AARHUS
UNIVERSITET

23 FEBRUAR 2011

NOVEL APPROACHES TO THE INDEXING OF MOVING OBJECT TRAJECTORIES

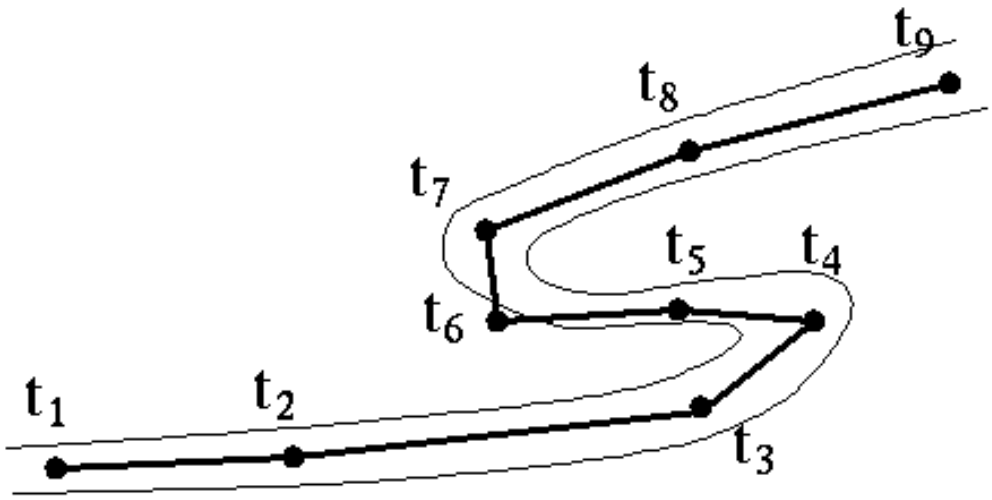
DIETER PFOSER, CHRISTIAN S. JENSEN, YANNIS THEODORIS

OVERVIEW

- Data: Trajectories
- Queries
- Data structures
  - R-tree
  - Spatio-Temporal R-tree (STR-tree)
  - Trajectory Bundle tree (TB-tree)
- Query handling
- Performance

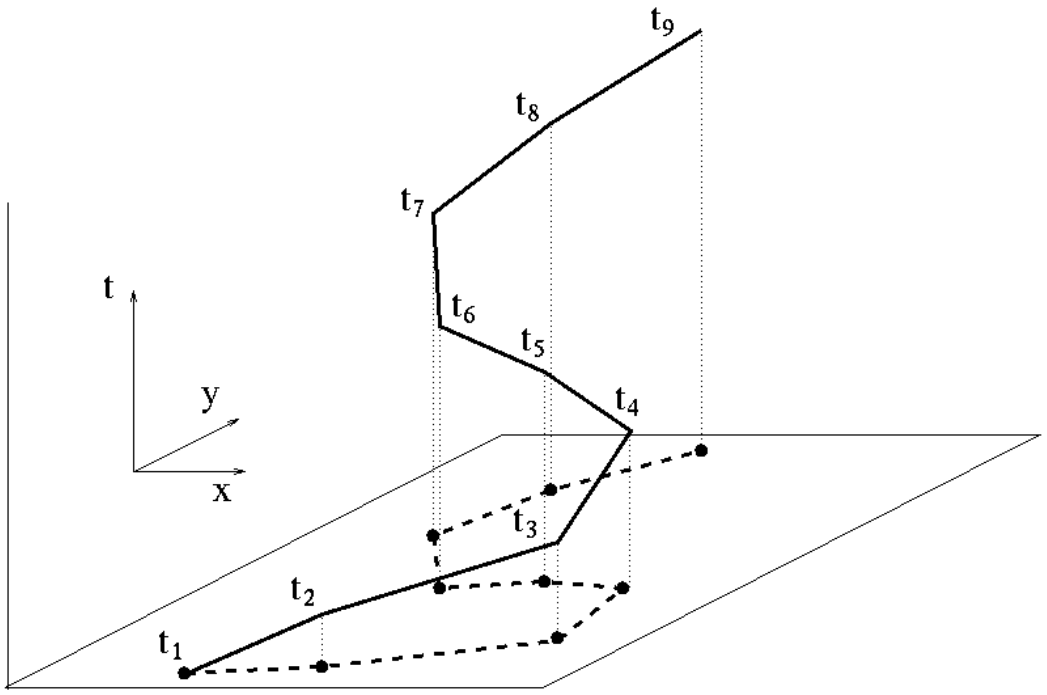
DATA: TRAJECTORIES (1)

› Sample position of object at time points and interpolate between samples



DATA: TRAJECTORIES (2)

- Objects are tracked on a
- Add time along the vertic
- Represent in 3D:
  - 2D spatial
  - 1D temporal



- The resulting 3D polyline consisting of **segments** is the **trajectory** of the moving object

IMPLICATIONS

- Typical: Dataset → **Objects**
- Now: Dataset → **Trajectories** → **Segments**
- By considering time and trajectories,
  we can **derive** further information:
  Speed of object
  Is object entering or leaving?
  Was it here at the same time as another object?
  Where did object go for the entire day?

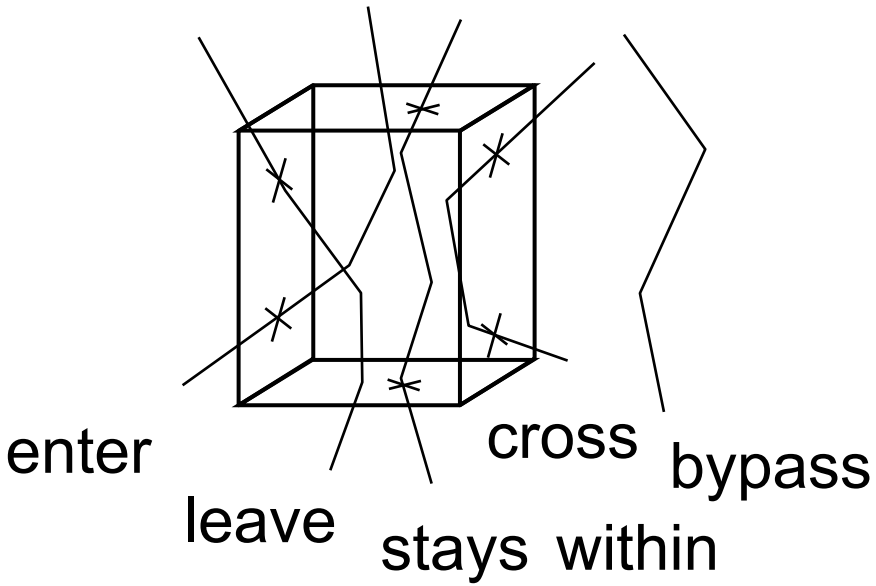
QUERIES

- Coordinate-based queries
  - Point
  - Range
  - Nearest neighbour
- Trajectory-based queries
  - Topological: Enter, leave, cross, bypass
  - (Navigational: Using derived information, e.g. speed, heading…)
- Combined queries

TOPOLOGICAL QUERIES

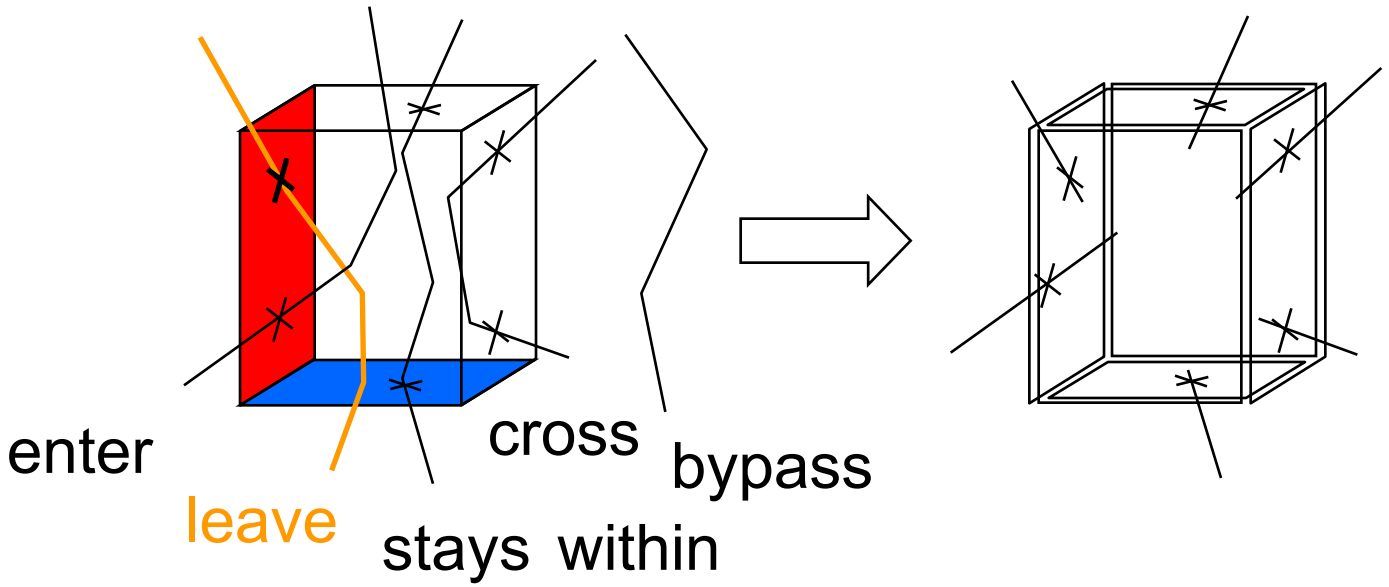
› Trajectories **enter**, **leave**, **cross**, **stay within**, or **bypass** a given spatiotemporal range

› Signature: range × {trajectories} → {trajectories}



TOPOLOGICAL QUERIES

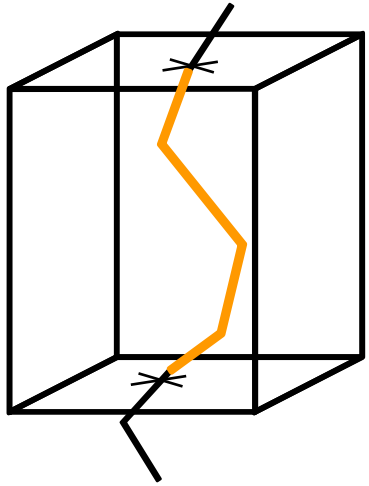
"Which taxis left Tucson between 7 and 8 a.m. today?"



(NAVIGATIONAL QUERIES)

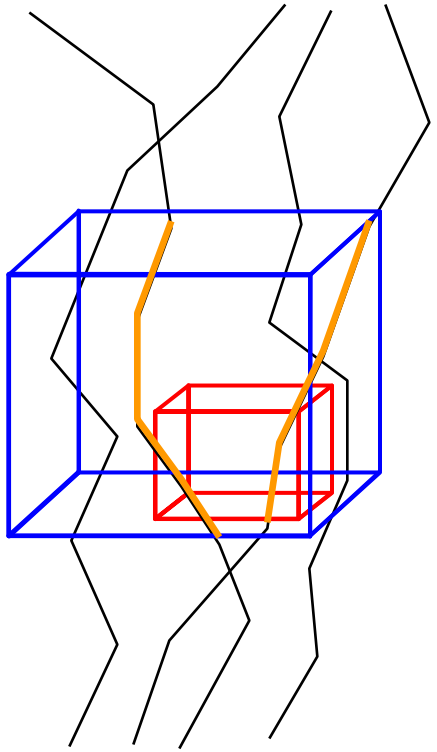
› Considering derived information in a query,
e.g., **speed** (top, average), **heading**, **traveled distance**, **covered area**, etc.

› Signature: range × {trajectories} → int | real | bool



COMBINED QUERIES

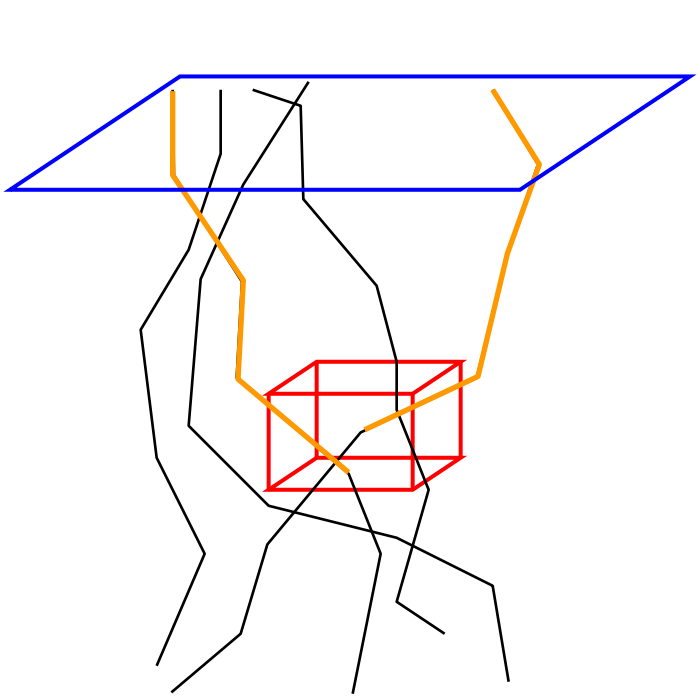
- Dataset → **Trajectories** → **Segments**
- *First*, selecting the *trajectory*, and *subsequently* selecting the *segments*
- Trajectory selection
  - trajectory identifier
  - any type of query

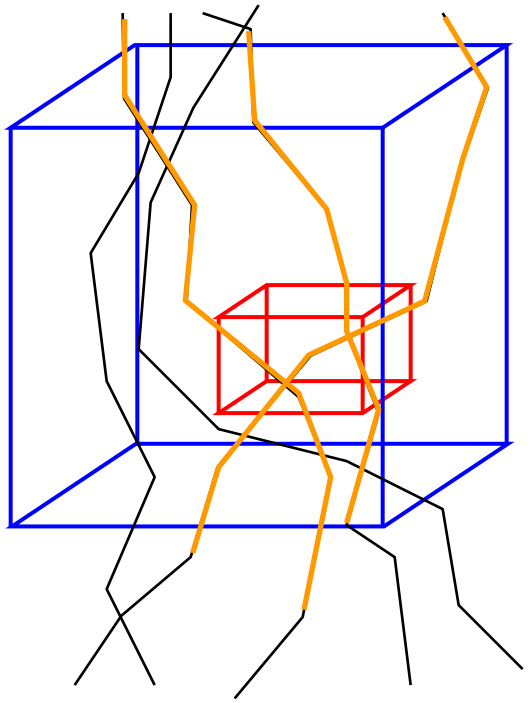


"What were the trajectories of the taxis that were in Tucson between 7 and 8 a.m. today until either noon or leaving Arizona?"

COMBINED QUERIES

"What were the trajectories of the taxis until 12 p.m. after they left Tucson between 7 and 8 a.m. today?"



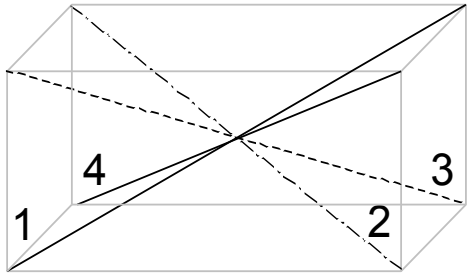


OVERVIEW

- Data: Trajectories
- Queries
- **Data structures:**
  - **R-tree**
  - **Spatio-Temporal R-tree (STR-tree)**
  - Trajectory Bundle tree (TB-tree)
- Query handling
- Performance

DATA STRUCTURES

- We store **segments** of **trajectories**
- R-tree
  - approximating **bounding box**
  - **orientation** of the segment, and
  - **trajectory id**

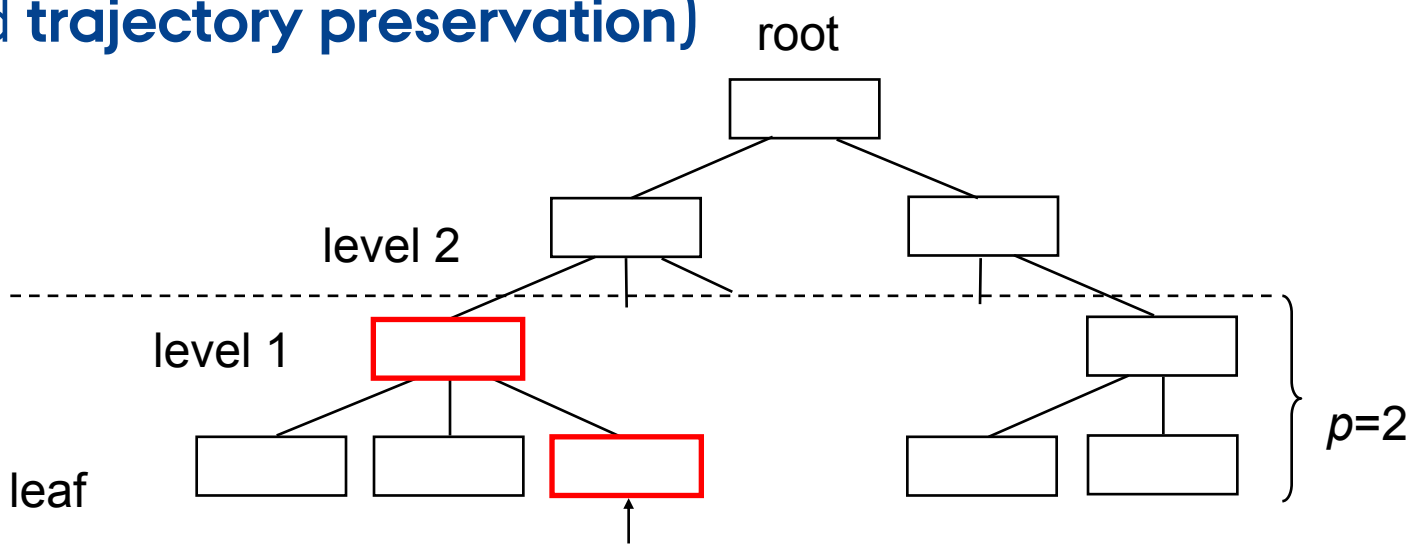


- STR-tree (SpatioTemporal R-tree)
- **spatial discrimination**, i.e., preserve spatial proximity of segments in a leaf node, and **trajectory preservation**, i.e., segments belonging to the same trajectory

- The tree structures are identical – the strategies for inserting and splitting differ

STR-TREE: INSERTION (1)

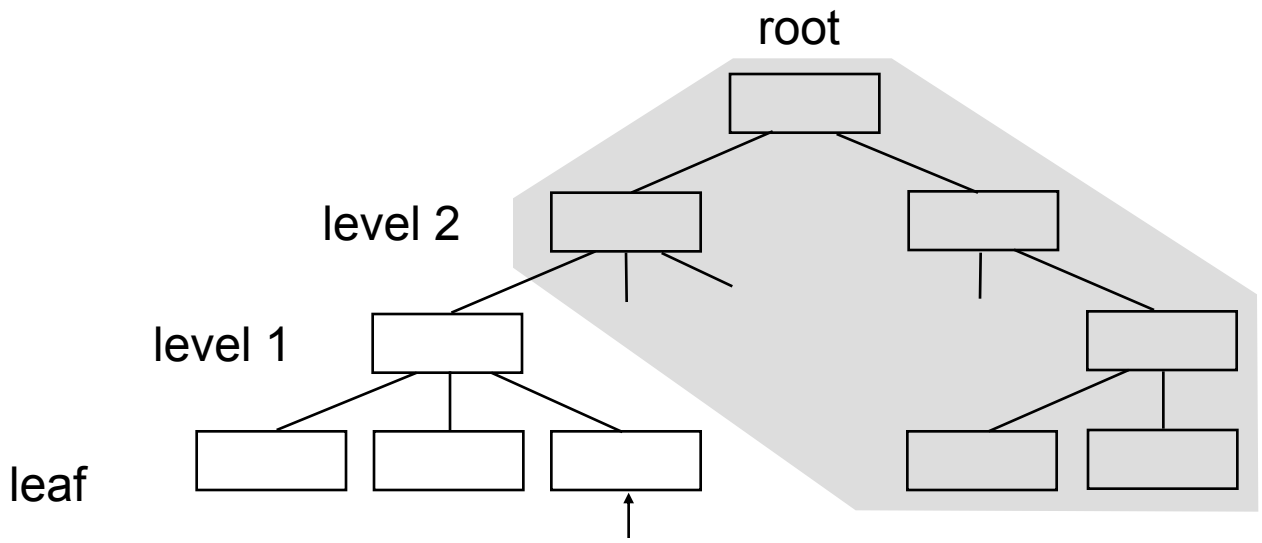
- Insert new segment in leaf node containing its trajectory predecessor
- If node full: Parameter *p* specifies how far back to search for a parent node to split (trade-off between **spatial packing** and **trajectory preservation**)



STR-TREE: INSERTION (2)

› Insert the segment in the leaf node that needs least expansion, like in R-tree

› Exclude the already traversed part of the tree



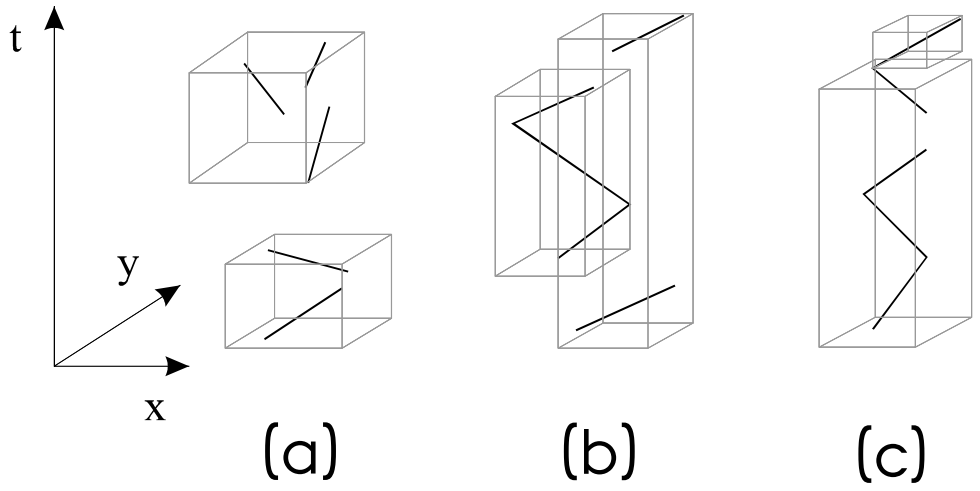
STR-TREE: NODE SPLITTING

› 3 different node types require 3 split strategies

(a) quadratic split – like in R-tree

(b) disconnected segments in new node - *separating trajectories*

(c) most recent segment in new node - *continuing trajectories*



OVERVIEW

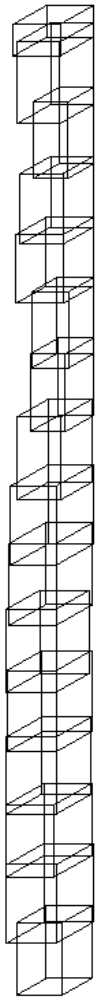
- Data: Trajectories
- Queries
- Data structures:
  - R-tree
  - Spatio-Temporal R-tree (STR-tree)
  - **Trajectory Bundle tree (TB-tree)**
- Query handling
- Performance

TB-TREE

- TB-tree (Trajectory Bundle)
  - **strict trajectory preservation**,

    one leaf node contains segments of only one trajectory
  - **neglecting spatial discrimination**

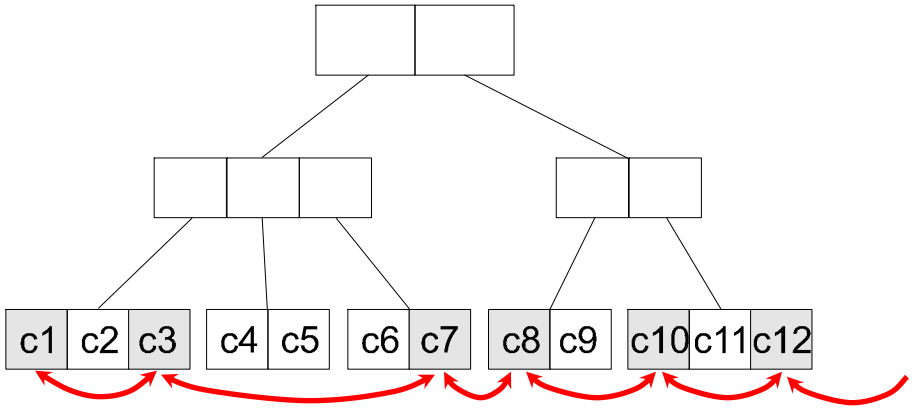
    with respect to the two spatial dimensions

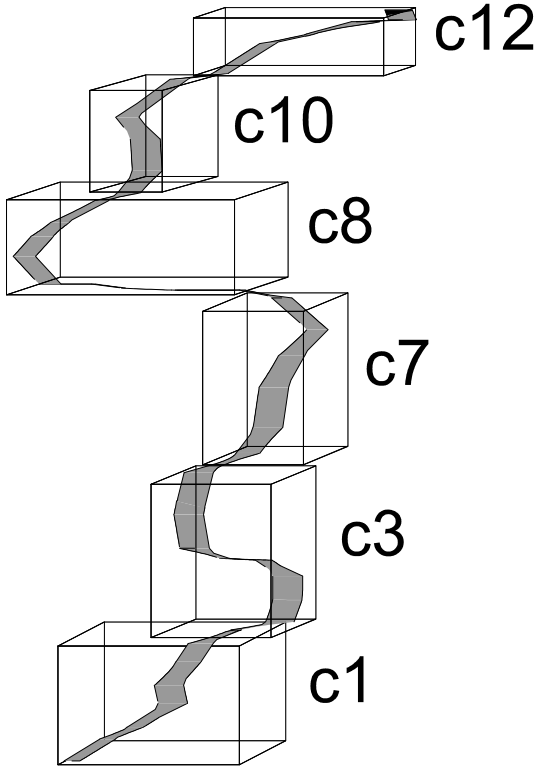


TB-TREE

› A leaf node contains a (partial) trajectory

› Leaf nodes are additionally connected in a linked list to allow easy traversal of trajectories





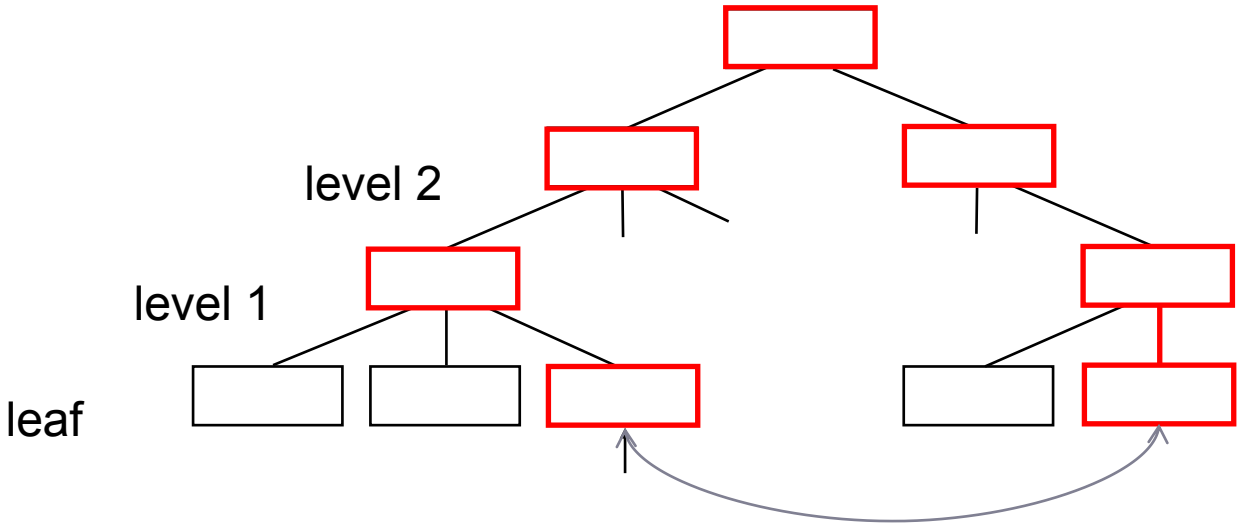
TB-TREE: INSERTION

› Try to insert segment in leaf node containing previous segment

› Strict trajectory preservation; one leaf node, one trajectory

→ filling up nodes and no node splitting necessary

› If full, create new with new segment as only entry

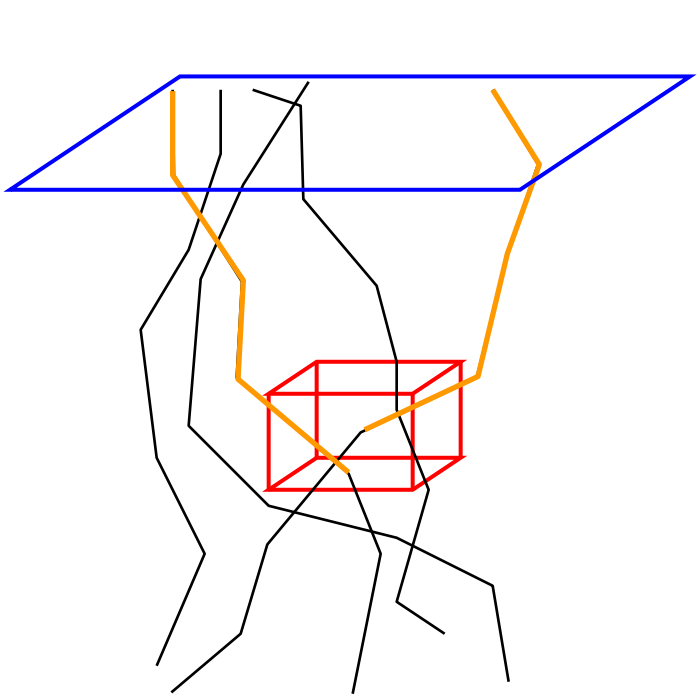


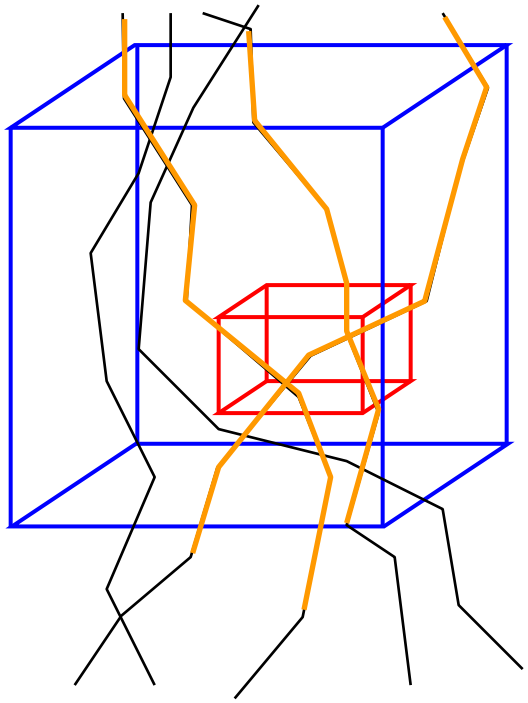
OVERVIEW

- Data: Trajectories
- Queries
- Data structures:
  - R-tree
  - Spatio-Temporal R-tree (STR-tree)
  - Trajectory Bundle tree (TB-tree)
- **Query handling**
- Performance evaluation

COMBINED QUERIES

"What were the trajectories of the taxis until 12 p.m. after they left Tucson between 7 and 8 a.m. today?"

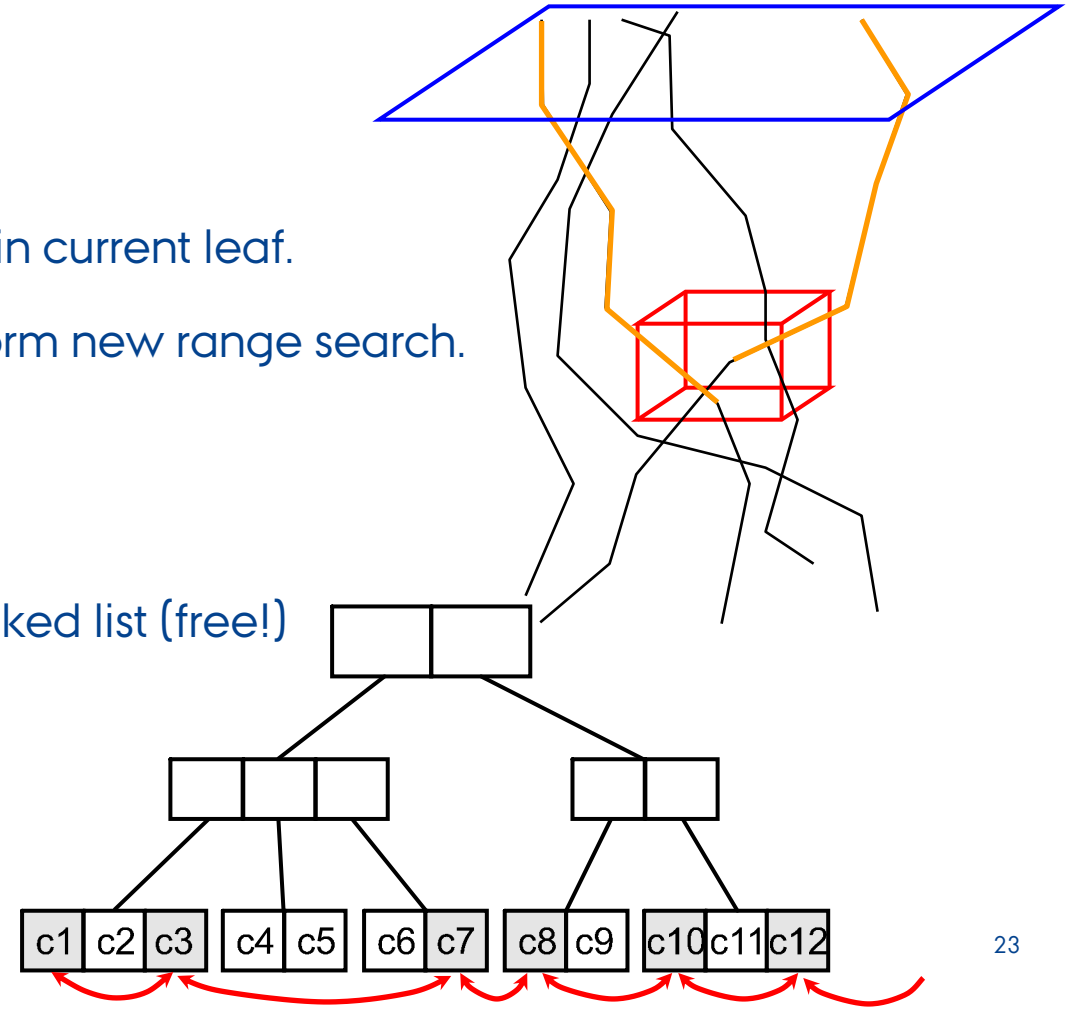




COMBINED QUERY HANDLING

- R- and STR-tree
  - Perform initial range query
  - Expand trajectories by looking in current leaf.

    If no connecting segment: Perform new range search.
- TB-tree
  - Perform initial range query
  - Expand trajectories using the linked list (free!)



OVERVIEW

- Data: Trajectories
- Queries
- Data structures:
  - R-tree
  - Spatio-Temporal R-tree (STR-tree)
  - Trajectory Bundle tree (TB-tree)
- Query handling
- **Performance evaluation**

INDEX SIZE

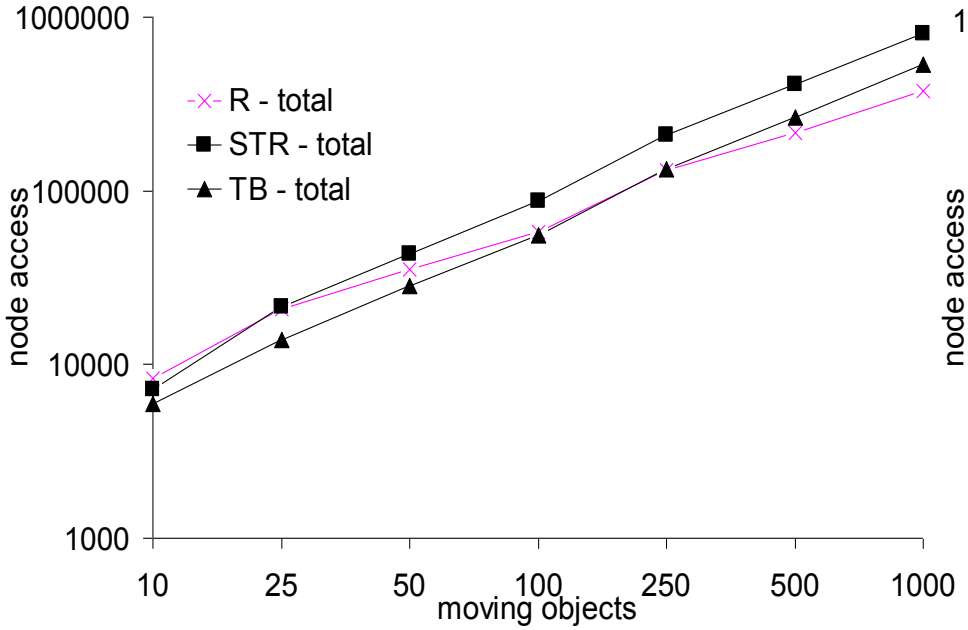
› STR and TB-tree: Nodes are full, because new segments are appended to their trajectories when possible -> small index

› TB-tree saves a little space because we only need to store one trajectory ID pr leaf

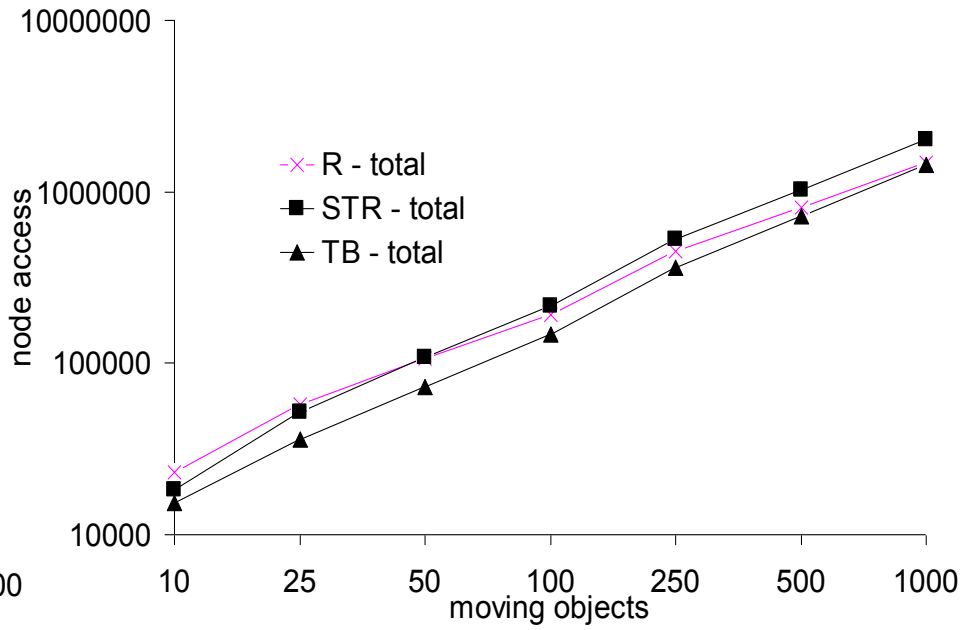
	R-tree	STR-tree	TB-tree
Index Size	~ 95 KB per object	~ 57 KB per object	~ 51 KB per object
Space Utilization	55%-60%	~100%	~100%

RANGE QUERIES

- Increasing number of moving objects
- less emphasis on temporal discrimination, need for spatial discrimination



Query: 10% range in each dimension



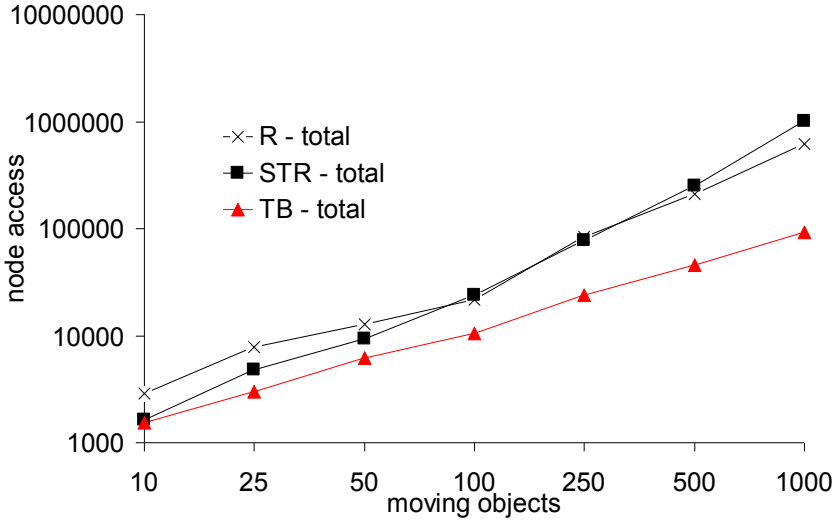
Query: 20% range in each dimension

COMBINED QUERIES

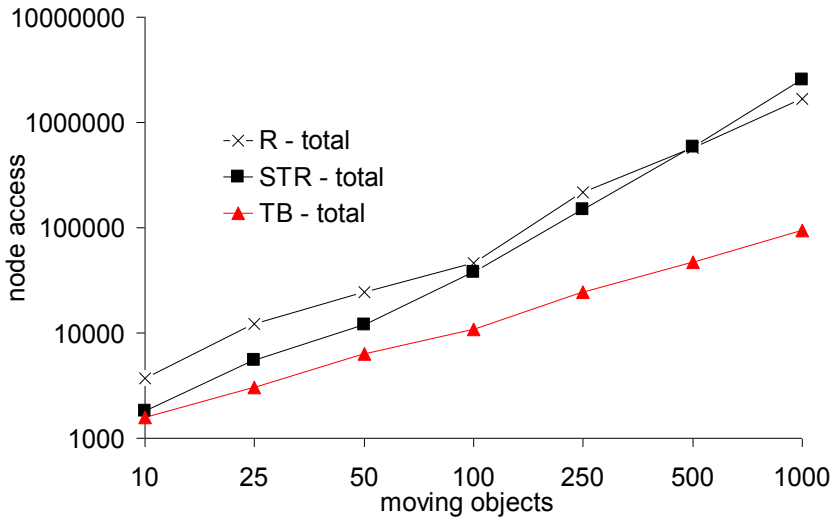
- **TB-tree**: linked list reduces node access once the trajectory is identified
- **STR vs. R**:

  With low number of objects, the more tightly packed STR-tree is better.

  With high number, better spatial discrimination of R-tree wins.



1% inner range, 10% outer



1% inner range, 20% outer

SUMMARY

- Definition of **trajectory data** and **queries**
- Proposal of the **STR-tree** and the **TB-tree**
- Performance
  - the TB-tree performs generally better than the STR-tree
  - the TB-tree is competitive to the R-tree for coordinate-based queries
  - the TB-tree outperforms the other methods for combined queries