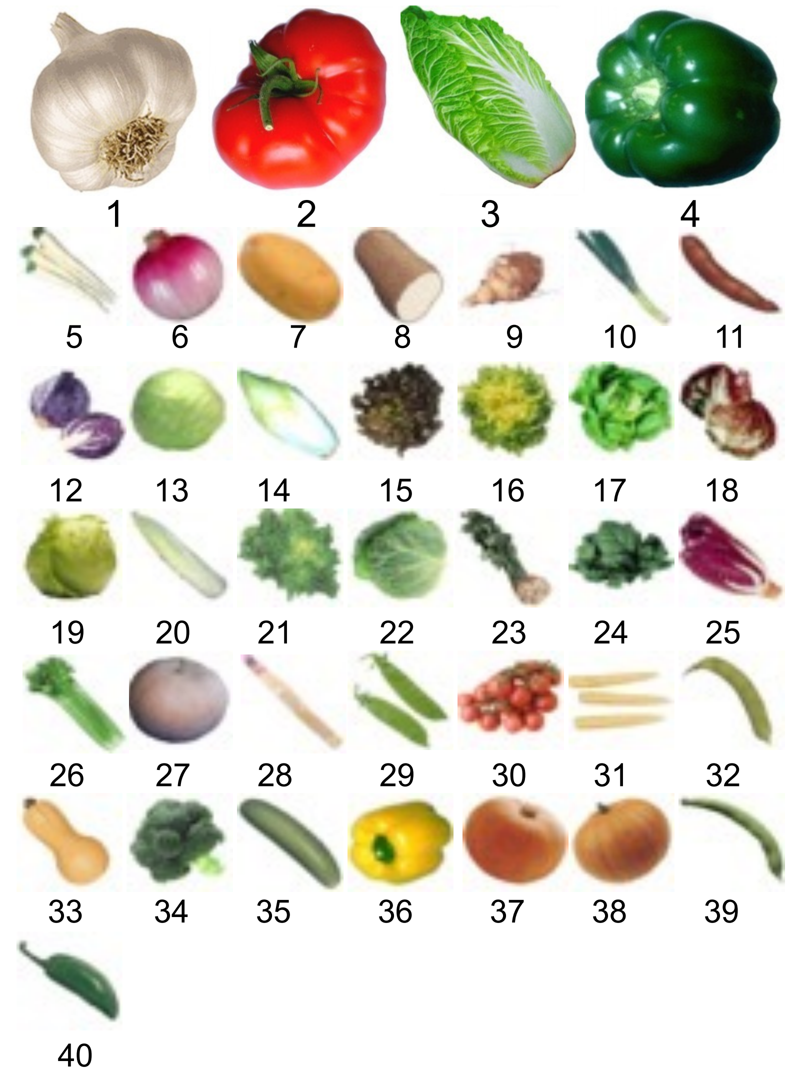
vegetables

smaller searching scope

fruits



vegetables

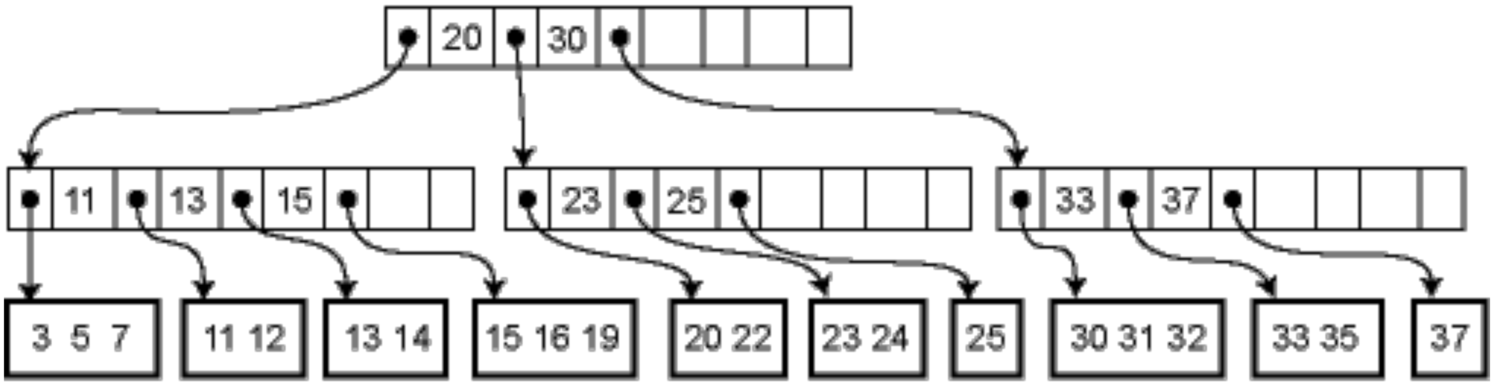


differentiation of icon sizes

IV.1 B-tree

B-tree was developed by R. Bayer and E. McCreight at Boeing Research Labs in 1971.

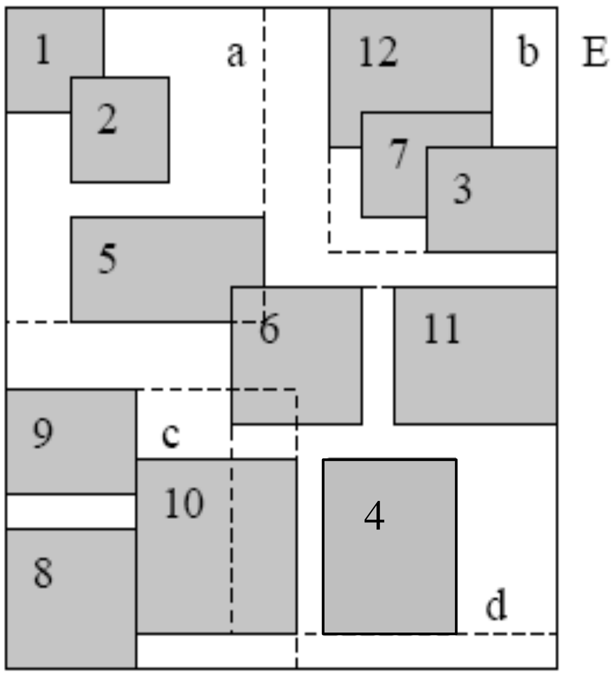
Each node contains *n* separation values to cut a tree into *n+1* subtrees. All values in the leftmost subtree are less than the first separation value, all values in the rightmost subtree will be greater than or equal to the last separation value.

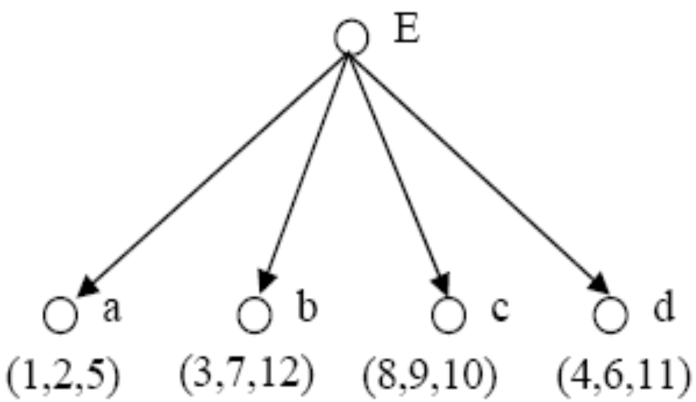


Example of a B-tree

IV.2 R-tree

Each node in the *R-tree* is associated with the minimum bounding box of all its descendants. During the search only the branches containing data are accessed. Bounding boxes may partly overlap, but each object belongs entirely to one bounding box.

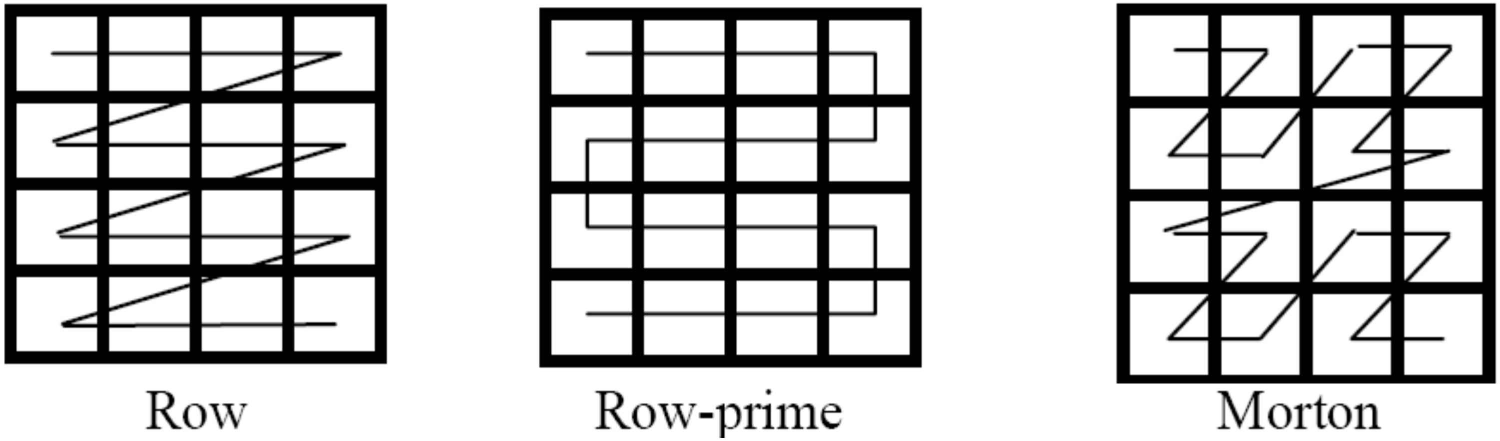




IV.3 Space filling curves

An ideal index should allow neighboring objects to have a similar index value. This should be realized by a continuous bijective function *f* that maps a 2D connected dataset R^2 onto a *1D* connected set *R*.

Theoretically, there exist no such continuous bijective functions. *Space filling curves* have pragmatic properties to be used as index functions.



Typical space filling curves

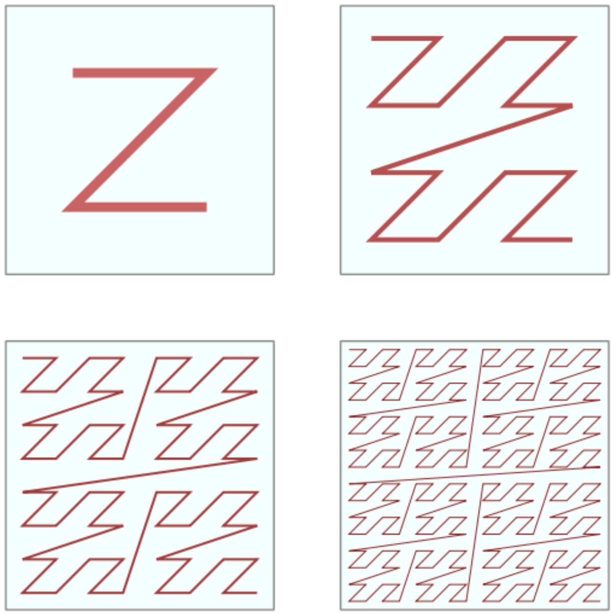
The principle of Morton key *mk*:
Convert the decimal values of *x* and *y* to binary digits.

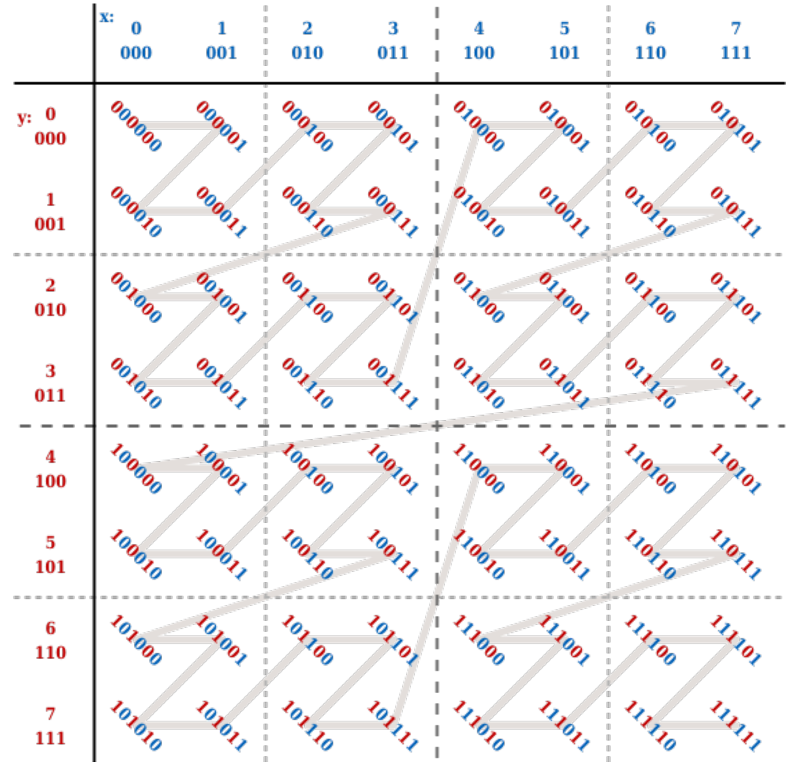
Take the last digit in *x* and place it as the last digit for *mk*. Then take the last digit in *y* and place it as the second last digit for *mk*. Proceed by taking the second last digit for x, and continue until all digits are placed. If the binary numbers *x* and *y* are not of the same length it is only to add zeros in the start of the binary number. *mk* is the decimal number converted from the binary chain.

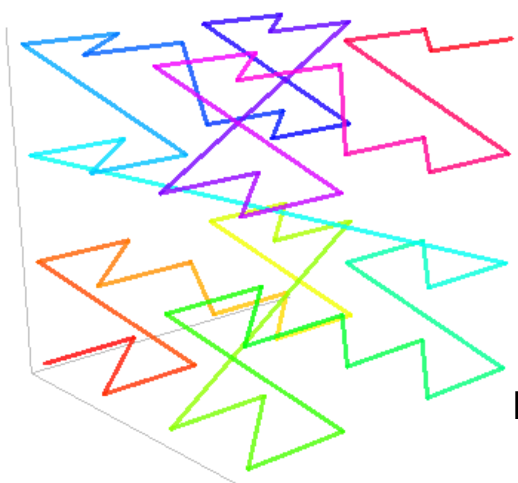
The *mk* of [10,7]

x → 1010
y → 0111

$01101110_2 = 110_{10}$







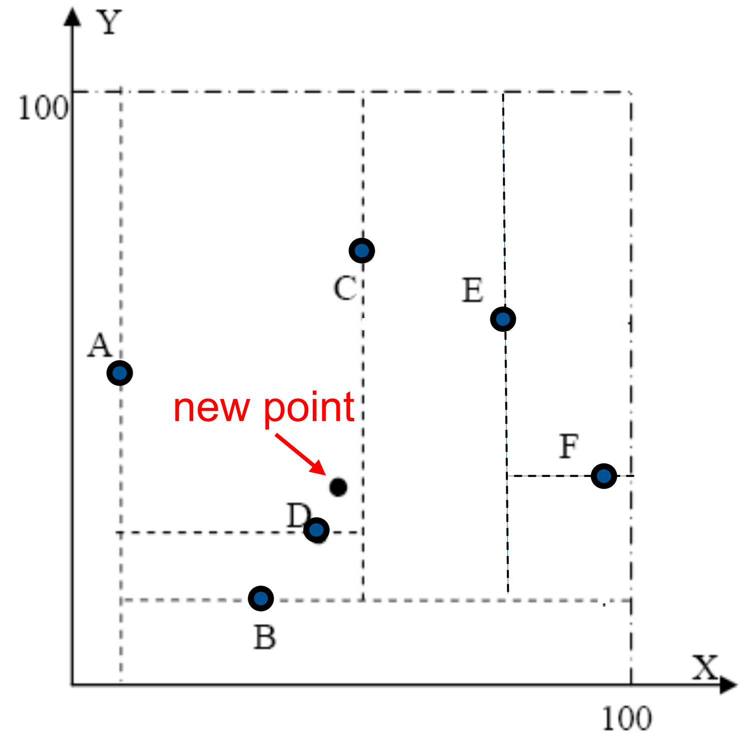
mk ist extendable to 3D

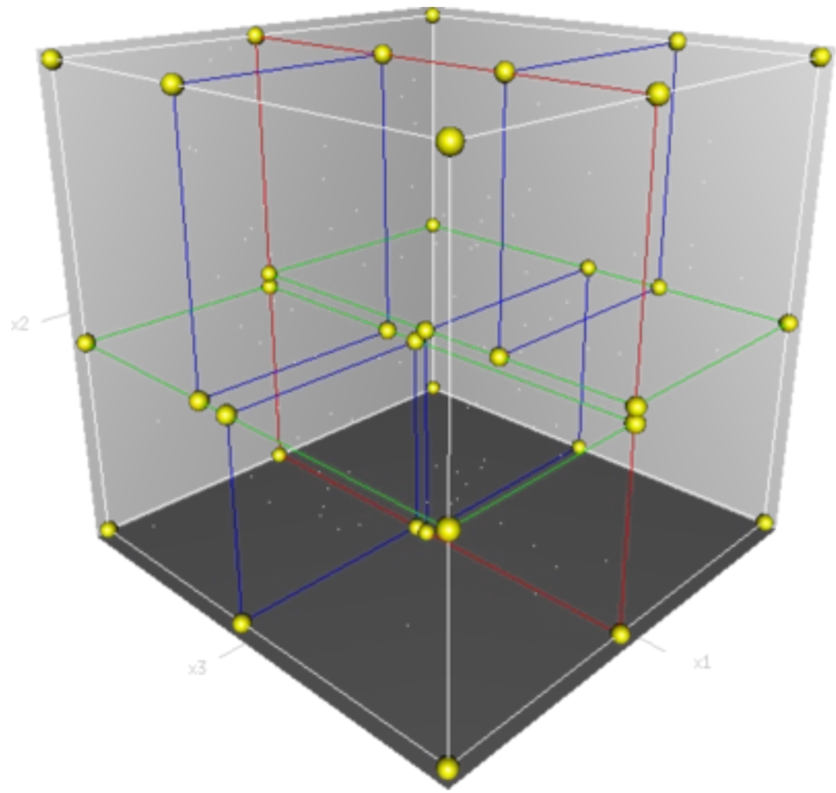
IV.4 kD-tree

In the *kD-tree* (*k*=1, 2,…) each point in the data set is a node dividing the space in two halves.

2D-tree:
The root node *A* divides the planar space vertically. The next point *B* is then placed in the space to the left or right of the root node depending on its position in the space. The new point divides the half horizontally. This procedure is then repeated for each point.

Point	X-coordinate	Y-coordinate
A	10	42
B	30	14
C	53	74
D	44	24
E	77	64
F	93	34

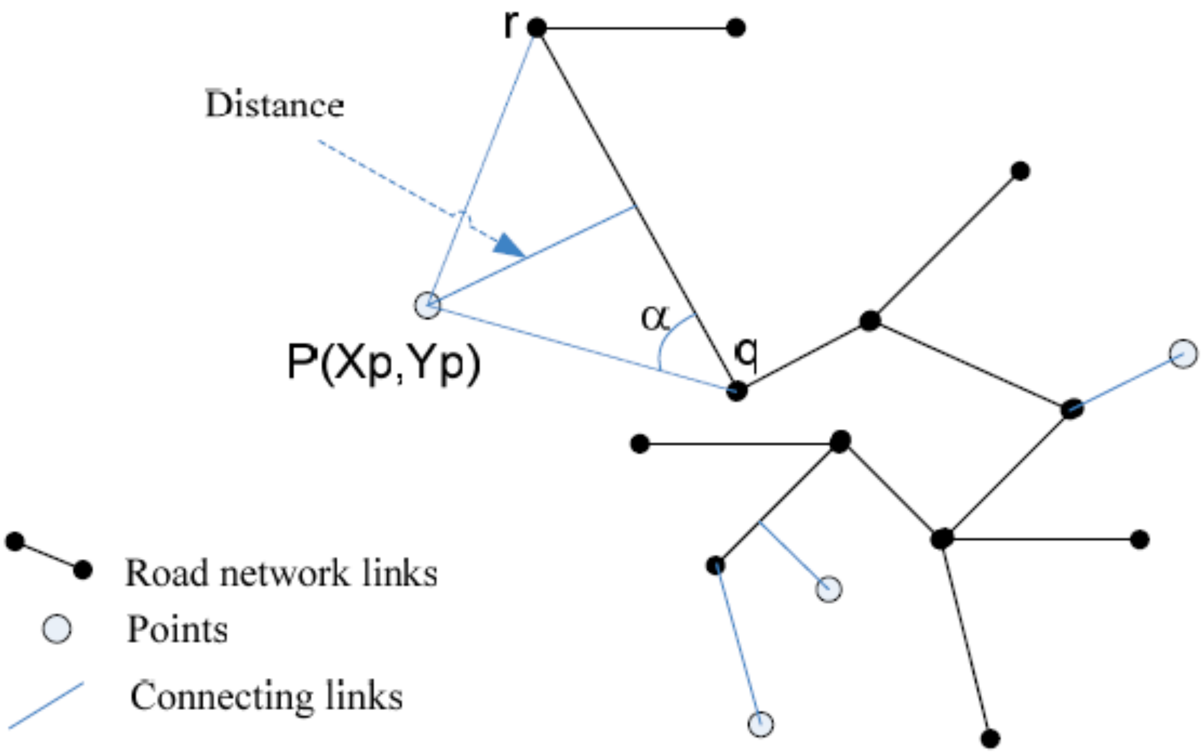




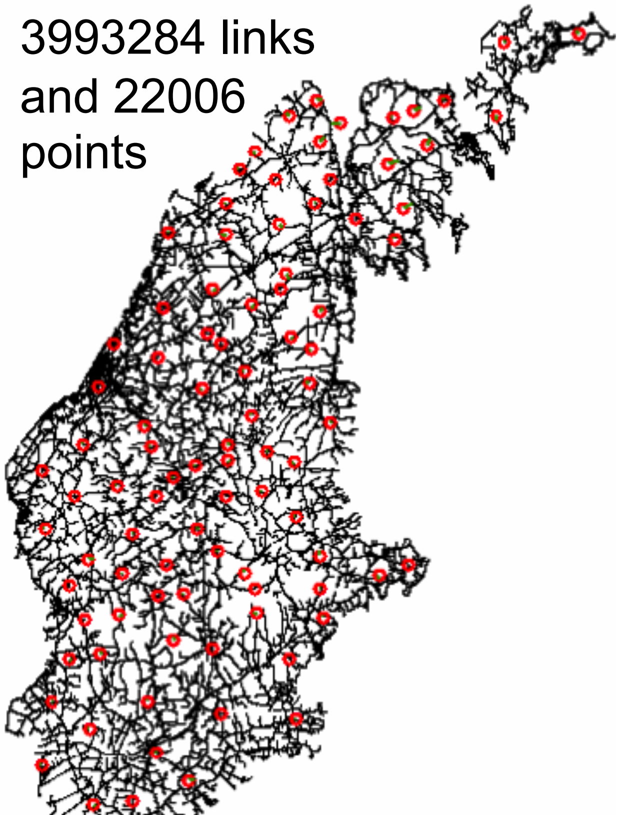
3D-tree: The first split (red) cuts the root cell (white) into two subcells, each of which is then split (green) into two subcells. Finally, each of those four is split (blue) into two subcells. The yellow balls represent the tree vertices.

An example of accessibility studies:

The task is to connect points to the closest line segments in a network. This can be performed by computing the distance between the points and all line segments, but this often leads to unacceptable processing times; therefore spatial indexes are used.



Test data with 3993284 links and 22006 points



A successful index method should rapidly create a small candidate set containing the correct solution.

The following methods were compared:

- No index
- Morton key
- kD-tree.

Index method	Time for building index	Time for connecting points
No index	--------	More than 2 h
kD-tree	1 min 31 sec	2 min 9 sec
Morton code	49 sec	2 min 9 sec

Challenges of search engine indexing (web indexing)

- Parallelism – collision between competing tasks, such as updating of the index while responding to search queries
- Compression – efficiency for large-scale searching
- Natural language processing – tokenization in multilingual context
- Search neutrality – returning the most relevant results without manipulation. This is an ethical challenge!

1. What are topological relationships and metric relationships?
2. What is the difference between vector data model and raster data model?
3. What is a Stixel and for what is it useful?
4. Why is process modeling less precise than field modeling which in turn is less precise than entity modeling?
5. How many topological relations can you find between two polygons with or without boundary width?
6. How do you understand the temporal relationships?
7. Explain the principles and challenges of location-based, entity-based and time-based representations of spatio-temporal relations.
8. Which data structures do you know and how do they work?
9. Why do we need index in a database? Use examples to explain.
10. Explain the working principles of B-tree and R-tree.
11. Which space filling curves do you know and why are they useful?
12. Explain the working principle of Morton key.
13. What is a kD-tree and how does it work?
14. Explain some of the challenges of search engine indexing.