

Detecting Spatial Outliers: Algorithm and Application

Chang-Tien Lu

Spatial Database Lab

Department of Computer Science

University of Minnesota

ctlu@cs.umn.edu

Phone: 612-378-7705

Outline

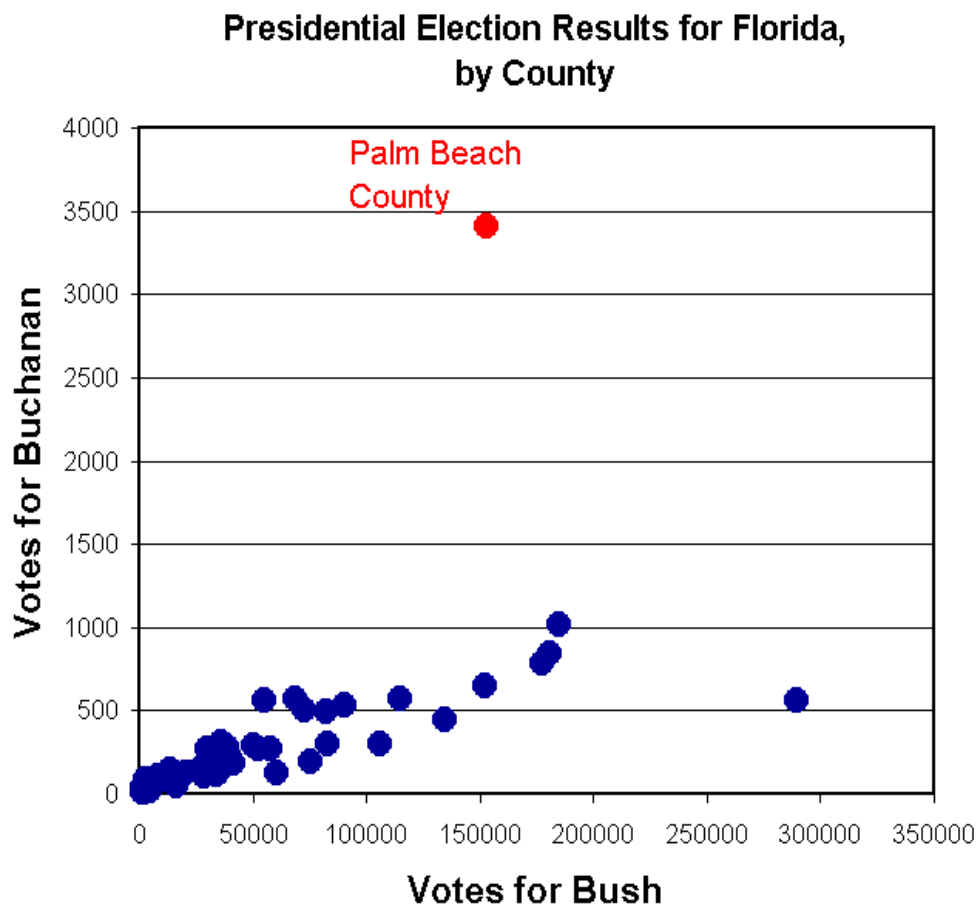
- Introduction
 - Motivation
 - Problem Definition
 - Related Work
- Proposed Approach
- Evaluation of Proposed Approach
- Conclusions

Spatial Data Mining

- Spatial Databases are too large to analyze manually
 - NASA Earth Observation System(EOS)
 - National Institute of Justice - Crime mapping
 - Census Bureau, Dept. of Commerce - Census Data
- Spatial Data Mining
 - Discover frequent and interesting spatial patterns for post processing (knowledge discovery)
 - Pattern examples: outliers, hot-spots, land-use classification

Spatial Outlier

- Definition
 - A data point that is extreme relative to its neighbors
 - Individual attribute value is not necessarily extreme in the total population, but is extreme in its adjacent area
- An example
 - Item: Palm Beach County,
Neighbors(item) = Counties in Florida



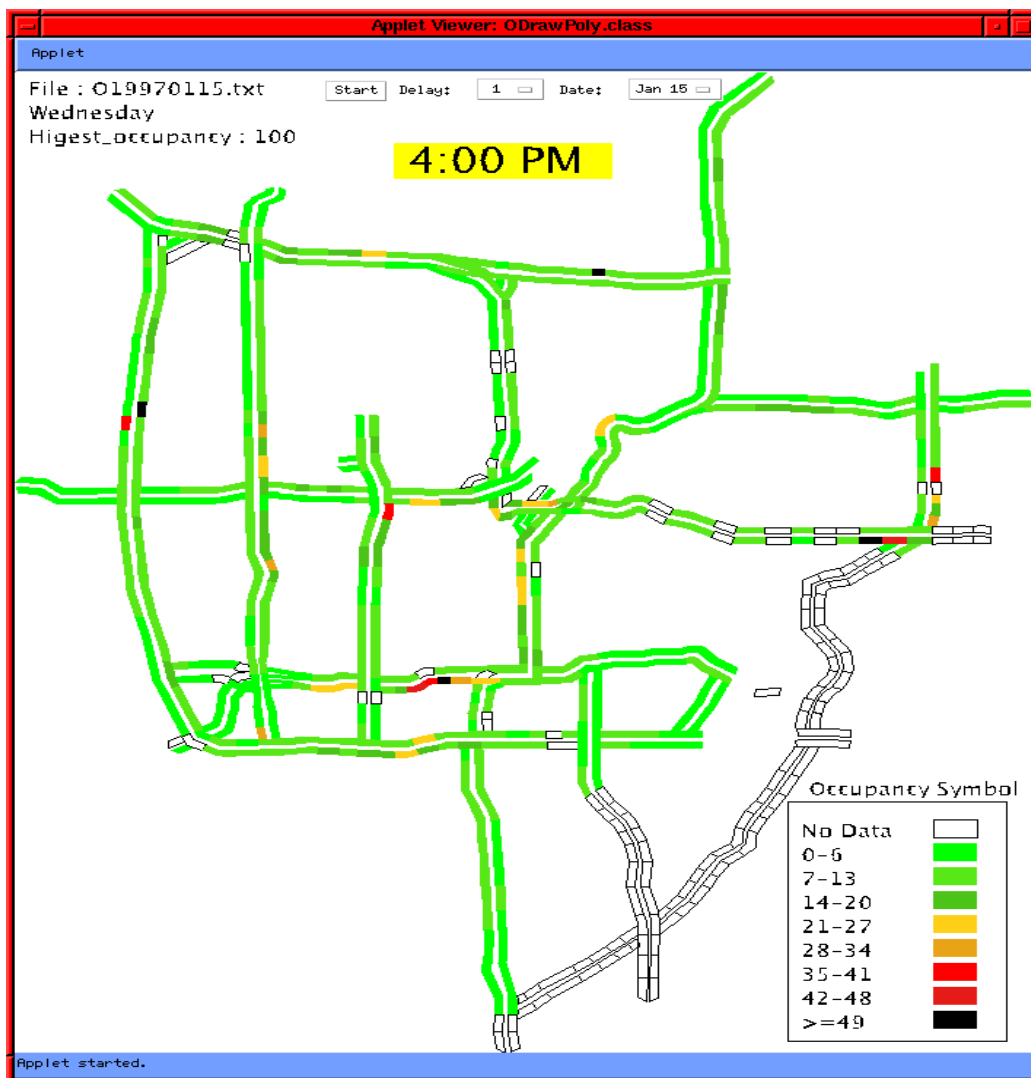
Prof. Greg Adams (Carnegie Mellon Univ.)

Prof. Chris Fastnow (Chatham College)

Source: <http://madison.hss.cmu.edu/buchanan-bush.gif>

Application Domain

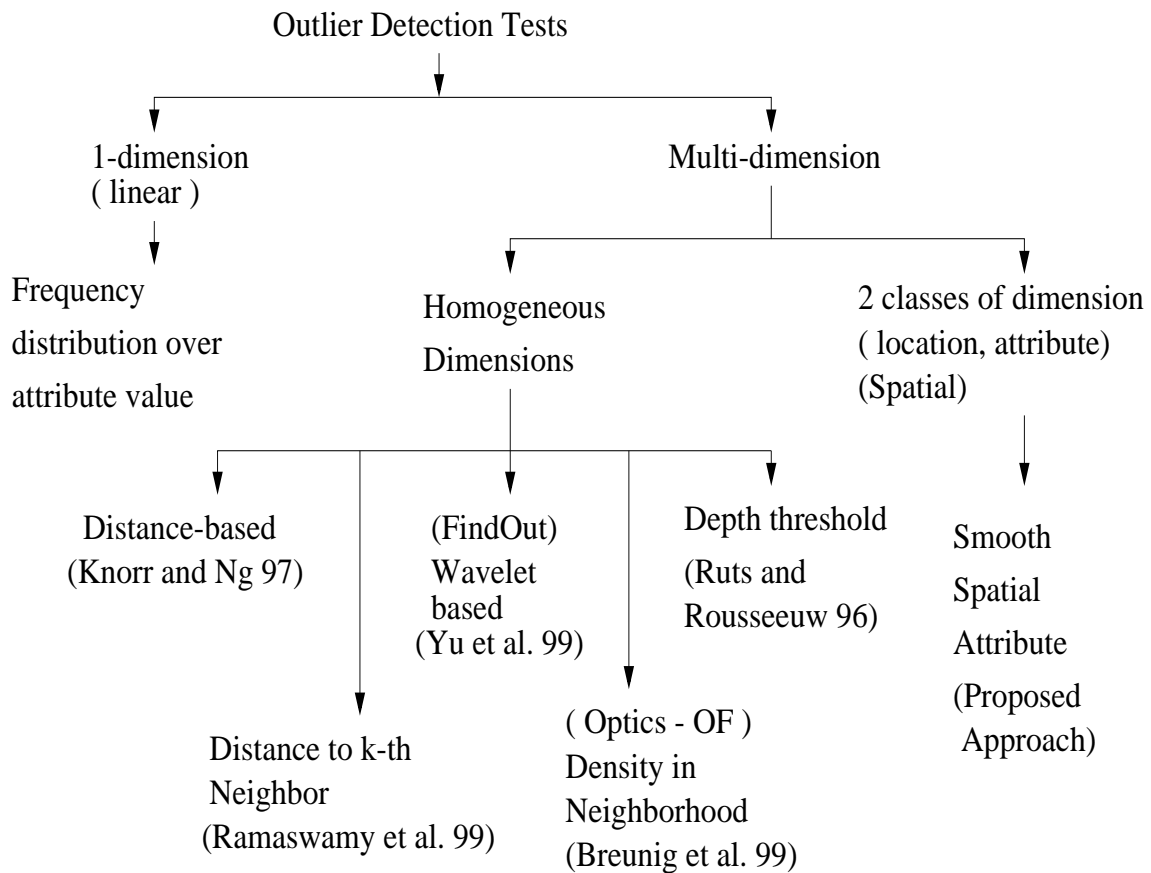
- Minneapolis-St. Paul (Twin-Cities) Traffic Data Set
 - 930 detectors (stations) installed on major highways
 - Periodical measuring attributes: volume, occupancy, speed
 - Interesting spatial outliers - discontinuities
 - Assume smooth spatial attribute



Problem Formulation

- Given
 - A spatial graph $G = \{V, E\}$
 - A neighbor relationship (K neighbors per node)
 - An attribute $f : V \rightarrow R$ (set of real numbers)
 - An aggregation function $f_{aggr} : R^K \rightarrow R$
 - Confidence level threshold θ
- Find
 - $O = \{v_i \mid v_i \in V, v_i \text{ is a spatial outlier}\}$
- Objective
 - Correctness: The attribute value of v_i is extreme, compared with the attribute values of its neighbors
 - Computational efficiency
- Constraints
 - Attribute distribution $\sim N(\mu, \sigma^2)$
 - Computation cost dominated by disk I/O cost

Related Work - Outlier Detection Tests



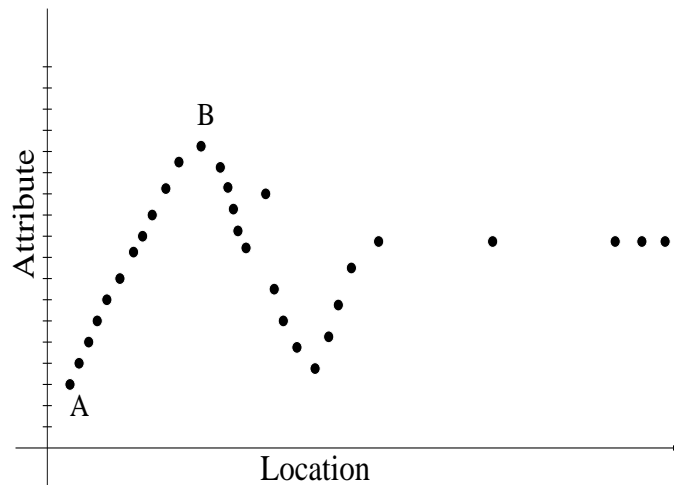
Related Work - Outlier Detection Tests

- Related work: two families
 - 1-dimension - ignores geographic location
 - Homogeneous Multi-dimension - mixes location with attributes
 - Proposed Approach (Spatial)
 - 2 classes of dimensions - location, attributes
 - Neighborhood - based on location dimensions
 - Difference - compares attribute dimensions
- Comparison of outlier detection methods

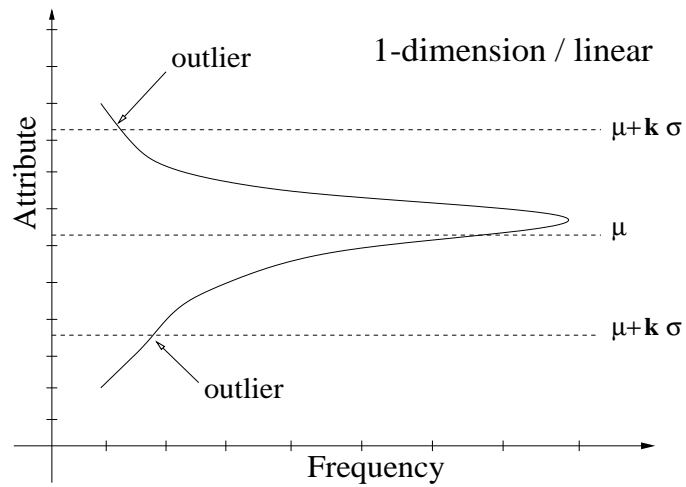
	1-dimension (linear)	Homogeneous Multi-dim.	Spatial/ Proposed
Neighbor Definition	N/A	location and attribute	location
Comparison	with population (μ , σ)	location and attribute	attribute values of neighbors

Examples of Outlier Types

- Base data set

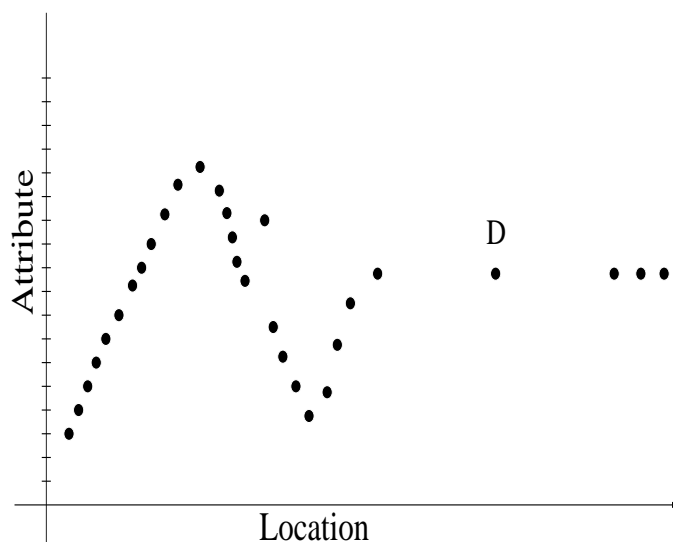


- 1-dimension (linear) outliers: A, B

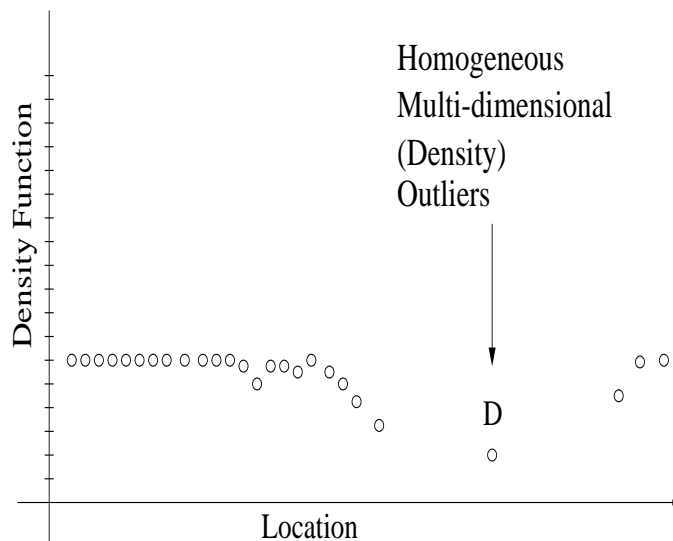


Examples of Outlier Types

- Base data set

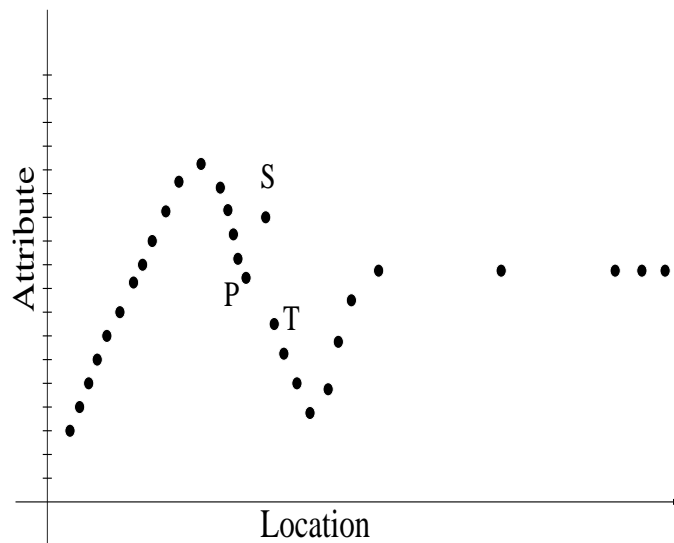


- Homogeneous Multi-dimension (Density based) outliers: D

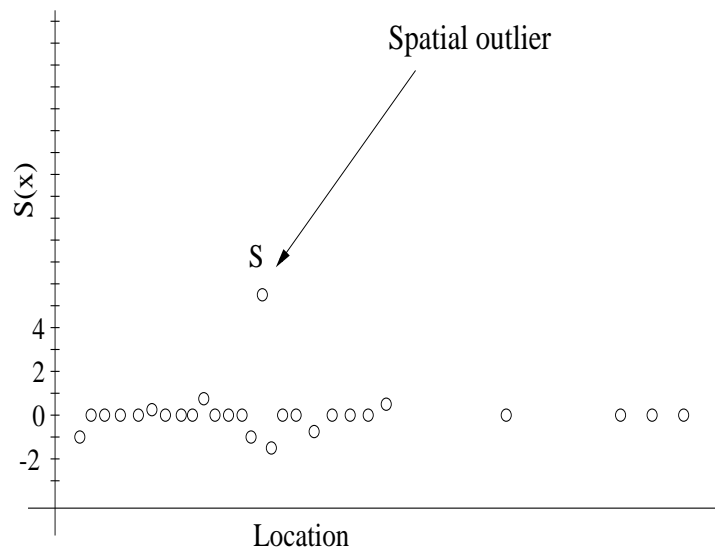


Example of Outlier Types

- Base data set



- Spatial outlier: S



Outline

- Introduction
- Proposed Approach
 - Test
 - Efficient algorithm for outlier detection
 - Cost model
- Evaluation of Proposed Approach
- Conclusions

Proposed Outlier Detection Test

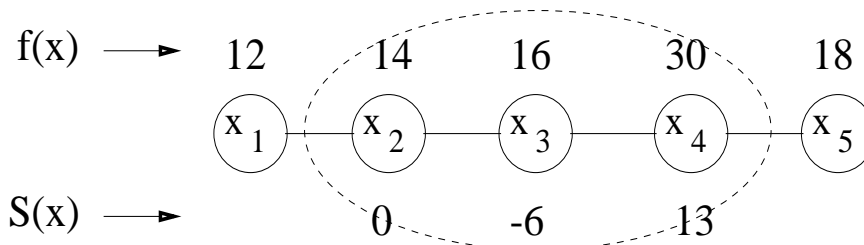
- Choice of Spatial Statistic

- $S(x) = [f(x) - E_{y \in N(x)}(f(y))]$
 - $f(x)$: attribute value of location x
 - $N(x)$: neighbors of x
 - $E_{y \in N(x)}(f(y))$: average attribute value of $N(x)$

- Test for Outlier Detection

- $|\frac{S(x) - \mu_s}{\sigma_s}| > \theta$

- An example



- Explanation of $S(x)$

- Consider location x_3
- Neighborhood $N(x_3) = \{x_2, x_4\}$
- $E_{y \in N(x)}(f(y)) = \frac{14+30}{2} = 22$
- $S(x) = [f(x) - E_{y \in N(x)}(f(y))] = 16 - 22 = -6$

Statistical Interpretation of Test

- Lemma: $S(x) = [f(x) - E_{y \in N(x)}(f(y))]$ is normally distributed if $f(x)$ is normally distributed.

- Proof Sketch:

- Attribute variable X_1 of node x_1 is normally distributed from $N(\mu, \sigma^2)$
- $x_{11}, x_{12}, \dots, x_{1k}$ are k neighbors of x_1
- Attribute variables $X_{11}, X_{12}, \dots, X_{1k}$ of nodes $x_{11}, x_{12}, \dots, x_{1k}$ are normally distributed from $N(\mu_{1i}, \sigma_{1i}^2), 1 \leq i \leq k$

- $(X_1, X_{11}, \dots, X_{1k})^T \sim N \left(\begin{pmatrix} \mu_1 \\ \mu_{11} \\ \cdot \\ \mu_{1k} \end{pmatrix}, \begin{pmatrix} \sigma_1^2 & \sigma_1 \sigma_{11} \rho_{X_1, X_{11}} & \cdot & \cdot \\ \cdot & \sigma_{11}^2 & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \sigma_{1k}^2 \end{pmatrix} \right)$

- Linear Transformation

$$S(x) = X_1 - \bar{X}_k = (1, -\frac{1}{k}, -\frac{1}{k}, \dots, -\frac{1}{k}) \begin{pmatrix} X_1 \\ X_{11} \\ \cdot \\ X_{1k} \end{pmatrix} \sim N(A\mu_k, A\Sigma A')$$

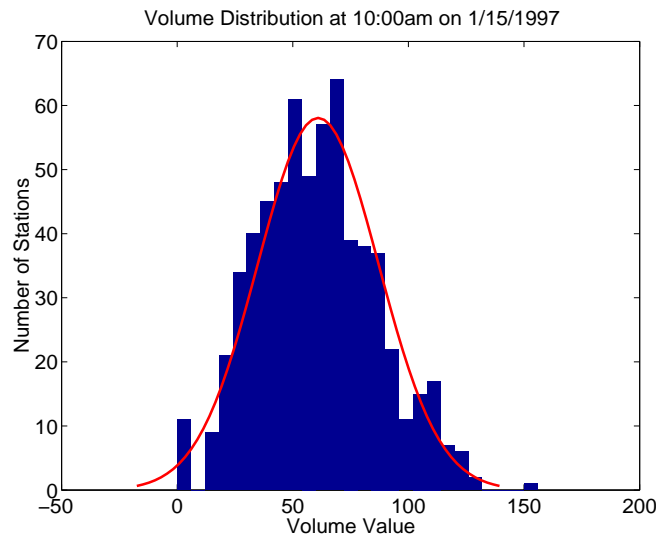
Where $A = (1, -\frac{1}{k}, -\frac{1}{k}, \dots, -\frac{1}{k})$, $\mu_k = \begin{pmatrix} \mu_1 \\ \mu_{11} \\ \cdot \\ \mu_{1k} \end{pmatrix}$, $\Sigma = \begin{pmatrix} \sigma_1^2 & \sigma_1 \sigma_{11} \rho_{X_1, X_{11}} & \cdot & \cdot \\ \cdot & \sigma_{11}^2 & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \sigma_{1k}^2 \end{pmatrix}$

- Standardization

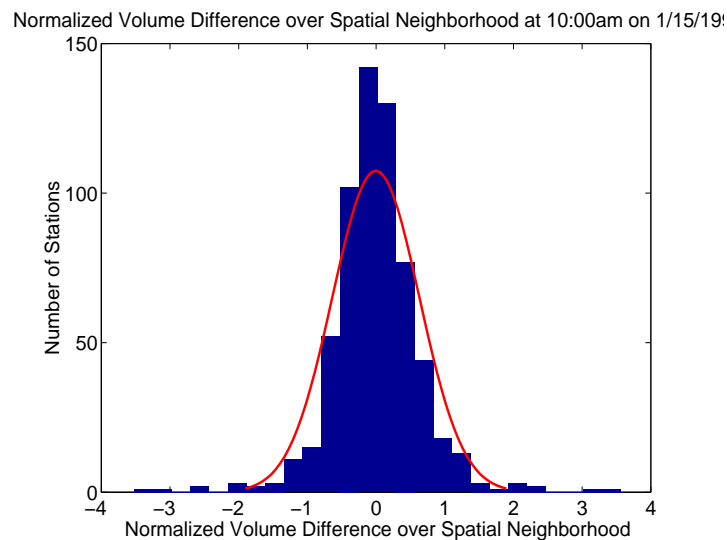
$$\frac{(X_1 - \bar{X}_k) - A\mu_k}{\sqrt{A\Sigma A'}} \sim N(0, 1)$$

Evaluation of Statistical Assumption

- Distribution of highway attribute $f(x)$ looks normal



- Distribution of $S(x) = [f(x) - E_{y \in N(x)}(f(y))]$ looks normal too!



Proposed Algorithms

- Goal - I/O efficiency
- Algorithm A1: Test Parameter Computation
 - Build a statistical model
- Algorithms A2: Test Result Computation
 - (a) Check whether nodes of a randomly generated set are outliers
 - (b) Check whether nodes of a route are outliers

Proposed Algorithms

- Algorithm A1: steps
 - For each location x
 - Retrieve data record of $x = (f(x), \text{identities of neighbors}(x))$
 - Get-All-Neighbors(x): Retrieve data records of neighbor(x)
 - * if neighbor y is not in the memory buffer
 - * request another I/O operation
 - Compute $S(x) = [f(x) - E_{y \in N(x)}(f(y))]$
 - Accumulate $Sum(S(x))$, $Sum(S^2(x))$, and $Count(x)$
 - Compute $\mu(S(x))$ and $\sigma(S(x))$
- I/O cost is determined by
 - Dominant operation: Get-All-Neighbors(x)
 - I/O cost of Get-All-Neighbors(x) is determined by the clustering efficiency
 - Grouping nodes into disk page

Proposed Algorithm

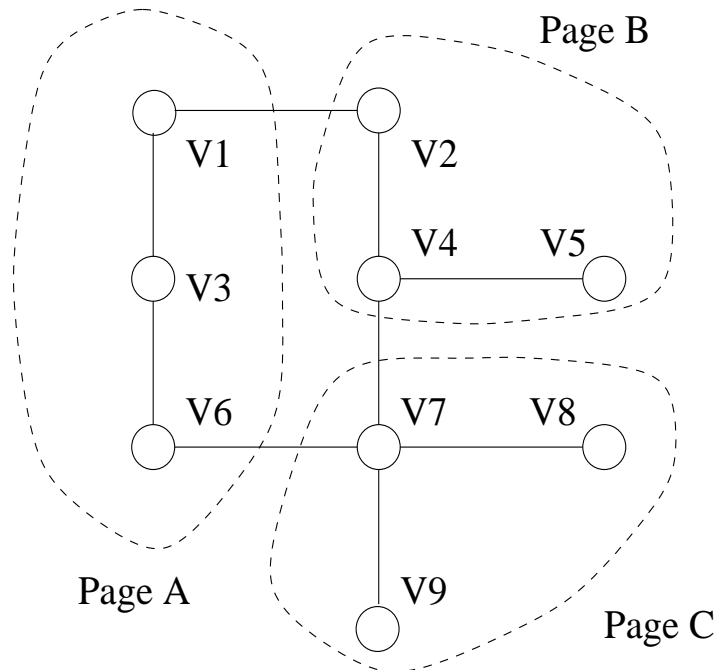
- Algorithm A2(a): steps
 - For each location x along a route
 - Retrieve data record of $x = (f(x), \text{identifies of neighbors}(x))$
 - Get-All-Neighbors(x): Retrieve data records of neighbor(x)
 - * if neighbor y is not in the memory buffer
 - * request another I/O operation
 - Compute $S(x) = [f(x) - E_{y \in N(x)}(f(y))]$
 - if $|\frac{S(x) - \mu_s}{\sigma_s}| > \theta$
 - Declare x as an outlier
- Algorithm A2(b): steps
 - For each location x within the random set
 - Retrieve data record of $x = (f(x), \text{identities of neighbors}(x))$
 - Get-All-Neighbors(x): Retrieve data records of neighbor(x)
 - * if neighbor y is not in the memory buffer
 - * request another I/O operation
 - Compute $S(x) = [f(x) - E_{y \in N(x)}(f(y))]$
 - if $|\frac{S(x) - \mu_s}{\sigma_s}| > \theta$
 - Declare x as an outlier

I/O Cost Model

- Definition
 - CE: Clustering Efficiency
 - N: Total number of nodes
 - Bfr: Blocking factor (number of nodes in a page)
 - K: Number of neighbors for each node
 - L: Number of nodes in a route
 - R: Number of nodes in a random set
- Assume two memory buffers
- Cost model of A1
 - $(N/Bfr) + N * K * (1 - CE)$
 - The cost to retrieve all nodes: (N/Bfr)
 - The cost to retrieve neighbors of all nodes: $N * K * (1 - CE)$
- Cost model of A2(a)
 - $L * (1 - CE) + L * K * (1 - CE)$
- Cost model of A2(b)
 - $R + R * K * (1 - CE)$

Clustering Efficiency Parameter

- Computation cost (I/O cost) is determined by Clustering Efficiency(CE)
- CE definition:
 - (Total number of unsplit edges)/(Total number of edges)
 - Probability[v_i and a neighbor of v_i are stored in the same disk page]
- An example
 - $CE = \frac{9-3}{9} = \frac{6}{9} = 0.66$



Outline

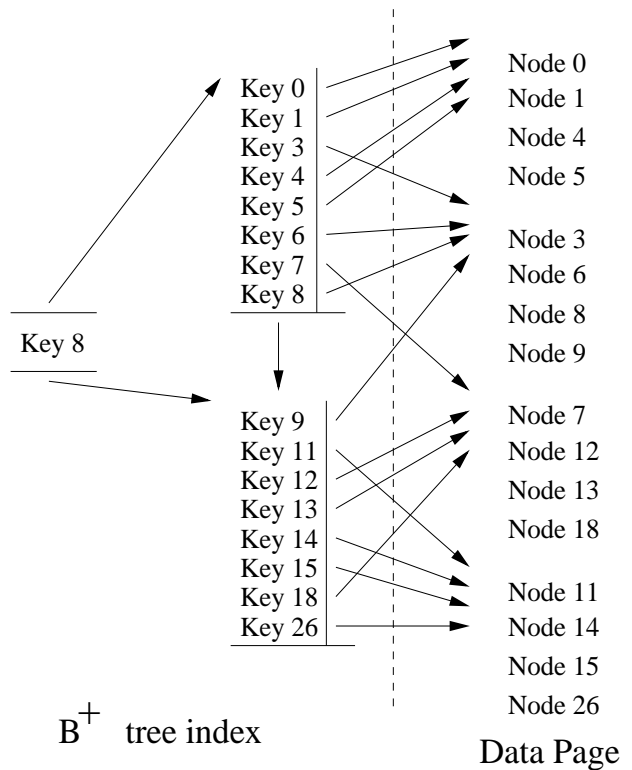
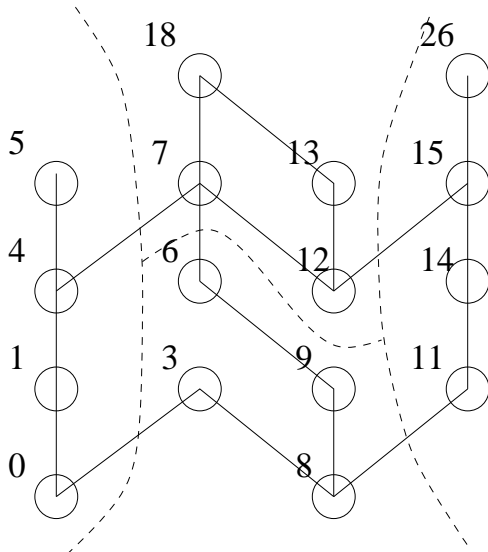
- Introduction
- Proposed Approach
- Evaluation of Proposed Approach
 - Candidates (Clustering Methods)
 - Experiment Design
 - Results
- Conclusions

Experimental Evaluation (Summary)

- Hypothesis:
 - I/O cost of the algorithms is determined by the clustering efficiency
- Physical Data Page Clustering Method
 - Graph-based method: CCAM
 - Geometric method: Cell Tree
 - Geometric method: Z-order
- Metrics: Clustering Efficiency (CE), I/O cost
- Benchmark data
 - Minneapolis - St. Paul Traffic data
- Benchmark tasks
 - Test Parameter Computation
 - Test Result Computation (Route)
 - Test Result Computation (Set of Random Nodes)
- Variable Parameters
 - Buffer size
 - Disk page size

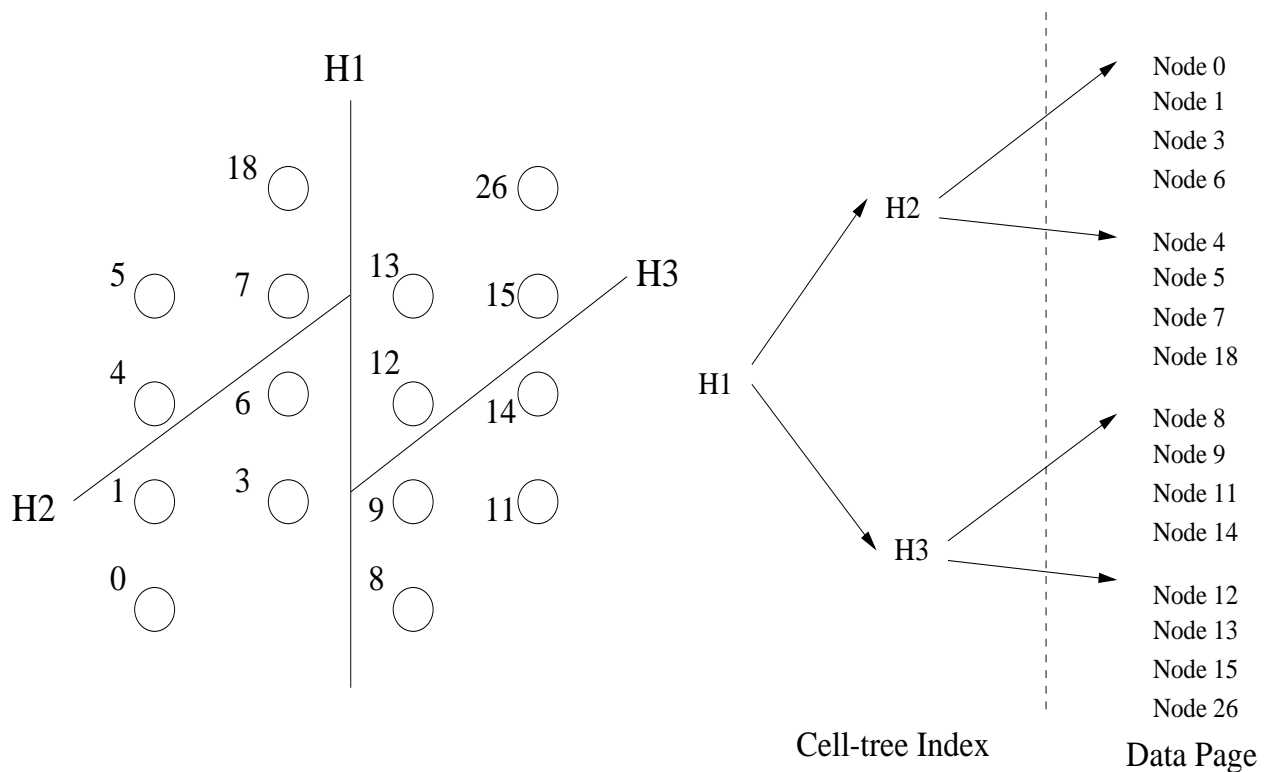
Clustering Method: CCAM

- Connectivity Clustered Access Method
- Cluster the nodes via min-cut graph partitioning
- Use B+ tree with Z-order as the secondary index



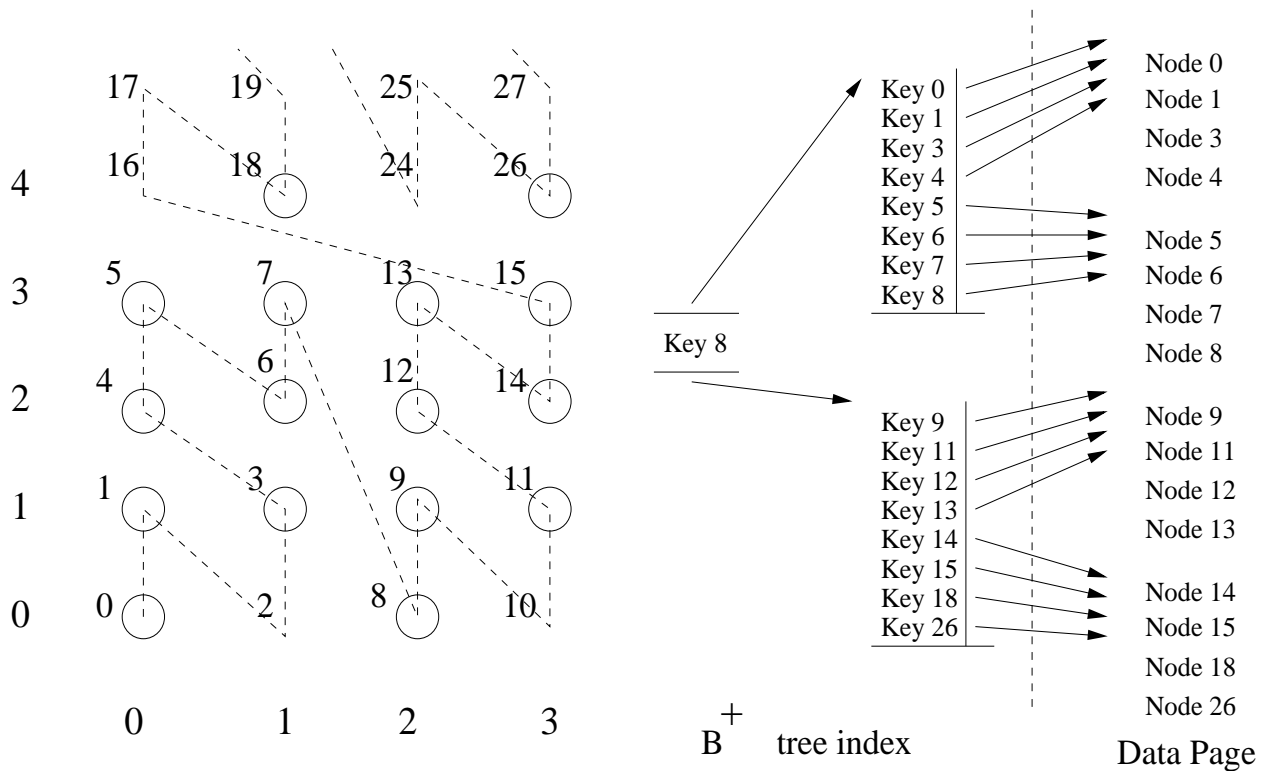
Clustering Method: Cell Tree

- Binary Space Partitioning(BSP)
- Decompose universe into disjoint convex subspaces
- Each leaf node corresponds to one of the subspaces
- Each tree node is stored on one disk page
- Cannot exploit edge information, pure geometric



Clustering Method: Z-order

- Impose a total order on the nodes
- Z order = interleave (bits of X , bits of Y)
- Use B+ tree as the primary index
- Cannot exploit edge information, pure geometric



Experiment Design

- Experiment Data Set
 - Twin-Cities Traffic Data
 - Each data object (node)
 - Attribute Values
 - Neighbor List
 - Size: 256 bytes

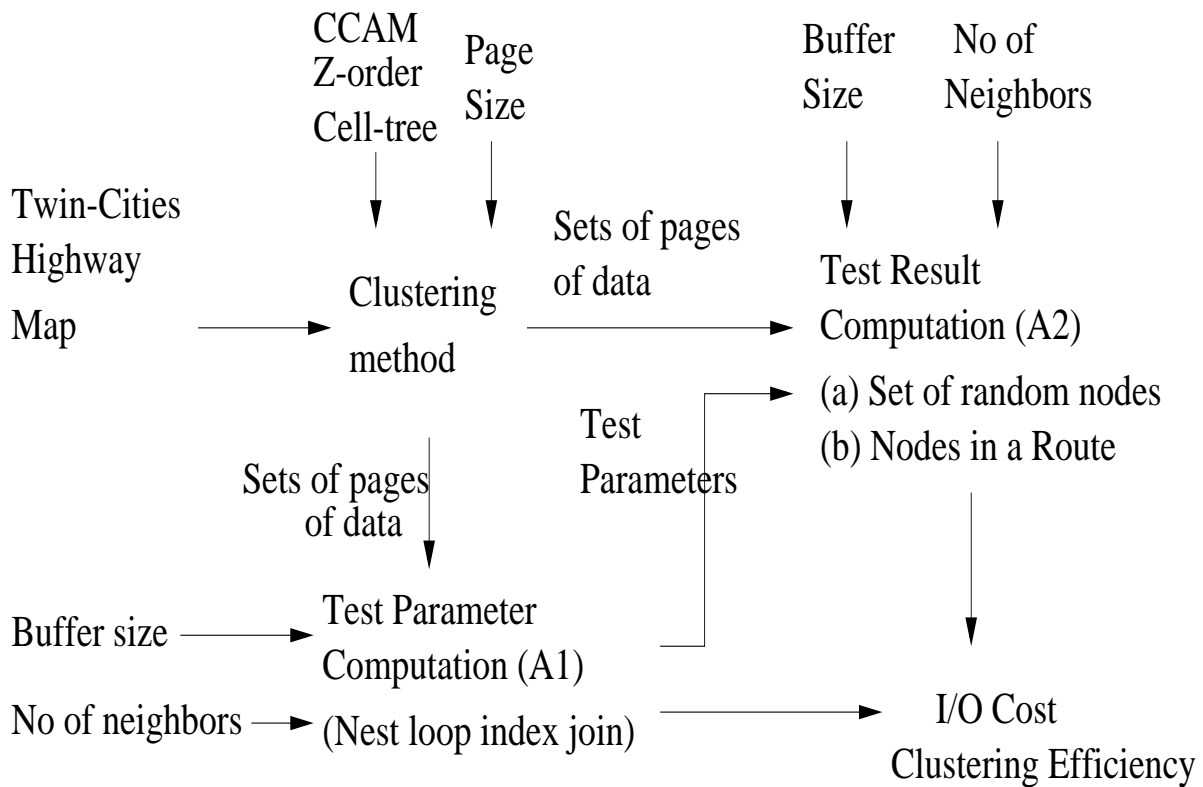


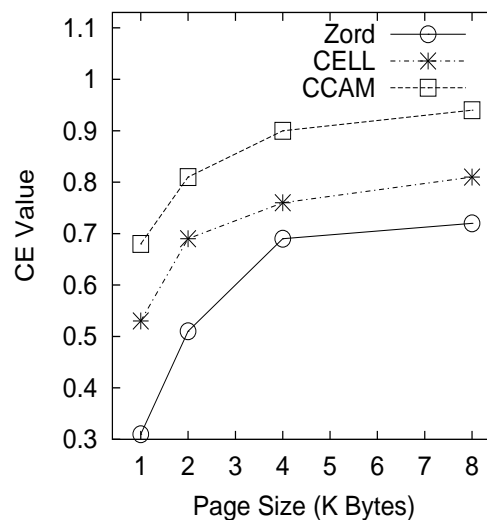
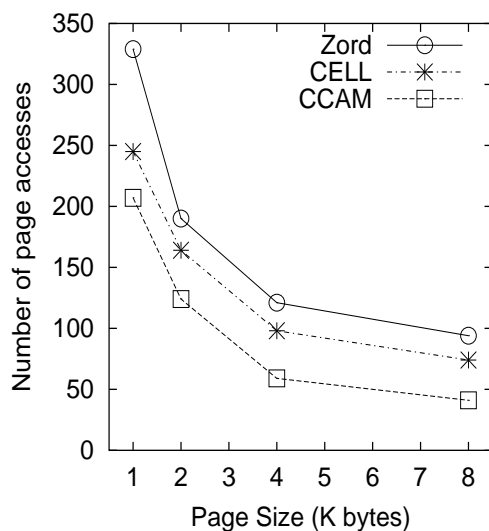
Figure 1: Experimental Layout

Experimental Observation and Results

- Step 1: Test Parameter Computation
 - Compute global data distribution
 - Nest Loop index join
- Step 2: Test Result Computation
 - Detect outliers along a Route
 - Detect outliers in a set of random nodes

Test Parameter Computation: Effect of Page Size

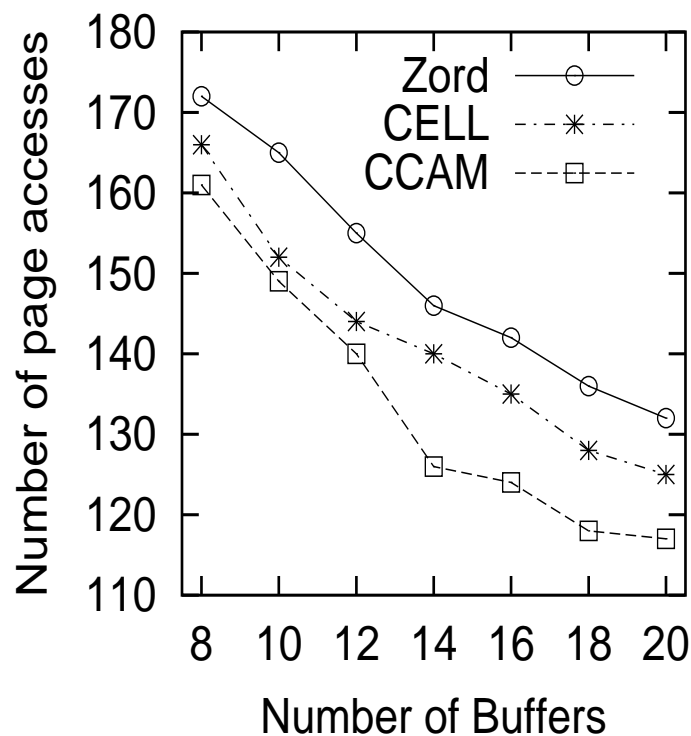
- Fixed Parameters
 - Buffer Size: 64k
- Variable Parameters
 - Page size, clustering strategy



- CCAM has the best performance
- CCAM has the highest CE value
- High CE \Rightarrow Low I/O cost
 - Cost Model: $(N/Bfr) + N * K * (1 - CE)$
- Increase page size \Rightarrow reduce number of page accesses

Test Parameter Computation: Effect of Buffer Size

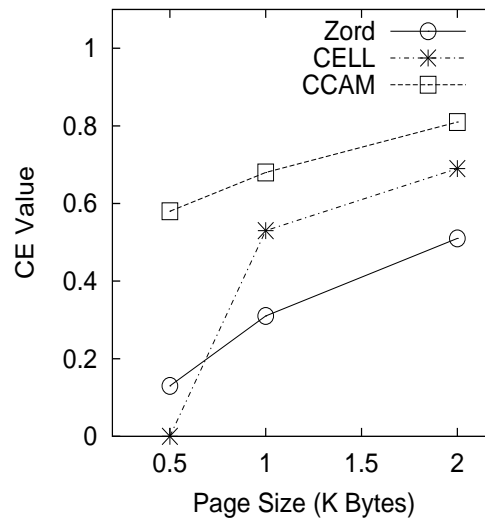
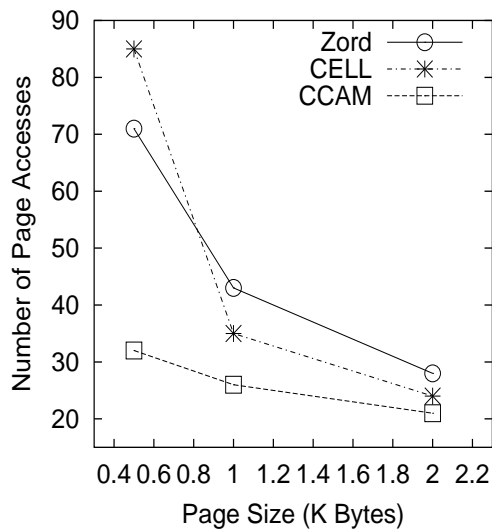
- Fixed Parameters
 - Page size: 2K
 - Clustering Efficiency:
 - CCAM=0.81, Cell=0.69, Z-ord=0.51
- Variable Parameters
 - Number of buffers, clustering strategy



- Increase Buffer size => reduce number of page accesses
- CCAM has the best performance

Test Result Computation (Route): Effect of Page Size

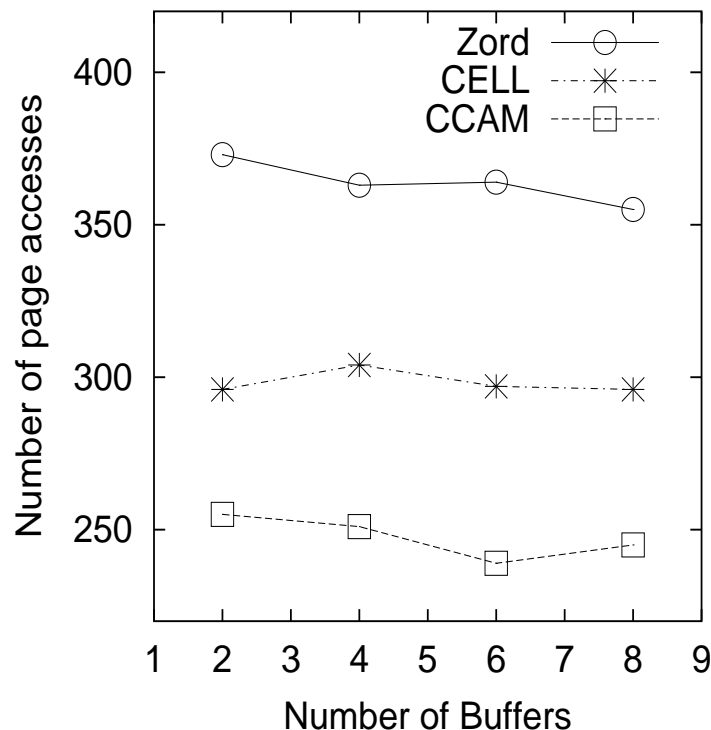
- Average I/O cost of outlier query over 50 routes
- Fixed Parameters
 - Buffer size: 4 Kbytes, Data point size = 256 bytes
- Variable Parameters
 - Page size, clustering strategy



- Increase page size => reduce no of page accesses
- CCAM has the best performance
- CCAM has the highest CE value
- High CE => Low I/O cost
 - Cost Model: $L \cdot (1 - CE) + L \cdot K \cdot (1 - CE)$
- Cell Tree has zero CE value when Bfr=2
- Increase page size => Performance gap reduces

Test Result Computation (Random Nodes) : Effect of Buffer Size

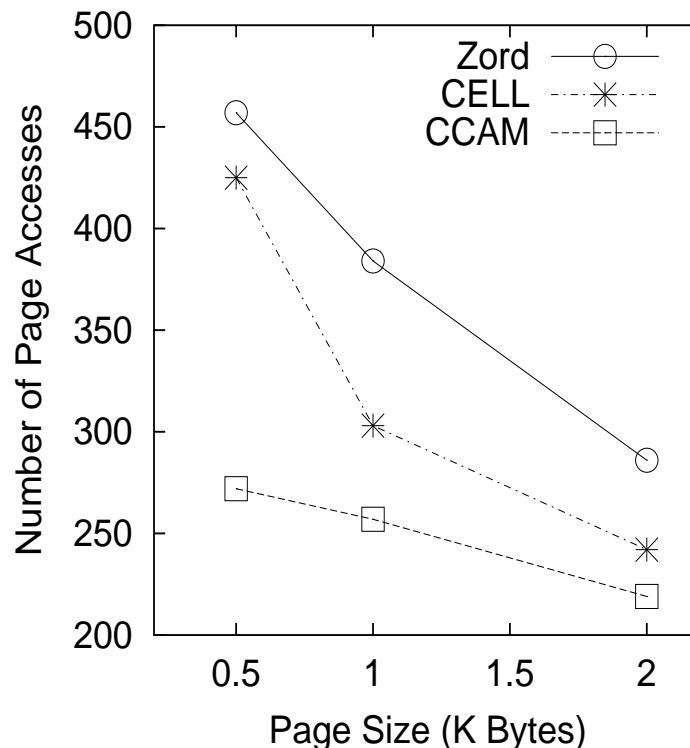
- Average I/O cost of outlier query over 50 sets of random nodes
- Fixed Parameters
 - Number of random nodes: 150, page size : 1 Kbytes
 - CE value : CCAM = 0.68 Cell = 0.53 Z-ord = 0.31
- Variable Parameters
 - Buffer Size, clustering strategy



- Increase buffer size => No effect
- CCAM has the best performance

Test Result Computation (Random Nodes) : Effect of Page Size

- Average I/O cost over 50 sets of random nodes
- Fixed Parameters
 - Number of random nodes: 150 points, buffer size: 4K
- Variable Parameters
 - Page size, clustering strategy



- Increase page size => reduce no of page accesses
- CCAM has the best performance

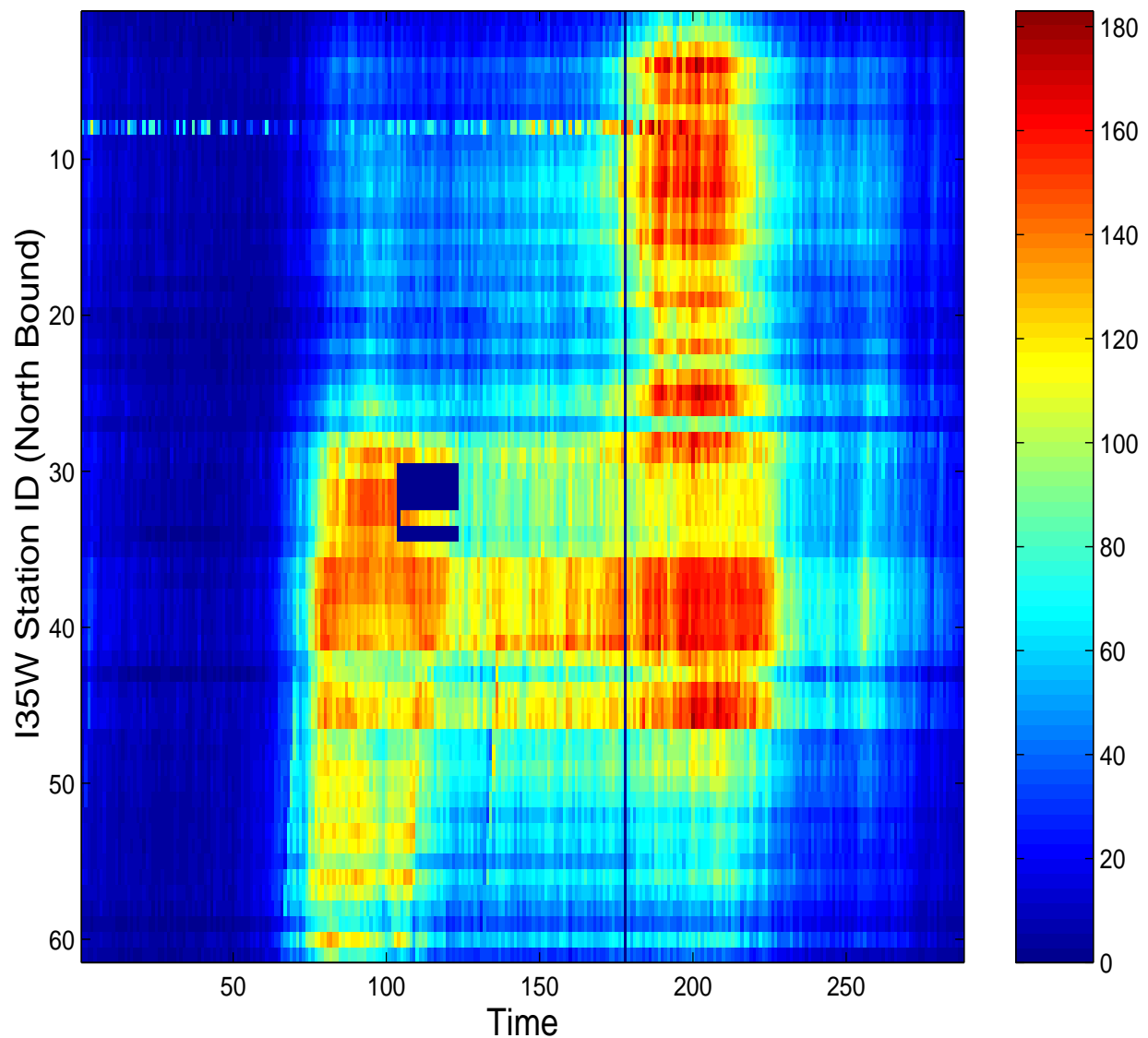
Summary of Experimental Results

- Clustering Efficiency
 - CCAM achieves higher clustering efficiency than Cell tree and Z-order
- Test parameter and test result computation
 - CCAM has lower I/O than Cell tree and Z-order
- Higher CE leads to lower I/O cost
 - CE is a good predictor of relative I/O performance
- Page size improves clustering efficiency of all methods
 - Reduces performance gap between methods

Application Domain

- I-35W North Bound
 - Volume: the number of vehicles passing a station within 5 minutes

Traffic Volume(Time v.s. Station)



Application Domain: Twin-Cities Traffic Data

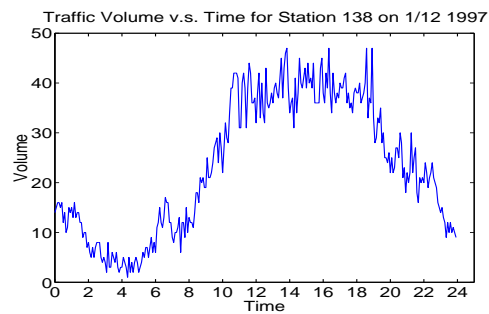


Figure 2: Station 138 on 1/12 1997

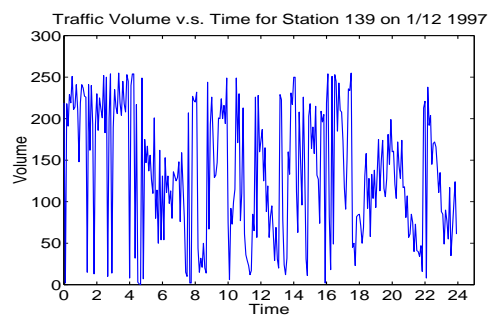


Figure 3: Station 139 on 1/12 1997

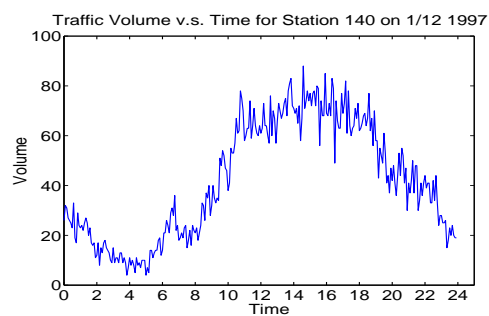


Figure 4: Station 140 on 1/12 1997

Conclusion

- A simple statistically meaningful test for spatial outliers
- Propose algorithms for spatial neighbor-based outlier detection
 - Finds many interesting outliers(e.g., faulty detectors) not found by other methods
- Recognize the computation structure of outlier detection algorithms
 - θ Self Join
 - Get-All-Neighbor() dominates I/O cost
- Develop Algebraic Cost Models
- Evaluate Alternate Page Clustering Methods

Future Direction

- Extend Spatial Outlier Detection Test
 - Multi attributes (non-location)
 - Location attribute includes time
 - Temporal and Spatial-Temporal Outliers
- Extend Experiments
 - NASA data sets - uniform grid
 - Hypothesis - Geometric clustering may perform well
- Explore other spatial patterns beyond spatial outlier
 - Land-use classification
 - Co-locations
 - Example:
 - Fire ignition source feature
 - Needle vegetation type feature
 - Drought feature