Processing Spatio-Temporal Queries in a Streaming Fashion

Rui Zhang

Department of Computer Science
and Software Engineering

University of Melbourne

The 22th Australasian Database Conference (ADC)

Outline

- A look back: Spatial queries and Temporal queries
- Background for processing Streaming Spatio-Temporal queries
- Key techniques
  - Incremental computation and shared computation
  - Safe region
  - Time constraining
- A look forward

A Look Back

- Before 1995:
  - Spatial: static point queries, range/window queries
  - Temporal: version indexes, time interval indexes
- 1995 – 2000:
  - Spatial: nearest neighbor (NN) queries, selectivity estimation/cost models, high-dimensional data
  - Temporal: version indexes, time interval indexes
- 2000 – 2005:
  - Spatial: reverse nearest neighbor (RNN) queries, spatial joins, skyline queries
  - Temporal: time series, similarity queries
  - Spatio-temporal: point, range, and NN queries on moving objects
  - Data streams
- 2005 – 2008:
  - Spatial: trajectories, location selection,
  - Temporal: trajectories
  - **Continuously Moving Queries on static objects**
  - **Continuous Queries on moving objects**
- After 2008 (look forward)

Streaming Spatio-Temporal Queries

- Key Characteristics of Streaming
  - Query is continuous
  - Query answer may change anytime due to continuous change of the query itself or the data
  - Prompt answer is important: highly efficient algorithm
- Changes in Streaming Spatio-Temporal queries
  - Static query, data objects moving
  - Static data objects, query moving
  - Both query and data objects moving
- Key techniques
  - Incremental computation and shared computation
  - Safe region
  - Time constraining

Incremental and shared computation

- Continuous Retrieval of 3D Objects [ICDE'08, VLDB Journal'10]

- Applications
  - Augmented reality
  - A rescue officer can see the structure of a building even if the building is on fire and filled with smoke

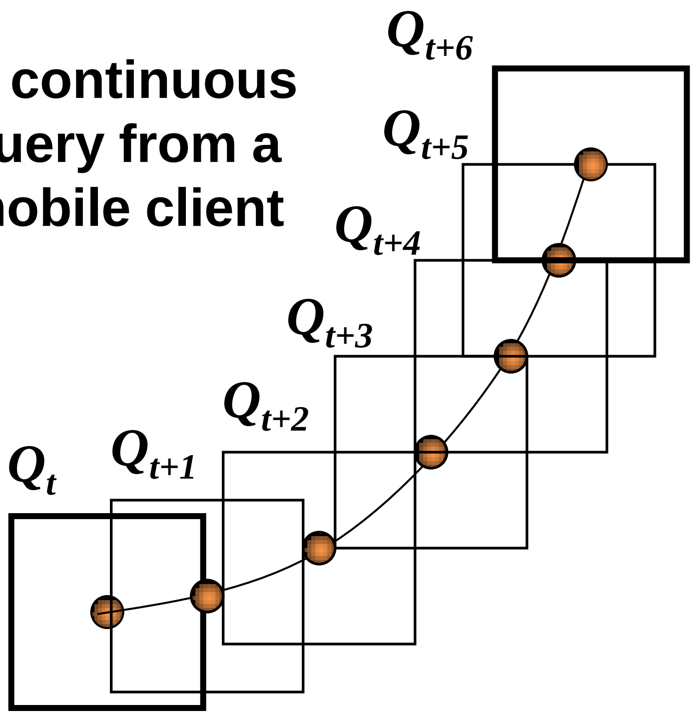
  - A smart phone to see the interior of restaurants
- Continuous window query on static 3D objects

Problem

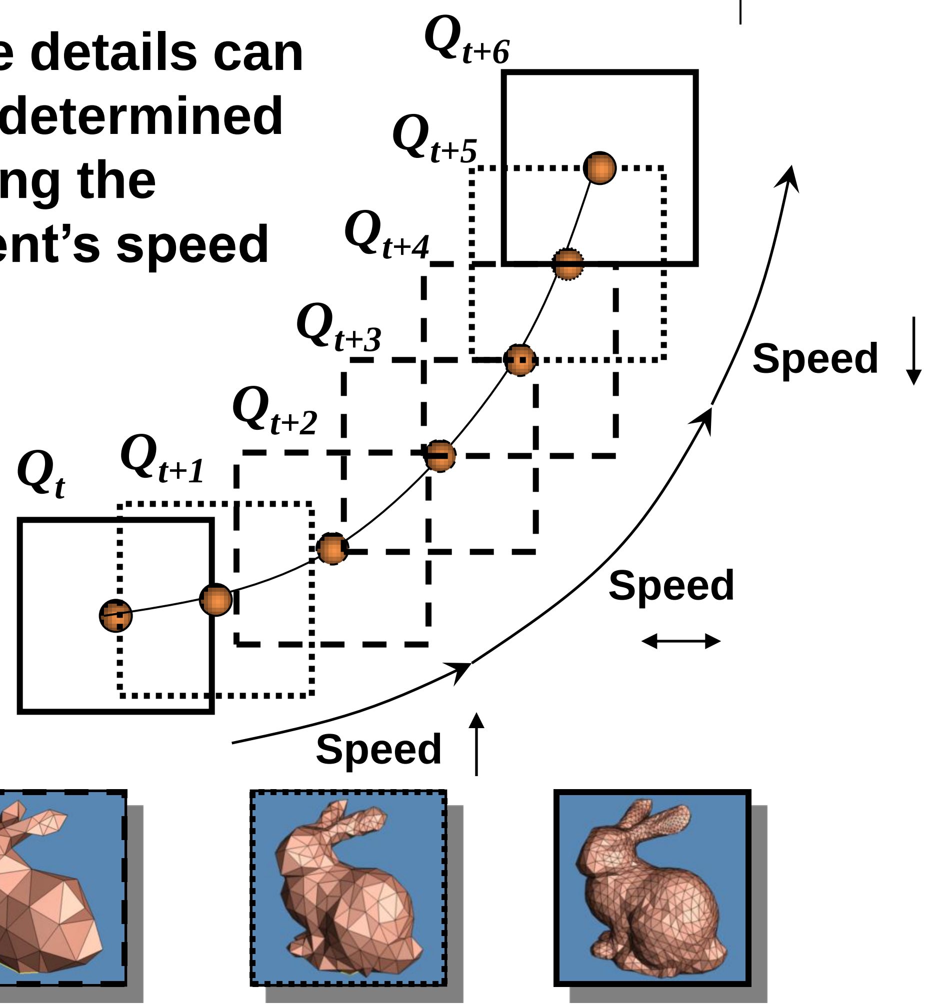
- Continuous retrieval of 3D objects in a window
- Model: client-server
- Bottle neck: bandwidth, especially when the view is moving fast
- To enable incremental/shared computation
  - Need to decompose the query answer into smaller components

Observation

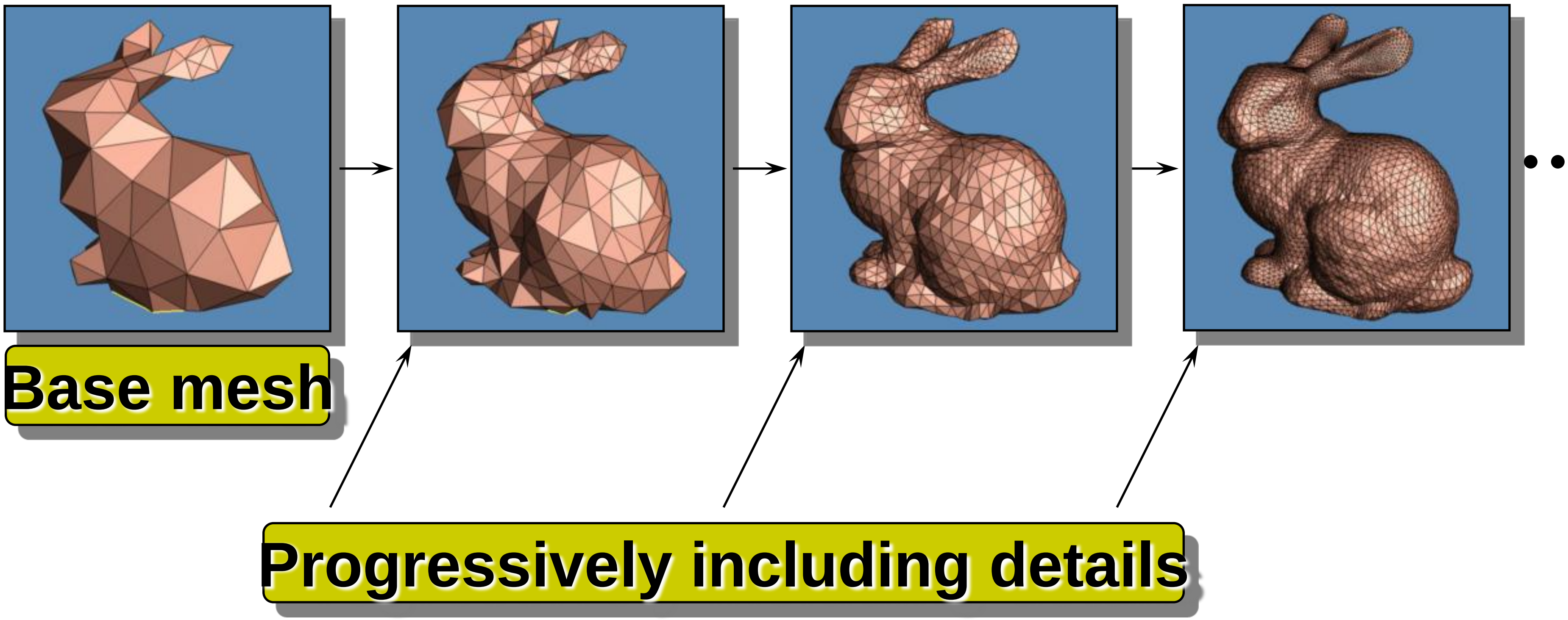
A continuous query from a mobile client



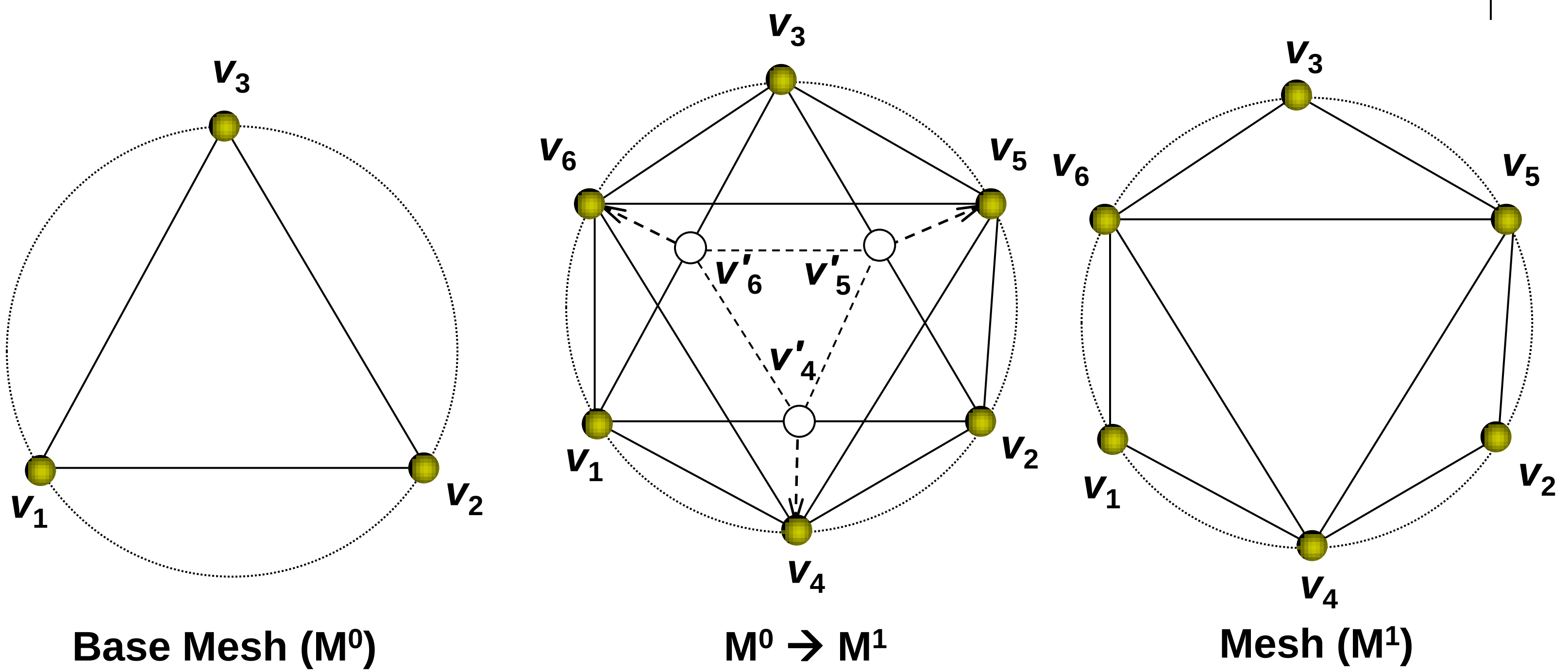
The details can be determined using the client's speed



Multi-resolution Representations

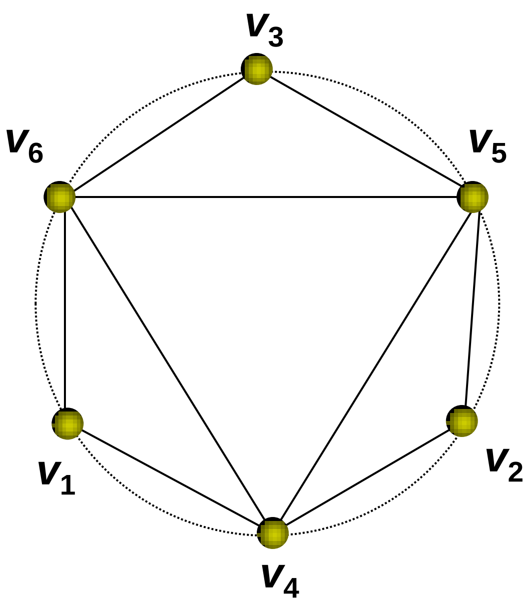


Example Wavelet Decomposition

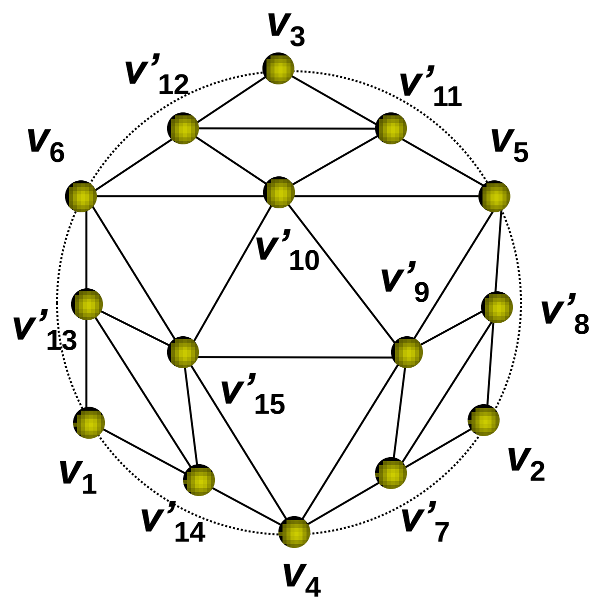


Wavelet coefficient, $d_4 = v_4 - (v_1+v_2)/2 = v_4 - v'_4$

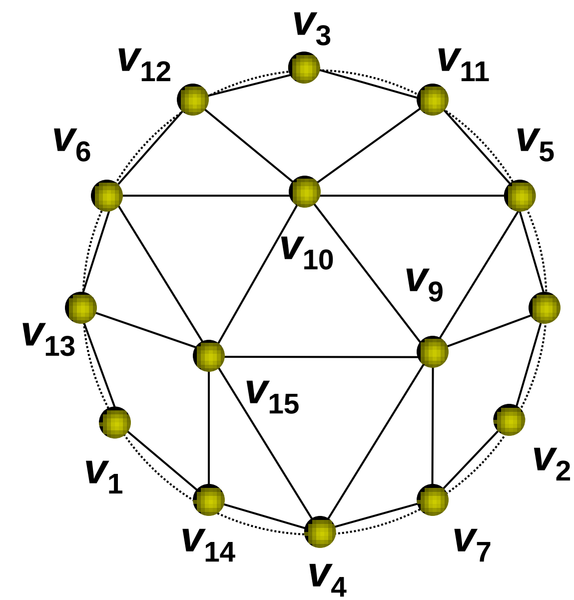
Example Wavelet Decomposition



Mesh (M^1)

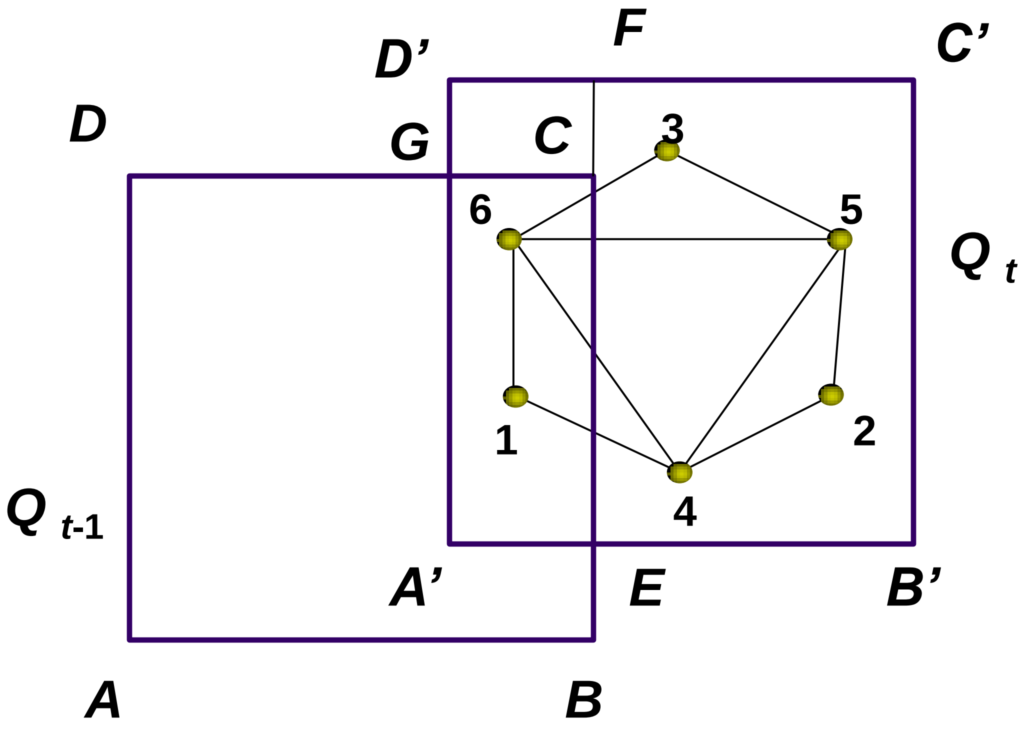


$M^1 \rightarrow M^2$

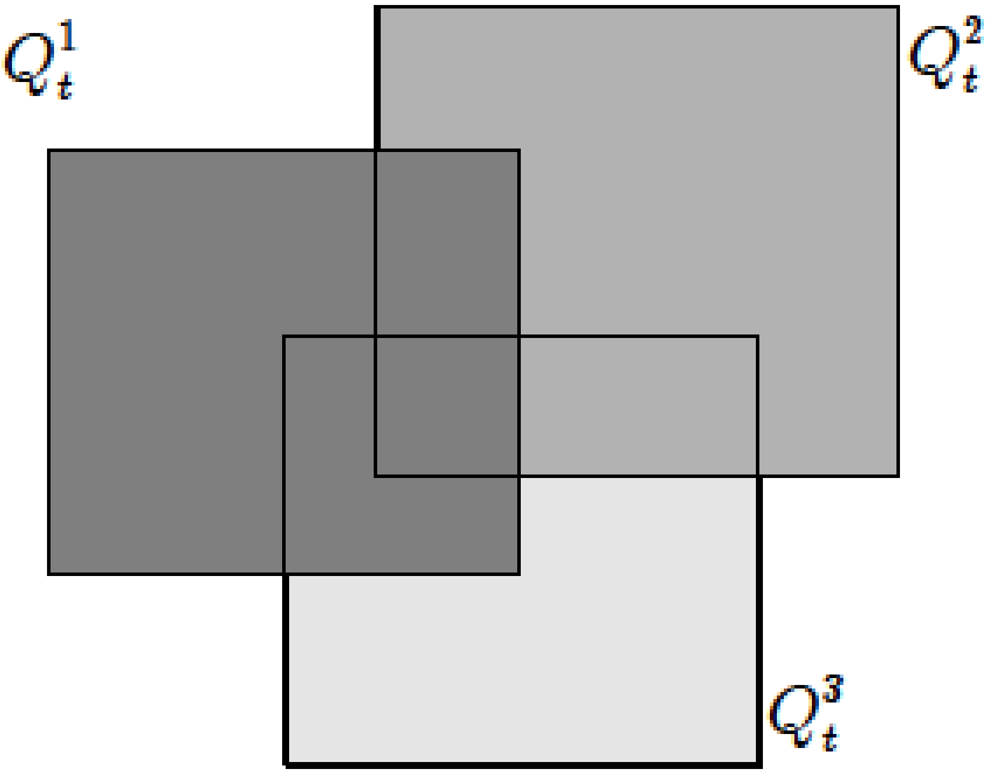


Mesh (M^2)

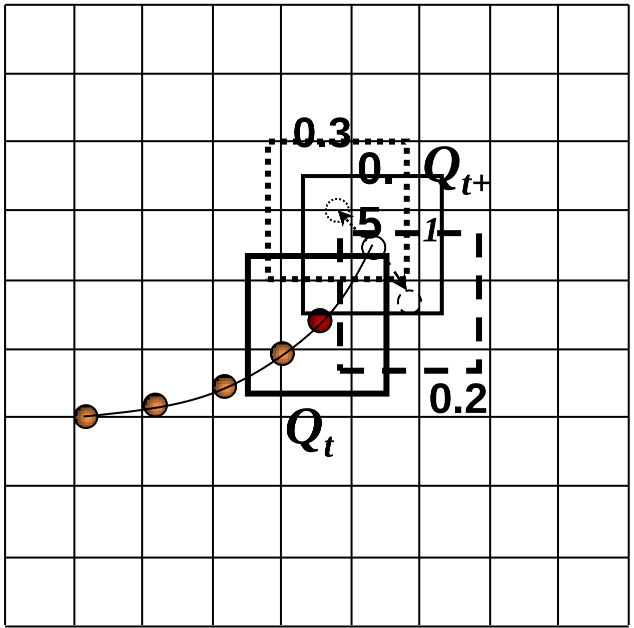
Incremental Retrieval

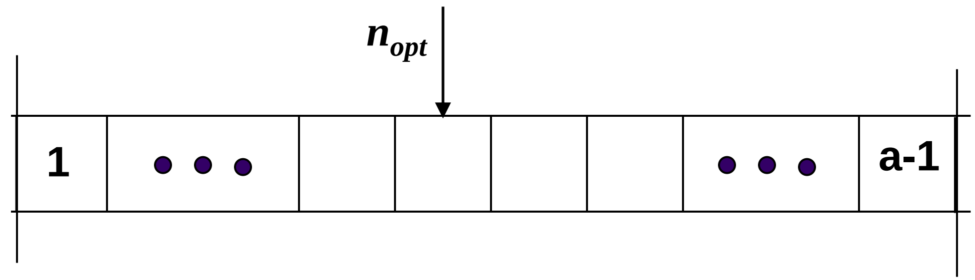


Group Queries



Motion-Aware Buffer Management

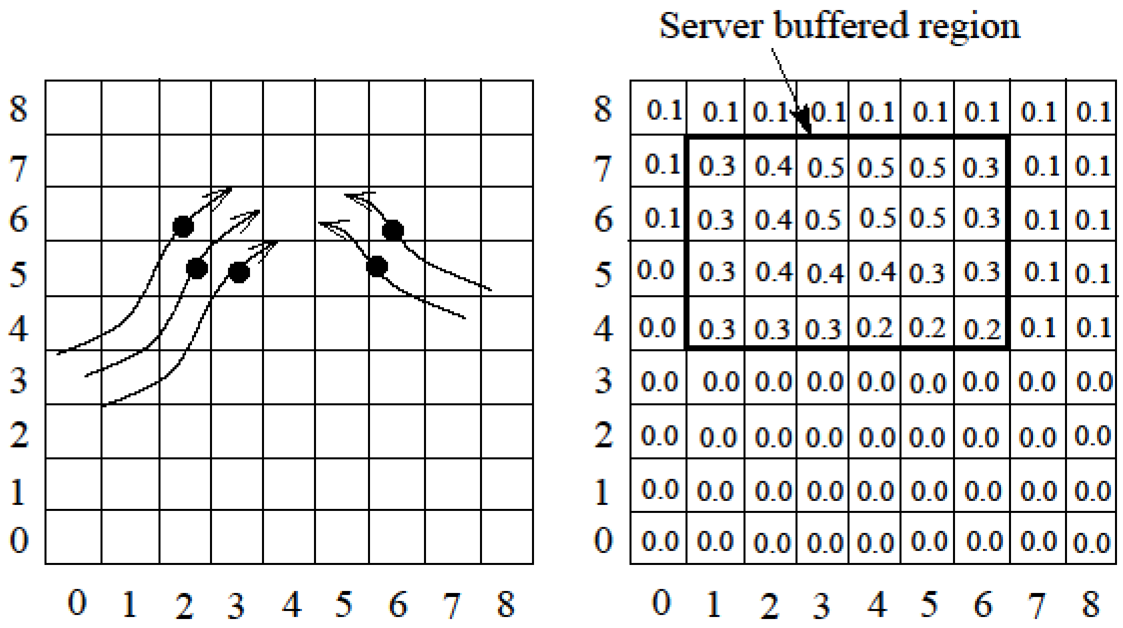




- Buffer: Given probabilities to move in one dimension to two directions

- Generalize one dimension to 2-dimensions

Buffer Management for Group Queries



(a) (b)

- (a) The paths for five different clients
- (b) combined weights of visiting probabilities of different data blocks

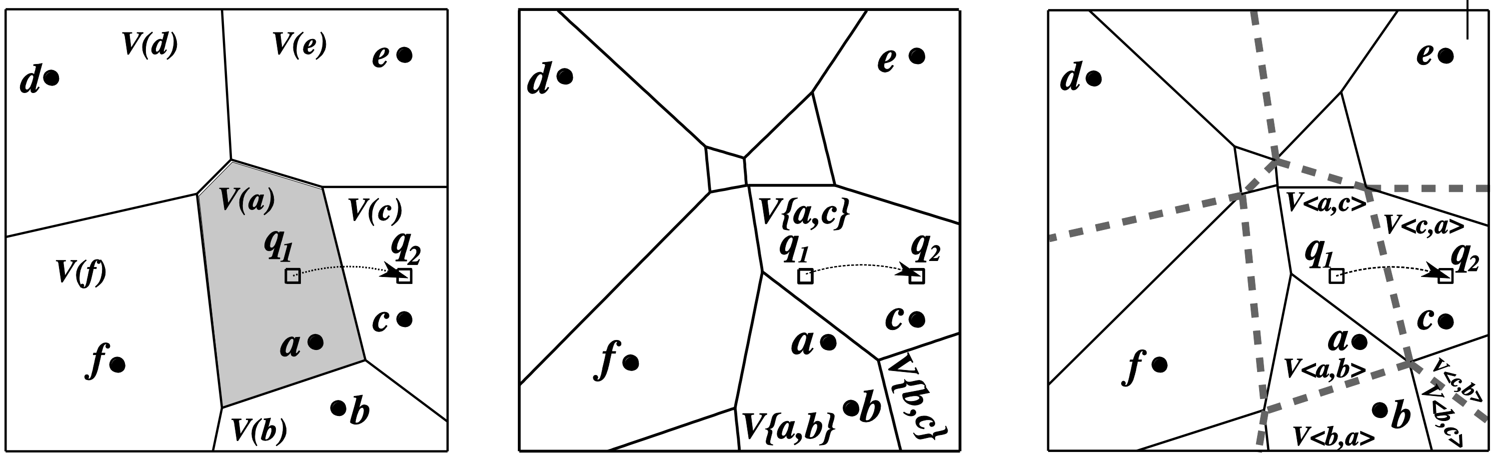
Safe Region

- Continuously returning k NNs for a moving query point [VLDB'08, VLDB Journal'10]

- Applications
  - Continuously reporting the nearest gas station, restaurant, ATM, etc.

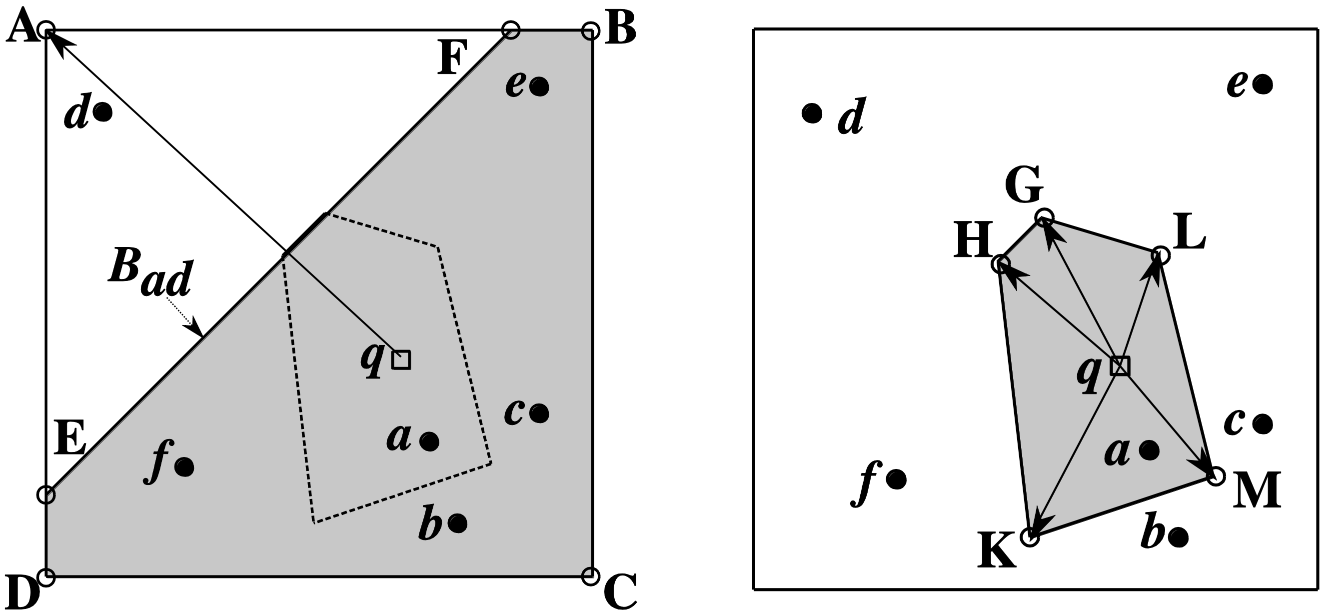
- Continuous kNN query on static 2D points

Traditional Approach: Voronoi Diagram



- Drawbacks:
  - Expensive precomputations (quadratic wrt k)
  - Inefficient update operations
  - No support for dynamically changing k values

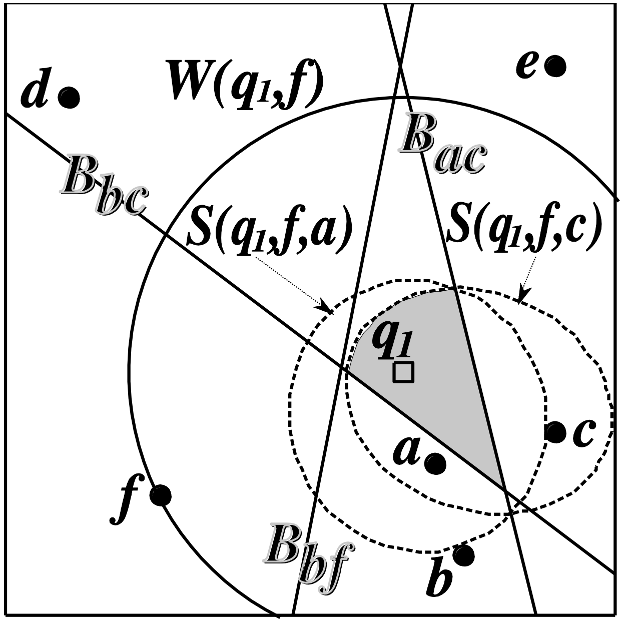
Best Existing Approach



- Computing a Voronoi cell locally

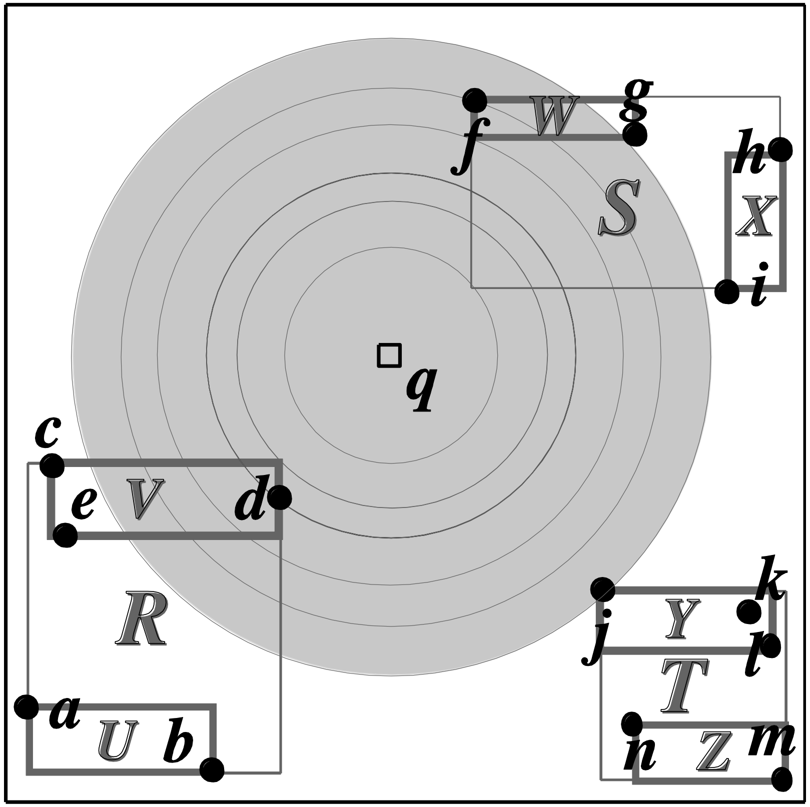
(Much) Better Approach: V*-Diagram

- Goals:
  - Requires no precomputation
  - Supports insertions and deletions of objects
  - Handles dynamically changing k



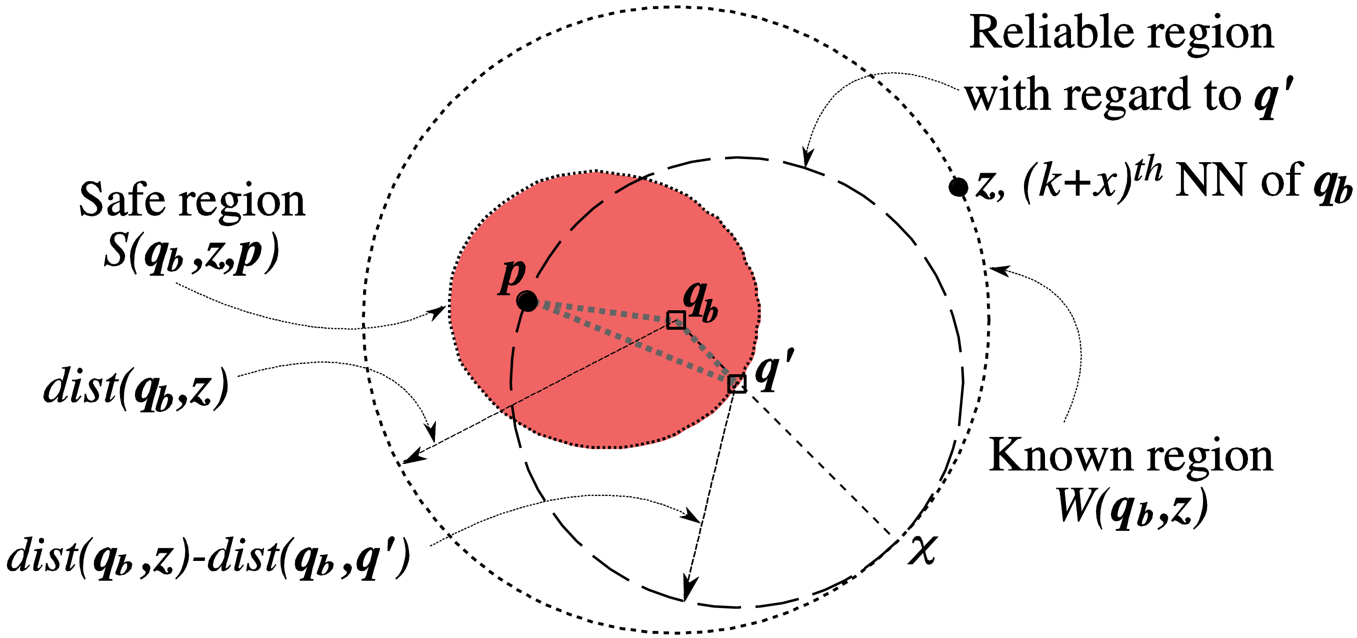
Preliminary

- Incremental kNN algorithm, and known region
  - If the known NNs to ***q*** are {***d***, ***f***, ***j***}, the known region *W*(***q***, ***j***) is {***v*** *: dist(****q***, ***v****) ≤ dist(****q***, ***j****)*}.



V*-Diagram: Safe region wrt a data point

- We retrieve (k + x) objects. In this example, k and x are both 1, so we retrieve $\boldsymbol{p}$ and $\boldsymbol{z}$.



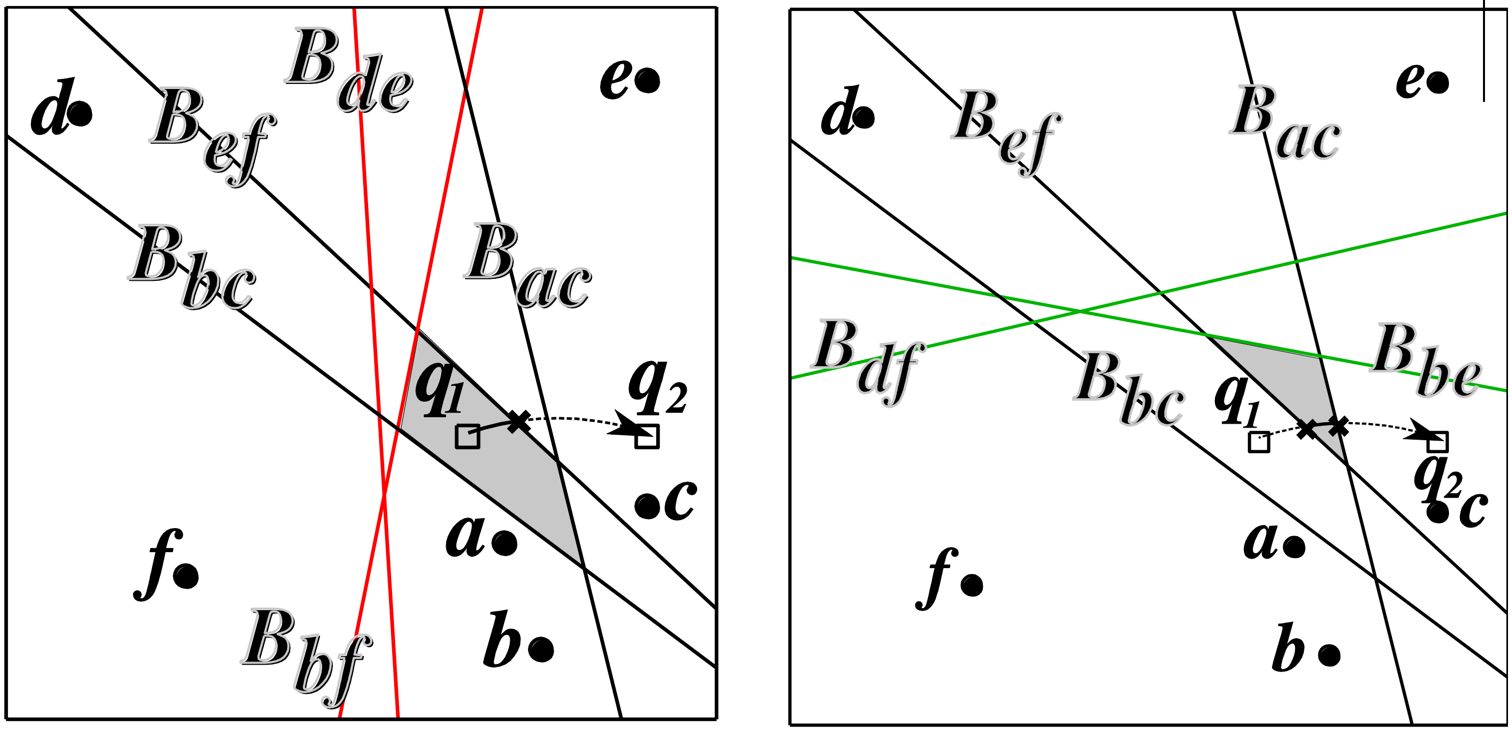
If $\boldsymbol{q'} \in S(\boldsymbol{q_b}, \boldsymbol{z}, \boldsymbol{p})$ then,

$$\forall\ \boldsymbol{p'} \notin W(\boldsymbol{q_b}, \boldsymbol{z}),\ \mathrm{dist}(\boldsymbol{q'}, \boldsymbol{p}) < \mathrm{dist}(\boldsymbol{q'}, \boldsymbol{p'}).$$

$$S(\boldsymbol{q_b}, \boldsymbol{z}, \boldsymbol{p}) = \{\boldsymbol{q'} : \mathrm{dist}(\boldsymbol{p}, \boldsymbol{q'}) \leq \mathrm{dist}(\boldsymbol{q_b}, \boldsymbol{z}) - \mathrm{dist}(\boldsymbol{q_b}, \boldsymbol{q'})\}$$

$$= \{\boldsymbol{q'} : \mathrm{dist}(\boldsymbol{p}, \boldsymbol{q'}) + \mathrm{dist}(\boldsymbol{q_b}, \boldsymbol{q'}) \leq \mathrm{dist}(\boldsymbol{q_b}, \boldsymbol{z})\}$$

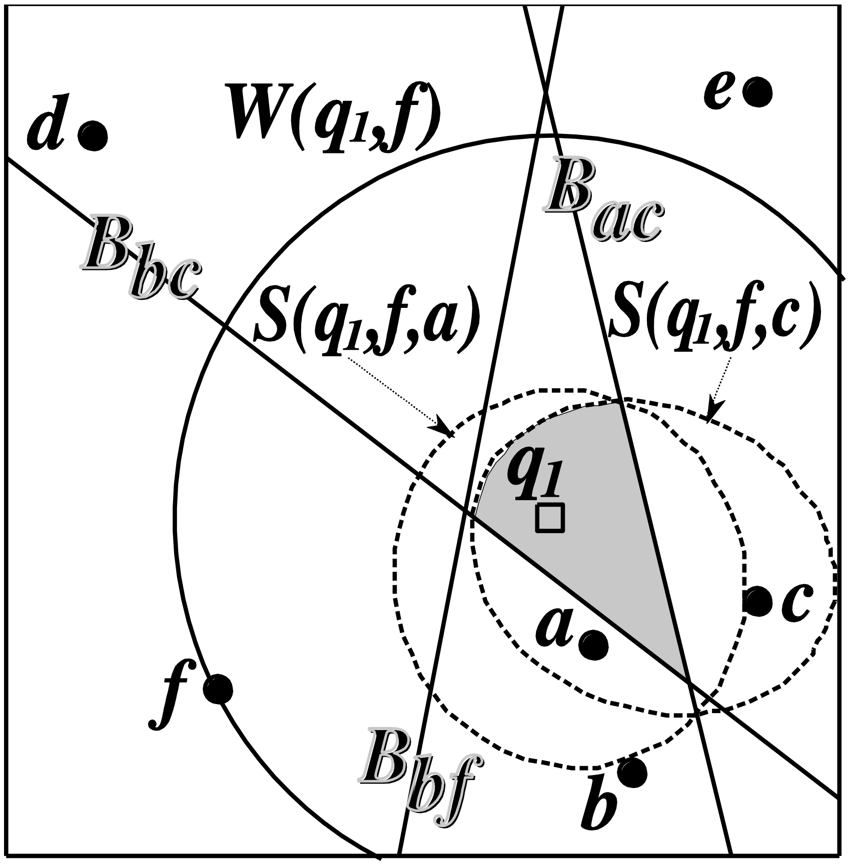
V*-Diagram: The Fixed-rank Region



Incremental rank update

V*-Diagram: Integrated Safe Region

- ISR is an intersection of
  - The safe region wrt k^{th} NN, $S(\boldsymbol{q_b}, \boldsymbol{z}, \boldsymbol{p_k})$;
  - The Fixed Rank Region of the $(k+x)$ NNs of $\boldsymbol{q_b}$.



Example V*-diagram: k =2 , x = 2

Time Constraining

- Continuously returning join result for continuously moving objects [ICDE'08]

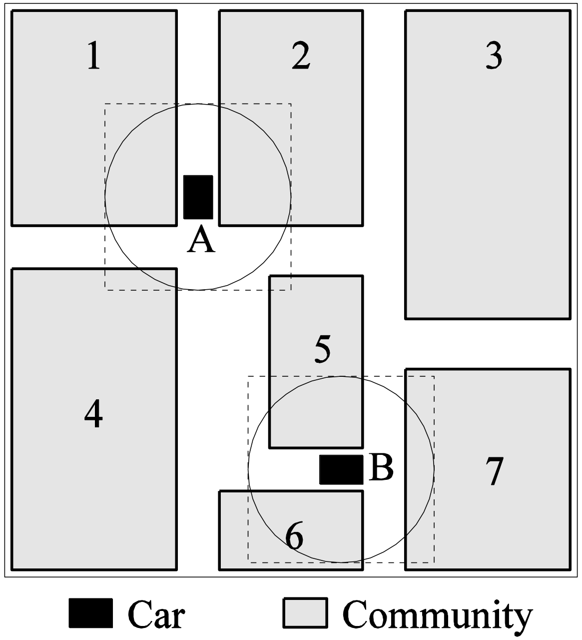
- Applications
  - Monitoring potential attackers in virtual military training programs, large scale multiplayer games

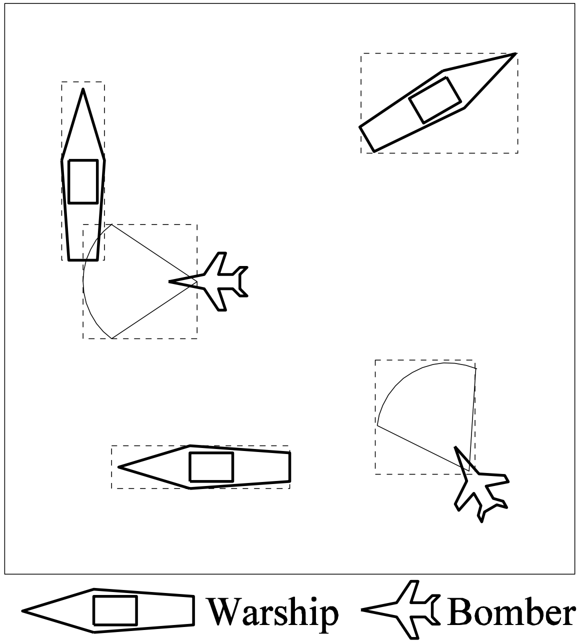
- Continuous intersection query on moving 2D objects

Motivation

- (Traditional) Intersection join
  - Given two sets of spatial objects A and B, find all object pairs ‹i,j›, where i∈A, j ∈B, such that i intersects j.

- Intersection join on moving objects
  - Moving
  - Continuous





Join Algorithms

- Nested loops join
  - Basic
  - Expensive

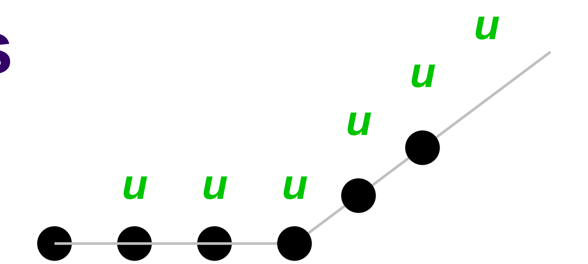
- Block nested loops join
  - Efficient
  - Dependent on buffer size

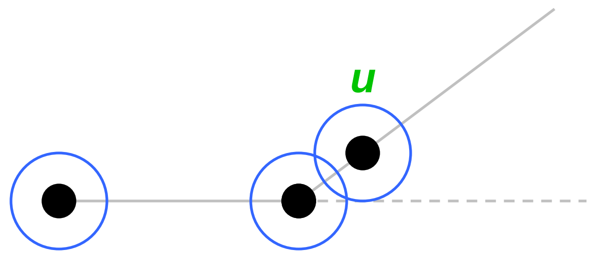
- Index nested loops join
  - Efficient and robust

- Sort-merge join
  - Efficient
  - Difficult for spatial objects

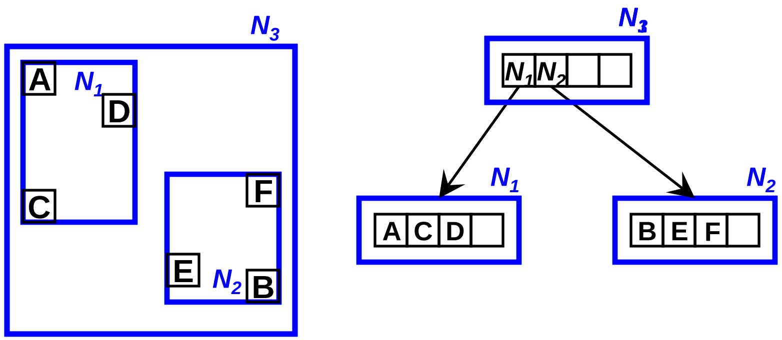
Indexing Moving Objects

- Monitoring moving objects
  - Sampling-based
  - Trajectory-based
    - $\boldsymbol{p} = \boldsymbol{p}(t_{ref}) + \boldsymbol{v}(t - t_{ref})$
    - $\boldsymbol{T_M}$: maximum update interval

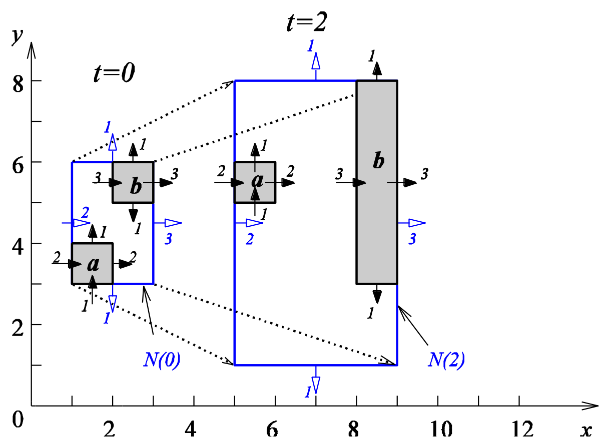




- R-tree [SIGMOD'84]
  - Minimum bounding rectangle (MBR)



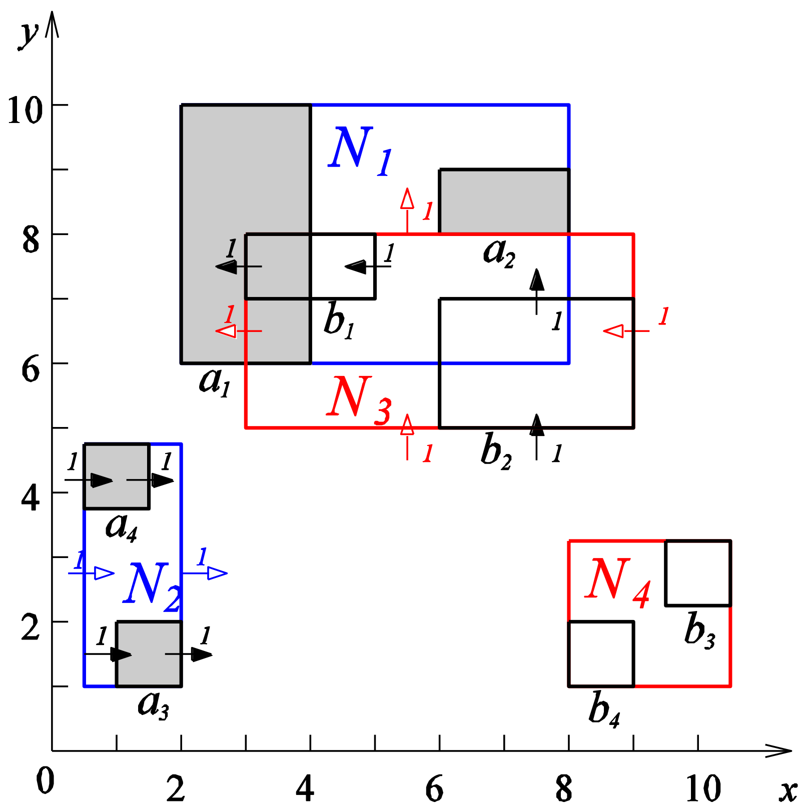
- TPR-tree [SIGMOD'00]
  - Add time parameters to the R-tree

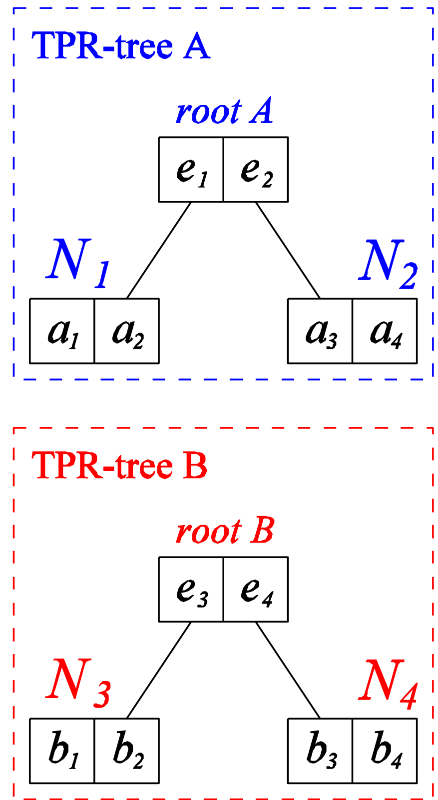


- Other indexes: B^x-tree [VLDB'04], STRIPES [SIGMOD'04]
  - Only for points

Naive Algorithm (NaiveJoin)

- Join nodes from two TPR-trees recursively
  - If intersected, check on children
  - Otherwise, disregard it
  - For an update, compute its join pairs and update the answer



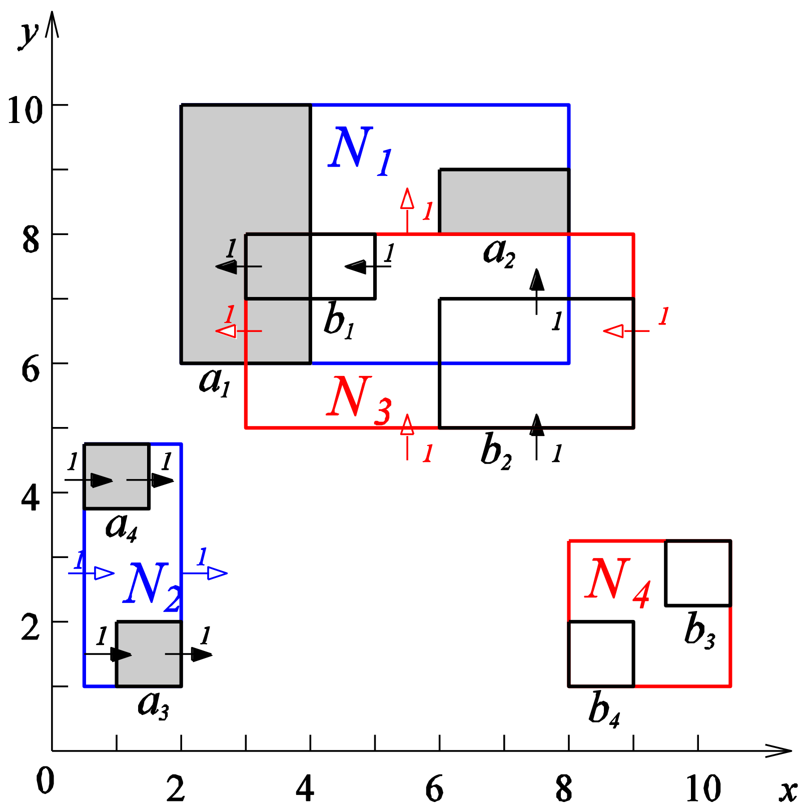


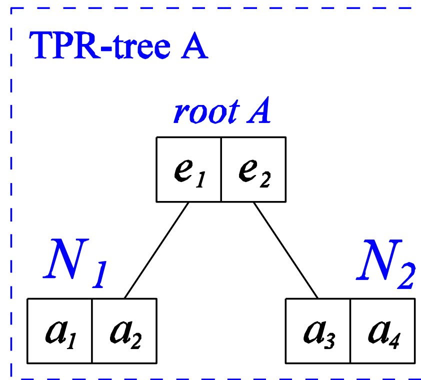
Join result
‹a_1,b_1›, [0,3]
‹a_2,b_2›, [1,4]
‹a_3,b_4›, [6,8]

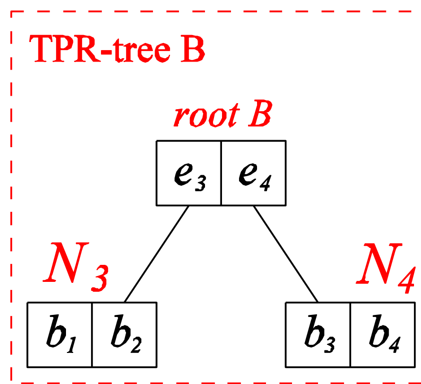
Node access (IO)
roots, N_1, N_2, N_3, N_4
Comparison (CPU)
root A vs root B, N_1 vs N_3, N_2 vs N_4

Extended TP-Join Algorithm (ETP-Join)

- Time Parameterized Join (TP-Join) [SIGMOD'02]
  - Current result ‹a_1,b_1›
  - Expiry time 1
  - Event that causes the change ‹a_2,b_2›







Join result
‹a_1,b_1›, [0,3]
‹a_2,b_2›, [1,4]
‹a_3,b_4›, [6,8]

For the 1st TP-Join

Node access (IO)
roots, N_1, N_3
Comparison (CPU)
root A vs root B, N_1 vs N_3

Summary

- NaiveJoin
  - One tree traversal per update, but expensive traversal

Node access (IO)
roots, N_1, N_2, N_3, N_4
Comparison (CPU)
root A vs root B, N_1 vs N_3, N_2 vs N_4

Too long

- ETP-Join
  - Cheaper traversal, but too frequent traversals

For the 1st TP-Join

Node access (IO)
roots, N_1, N_3
Comparison (CPU)
root A vs root B, N_1 vs N_3

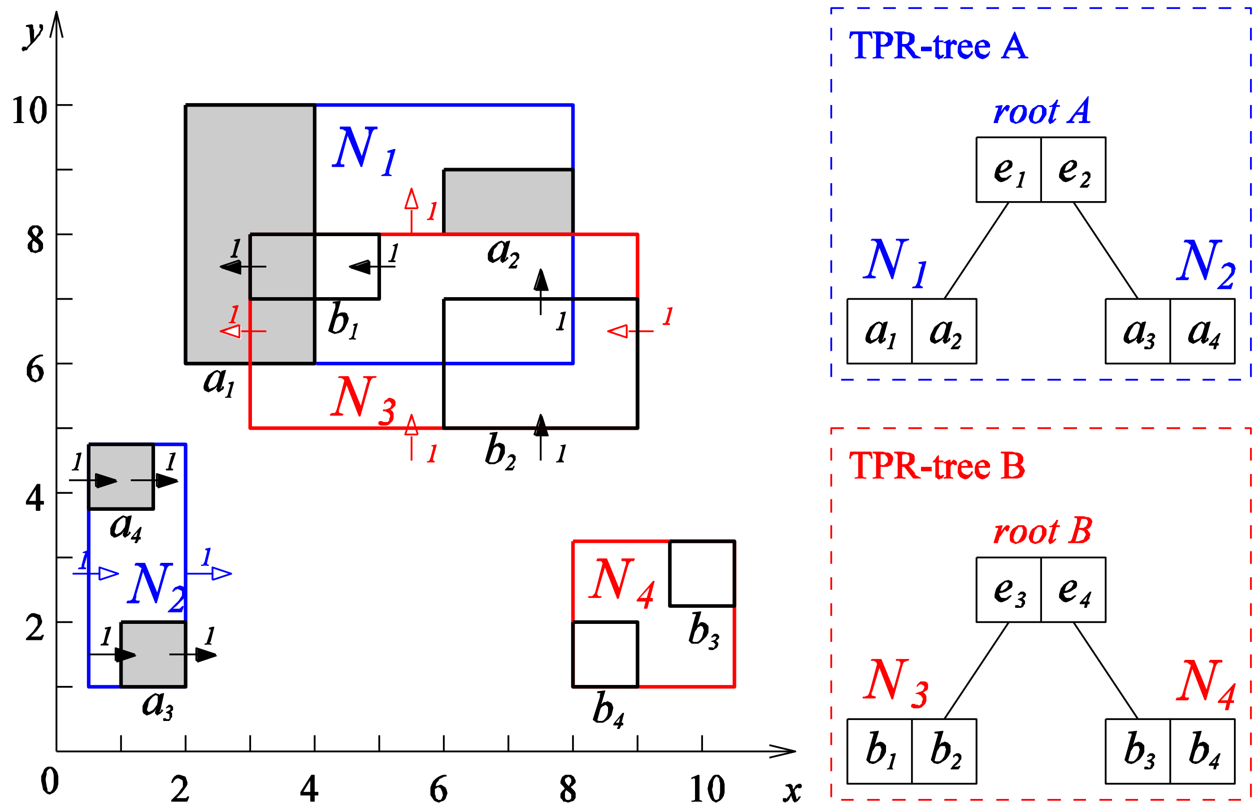
Too short

Key Problem

- Find a **good** time range for computing the join pairs
- Observation
  - Consider object a and b
  - Let the next update time for them be t_a and t_b
  - Perfect time range for computing their join result is $[t_c, \mathit{min}(t_a,t_b)]$
- How do we know t_a or t_b?
  - T_M gives a bound for them
  - Time range is cut from $[t_c, \infty]$ to $[t_c, t_c+T_M]$
- Is this correct for all objects?
  - Yes. Proof in technical report: http://www.cs.mu.oz.au/~rui/publication/TR_mj.pdf

Time Constrained Processing (TC-Join)

- NaiveJoin with constrained processing time range $[t_c, t_c+T_M]$

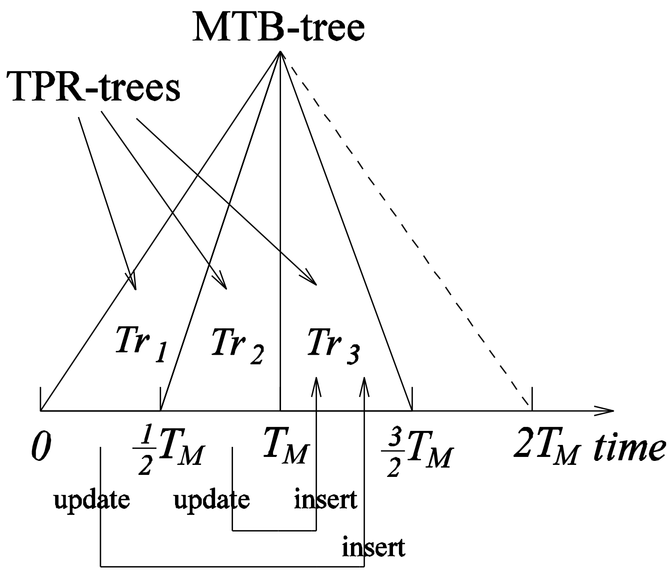


Join result
‹a_1,b_1›, [0,3]
‹a_2,b_2›, [1,4]
‹a_3,b_4›, [6,8]

Node access (IO)
roots, N_1, N_3
Comparison (CPU)
root A vs root B, N_1 vs N_3

Further Optimization (MTB-Join)

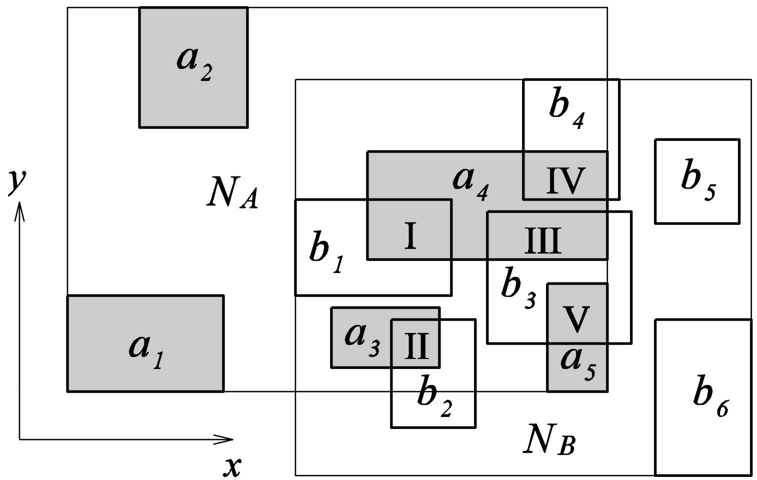
- Many objects will not update at the time bound
- Put objects in time buckets
  - Each time bucket has an associated TPR-tree
  - An object is inserted into the tree whose time bucket contains the object's latest update time



t_c is in $[T_M, 3/2T_M]$

Improvement on the Basic Join Algorithm

- Plane Sweep
  - Sorting based on the lower left corner in dimension x
  - Two sequences: $S_a = \langle a_3, a_4, a_5 \rangle$; $S_b = \langle b_1, b_2, b_3, b_4 \rangle$



- Two essential components for PS
  - Lower bound
  - Upper bound

Reflect on the Three Techniques

- Incremental computation and shared computation
- Safe region
- Time constraining
- Can we use them in other problems?

Chronicle (not complete or thorough)

- Notation: S:static M:moving Q:query D:data
  - Glenn S. Iwerks, Hanan Samet, Kenneth P. Smith: Maintenance of Spatial Semijoin Queries on Moving Points. [VLDB’04]
    MQMD, time constraining
  - Hyung-Ju Cho, Chin-Wan Chung: An Efficient and Scalable Approach to CNN Queries in a Road Network [VLDB’05]
    MQSD, safe region, precomputation (shared computation)
  - Mohamed F. Mokbel, Xiaopeng Xiong, Walid G. Aref: SINA: Scalable Incremental Processing of Continuous Queries in Spatio-temporal Databases. [SIGMOD’04]
    MQMD, incremental/shared computation
  - Haibo Hu, Jianliang Xu, Dik Lun Lee: A Generic Framework for Monitoring Continuous Spatial Queries over Moving Objects. [SIGMOD’05a]
    SQMD, safe region
  - Kyriakos Mouratidis, Marios Hadjieleftheriou, Dimitris Papadias: Conceptual Partitioning: An Efficient Method for Continuous Nearest Neighbor Monitoring. [SIGMOD’05b]
    MQMD, safe region

Chronicle (continued)

- Notation: S:static M:moving Q:query D:data
  - Mohammed Eunus Ali, Rui Zhang, Egemen Tanin, Lars Kulik. A Motion-Aware Approach to Continuous Retrieval of 3D Objects. [ICDE’08b]
    MQSD, incremental/shared computation
  - Rui Zhang, Dan Lin, Kotagiri Ramamohanarao, Elisa Bertino. Continuous Intersection Joins Over Moving Objects. [ICDE’08a]
    MD, time constraining
  - Sarana Nutanong, Rui Zhang, Egemen Tanin, Lars Kulik. The V*-Diagram: A Query Dependent Approach to Moving KNN Queries. [VLDB’08b]
    MQSD, safe region
  - Zaiben Chen, Heng Tao Shen, Xiaofang Zhou, Jeffrey Xu Yu: Monitoring path nearest neighbor in road networks. [SIGMOD’09]
    MQSD, incremental computation
  - Muhammad Aamir Cheema, Xuemin Lin, Ying Zhang, Wei Wang, Wenjie Zhang. Lazy Updates: An Efficient Technique to Continuously Monitoring Reverse kNN Queries [VLDB’09]
    MQMD, safe region

A Look Back

- Before 1995:
  - Spatial: static point queries, range/window queries
  - Temporal: version indexes, time interval indexes
- 1995 – 2000:
  - Spatial: nearest neighbor (NN) queries, selectivity estimation/cost models, high-dimensional data
  - Temporal: version indexes, time interval indexes
- 2000 – 2005:
  - Spatial: reverse nearest neighbor (RNN) queries, spatial joins, skyline queries
  - Temporal: time series, similarity queries
  - Spatio-temporal: point, range, and NN queries on moving objects
  - Data streams
- 2005 – 2008:
  - Spatial: trajectories, location selection,
  - Temporal: trajectories
  - **Continuously Moving Queries on static objects**
  - **Continuous Queries on moving objects**
- After 2008 (look forward)

Look Forward: Trend in the Last Few Years

- Queries on continuous queries on moving objects
  - Predictive range and knn queries [InfSys'10]
  - Continuous retrieval of 3D objects [ICDE'08b, VLDBJ'10b]
  - Continuous intersection join [ICDE'08a, VLDBJ'12]
  - Continuous knn join [GeoInformatica'10]
  - (Continuous) Moving knn queries [VLDB'08b, VLDBJ'10a]
  - Other types of incremental queries [TKDE'10]
- Handling very large and streaming temporal databases
  - Transaction time indexing with version compression [VLDB'08a]
  - The HV-tree: a Memory Hierarchy Aware Version Index [VLDB'10a]
  - Mining Distribution Change in Stock Order Streams.[ICDE'09]
- Exploring road network
  - Scalable network distance browsing in spatial databases. [SIGMOD'08]

Look Forward: Trend in the Last Few Years

- Mining locations and trajectories
  - Trajectory clustering: a partition-and-group framework. [SIGMOD'07]
  - On efficiently searching trajectories and archival data for historical similarities [VLDB'08c]
  - Fast approximate correlation for massive time-series data [SIGMOD'10]
  - Swarm: Mining Relaxed Temporal Moving Object Clusters. [VLDB'10b]
  - Mining Significant Semantic Locations From GPS Data [VLDB'10c]

References

- [VLDB'04] Glenn S. Iwerks, Hanan Samet, Kenneth P. Smith: Maintenance of Spatial Semijoin Queries on Moving Points. *International Conference on Very Large Data Bases (VLDB)* 2004.
- [VLDB'05] Hyung-Ju Cho, Chin-Wan Chung: An Efficient and Scalable Approach to CNN Queries in a Road Network. *International Conference on Very Large Data Bases (VLDB)* 2005.
- [SIGMOD'04] Mohamed F. Mokbel, Xiaopeng Xiong, Walid G. Aref: SINA: Scalable Incremental Processing of Continuous Queries in Spatio-temporal Databases. *ACM SIGMOD International Conference on Management of Data (SIGMOD)* 2004.
- [SIGMOD'05a] Haibo Hu, Jianliang Xu, Dik Lun Lee: A Generic Framework for Monitoring Continuous Spatial Queries over Moving Objects. *ACM SIGMOD International Conference on Management of Data (SIGMOD)* 2005.
- [SIGMOD'05b] Kyriakos Mouratidis, Marios Hadjieleftheriou, Dimitris Papadias: Conceptual Partitioning: An Efficient Method for Continuous Nearest Neighbor Monitoring. *ACM SIGMOD International Conference on Management of Data (SIGMOD)* 2005.
- [SIGMOD'07] Jae-Gil Lee, Jiawei Han, Kyu-Young Whang: Trajectory clustering: a partition-and-group framework. *ACM SIGMOD International Conference on Management of Data (SIGMOD)* 2007.
- [SIGMOD'08] Hanan Samet, Jagan Sankaranarayanan, Houman Alborzi: Scalable network distance browsing in spatial databases. *ACM SIGMOD International Conference on Management of Data (SIGMOD)* 2008.
- [ICDE'08a] Rui Zhang, Dan Lin, Kotagiri Ramamohanarao, Elisa Bertino. Continuous Intersection Joins over Moving Objects. *Proceedings of the 24th International Conference on Data Engineering (ICDE)* 2008.
- [ICDE'08b] Mohammed Eunus Ali, Rui Zhang, Egemen Tanin, Lars Kulik. A Motion-Aware Approach to Continuous Retrieval of 3D Objects. *Proceedings of the 24th International Conference on Data Engineering (ICDE)* 2008.
- [VLDB'08a] David Lomet, Mingsheng Hong, Rimma Nehme, Rui Zhang: Transaction Time Indexing with Version Compression. *Proceedings of the VLDB Endowment (PVLDB)*, 1(1), 870-881, 2008.
- [VLDB'08b] Sarana Nutanong, Rui Zhang, Egemen Tanin, Lars Kulik: The V*-Diagram: A Query Dependent Approach to Moving KNN Queries. *Proceedings of the VLDB Endowment (PVLDB)*, 1(1), 1095-1106, 2008.
- [VLDB'08c] Reza Sherkat, Davood Rafiei: On efficiently searching trajectories and archival data for historical similarities. *Proceedings of the VLDB Endowment (PVLDB)*, 1(1), 896-908, 2008.
- [SIGMOD'09] Zaiben Chen, Heng Tao Shen, Xiaofang Zhou, Jeffrey Xu Yu: Monitoring path nearest neighbor in road networks. *ACM SIGMOD International Conference on Management of Data (SIGMOD)* 2009.

References

- [VLDB’09] Muhammad Aamir Cheema, Xuemin Lin, Ying Zhang, Wei Wang, Wenjie Zhang. Lazy Updates: An Efficient Technique to Continuously Monitoring Reverse kNN Queries. *Proceedings of the VLDB Endowment (PVLDB)*, 2(1): 1138-1149, 2009.
- [ICDE’09] Xiaoyan Liu, Xindong Wu, Huaiqing Wang, Rui Zhang, James Bailey, Kotagiri Ramamohanarao. Mining Distribution Change in Stock Order Streams. Proceedings of the 26th International Conference on Data Engineering (ICDE), pp. 105-108, 2010.
- [GeoInformatica’10] Cui Yu, Rui Zhang, Yaochun Huang, Hui Xiong: High-dimensional kNN joins with incremental updates. *GeoInformatica*, 1 (14), 55-82, 2010.
- [VLDB’10a] Rui Zhang, Martin Stradling. The HV-tree: a Memory Hierarchy Aware Version Index. *Proceedings of the VLDB Endowment (PVLDB)*, 3(1), 397-408, 2010.
- [VLDB’10b] Zhenhui Li, Bolin Ding, Jiawei Han, Roland Kays: Swarm: Mining Relaxed Temporal Moving Object Clusters. *Proceedings of the VLDB Endowment (PVLDB)*, 3(1-2), 723-734 , 2010.
- [VLDB’10c] Xin Cao, Gao Cong, Christian Jensen: Mining Significant Semantic Locations From GPS Data. *Proceedings of the VLDB Endowment (PVLDB)*, 3(1-2), 1009-1020 , 2010.
- [VLDBJ’10a] Sarana Nutanong, Rui Zhang, Egemen Tanin, Lars Kulik. Analysis and Evaluation of V*-kNN: An Efficient Algorithm for Moving kNN Queries. *VLDB Journal*, 19(3): 307-332, 2010.
- [VLDBJ’10b] Mohammed Eunus Ali, Egemen Tanin, Rui Zhang, Lars Kulik. A Motion-Aware Approach for Efficient Evaluation of Continuous Queries on 3D Object Databases. *VLDB Journal*, 19(5): 603-632, 2010.
- [SIGMOD’10] Abdullah Mueen, Suman Nath, Jie Liu: Fast approximate correlation for massive time-series data. *ACM SIGMOD International Conference on Management of Data (SIGMOD)* 2010.
- [TKDE’10] Sarana Nutanong, Egemen Tanin, Rui Zhang. Incremental Evaluation of Visible Nearest Neighbor Queries. *IEEE Transactions on Knowledge & Data Engineering (TKDE)*, 22(5): 665-681, 2010.
- [InfSys’10] Rui Zhang, H. V. Jagadish, Bing Tian Dai, Kotagiri Ramamohanarao. Optimized Algorithms for Predictive Range and KNN Queries on Moving Objects. *Information Systems*, 35(8): 911-932, 2010.
- [VLDBJ’12] Rui Zhang, Jianzhong Qi, Dan Lin, Wei Wang, Raymond Chi-Wing Wong. A Highly Optimized Algorithm for Continuous Intersection Join Queries over Moving Objects. *VLDB Journal*, 21(4): 561-586, 2012.