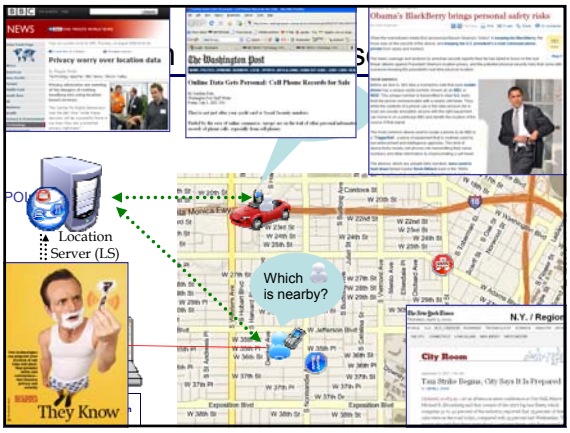
Blind Evaluation of Nearest Neighbor Queries Using Space Transformation to Preserve Location Privacy

Ali Khoshgozaran and *Cyrus Shahabi*
University of Southern California
Los Angeles, CA 90089-0781
[jafkhosh, shahabi]@usc.edu
http://infolab.usc.edu

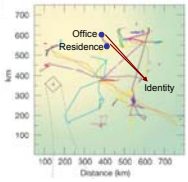


Isn't Confidentiality Enough?

Sensitive information obtained by anonymous location data

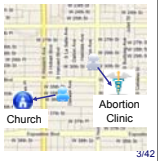
- Baraba´si et al., Nature'08

Human Mobility ⟹ Spatial Probability Distribution



- Anonymous queries leak information

Location Queries ⟹ Affiliations (political, religious, etc.)



Problem Definition

Objects S = {o_1, o_2, …, o_n}

kNN Query

~~kNN with respect to query point Q~~

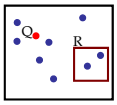
Prevalent Spatial Queries in Location Based Services

S' ⊆ S of k objects where for any object o' ∈ S' and o ∈ S - S' D(o',Q) ≤ D(o,Q)

Range (Window) Query

Range with respect to query window R:

S' ⊆ S of objects where for any object o' ∈ S' o' is within R



What is required to make these queries "Privacy Aware"?

Blind Evaluation Criteria

When querying LS, the location of the querying user should not be revealed to untrusted entities through R, Q, or the query result set.

Trust and Threat Model

- Users subscribe to LS's services
  - LS owns a publicly available POI database DB
- LS is *honest* but *curious*
  - Database software is trusted
  - LS might passively exploit sensitive information
- Any entity in the system can be adversarial
  - The LS and other clients
  - Slightly different for querying other users
- Secure client/server communication channel
  - Any privacy violation should have included LS
  - We focus on LS as the most powerful adversary

Privacy/Efficiency Dilemma

- Privacy: Hiding knowledge of object & query *locations* from LS
- Efficiency: LS requiring this knowledge for efficient query processing

Privacy ⟷ Efficiency

Our Contribution

Information-theoretic secrecy
- Privacy against an adversary with unbounded computational resources and infinite time
- Lower communication & computation bound: linear w.r.t. database size

Server knowing all information about queries and object locations
- No user location privacy possible

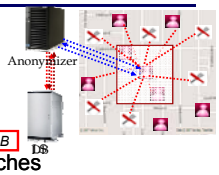
Related Work

Privacy

Cryptographic Techniques
- S. Zhong et al., TR'04
- Indyk et al. TCC'06
- G. Zhong et al., PET'07

No spatial query processing (MPC schemes)
O(n) computation and/or communication

Our Goal: Avoiding a linear scan of the entire DB



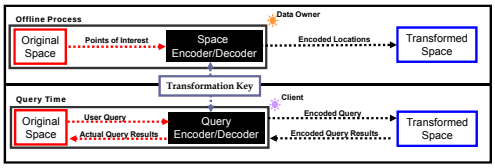
Efficiency

K-anonymity/Cloaking Approaches
- Gruteser et al. MobiSys'03
- Gedik et al. ICDCS'05
- Mokbel et al. VLDB'06
- Kido et al. ICPS'05
- Chow et al. GIS'06
- Ghinita et al. WWW'07& SSTD'07

Trusting an Anonymizer
Single point of failure/attack
Sensitive to number of subscribed users
Assuming all users are trustworthy
Dependence on other user locations
No query processing

Our Goal: Complete cloaking and anonymity

Space Encoding



Transformation Properties:
- ✓Efficiency (locality preserving)
- ✓Privacy (irreversible)

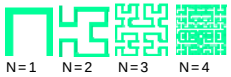
Space-Filling Curves

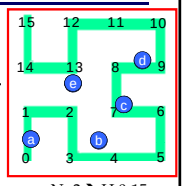
Background: Space Filling Curves

- Passing through (indexing) all points without crossing itself
- Example: <a,b,c,d,e> ➔ <0,4,7,9,13>

H-values

- *Proximity & distance preserving*





N=2➔ H:0-15

Hilbert Curves

$H = \ell(P)$:[0,2^N-1]^d ➔ [0,2^{Nd}-1]

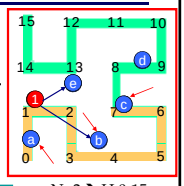
d=2: $H = \ell(P)$:[0,2^N-1]^2 ➔ [0,2^{2N}-1]

Hilbert Curves: Proximity Preserving

- Proximity in Hilbert space
- Example: <a,b,c,d,e> ➔ <0,4,7,9,13>

H-values

- 2NN(Q)=e because D(Q,e)< D(Q,b)



N=2➔ H:0-15

Approximate distance preservation

Complexity: Constant computation and communication

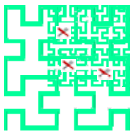
- Each node visited contains at least one object

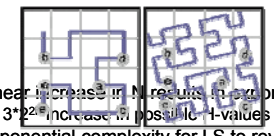
Hilbert Curves: One-wayness

- Five parameters decide how points are traversed (indexed)
- Possible when curve parameters are unknown
  - Space Decoding Key

SDK={X_0, Y_0, Θ, Γ, N}

Starting Point|Orientation|Scaling|Order

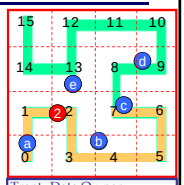




- Linear increase in N results in exponential increase in H-values
  - 3*2^{2N} increase in possible H-values
- Exponential complexity for LS to reverse the transformation
  - No knowledge of SDK

2-Phase kNN Query Processing

- Offline Space Encoding
  - Encoding points of original space
    - Trent chooses SDK
    - Trent constructs a lookup table DB
    - DB={<a,0>,<b,4>,<c,7>,<d,9>,<e,13>}
    - Trent encrypts objects identifiers
  - Trent uploads DB to LS
- Online Query Processing
  - Alice encodes her query point Q: $H = \ell(X_Q, Y_Q)$
  - Knowing *H* and k, LS computes the result set
  - H=2,k=3 ➔RS*={0,4,7}={$\ell(X_a,Y_a)$,$\ell(X_b,Y_b)$,$\ell(X_c,Y_c)$}
  - Knowing SDK, Alice gets *RS* ={(X_a,Y_a),(X_b,Y_b),(X_c,Y_c)}



Trent=Data Owner
LS= Location Server
Alice=User

Curve Rotation & kNN Search

- Issue: Approximation due to dimension reduction
  - Hilbert curves widely used for dimension reduction
- Indexing data with a rotated dual Hilbert curve
- Drawbacks of using a single curve:
  - 1. N ↑ (linear) ➔ Missed Sides ↑ (exponential)

  - 2. Reducing number of neighbors from 4 to 2

13/42

Dual Curve Query Resolution (DCQR)

- Trent indexes objects using both curves (SDK/SDK')
- Queries are evaluated on both curves
- kNN Search:
  - Alice computes $H = \ell(X_Q, Y_Q)$ & $H' = \ell(X_Q, Y_Q)$ for Q
  - LS runs two separate queries and returns 2k points to Alice
  - Alice sorts the result sets and pick the top k
- Query complexity is not affected by DCQR

14/42

Dual Curve Indexing

- We use a dual curve which is a replication of the original curve rotated and shifted
  - Rotation improves kNN search precision with no effect on range search
  - Translation reduces server throughput in processing range queries with positive effect on kNN search

15/42

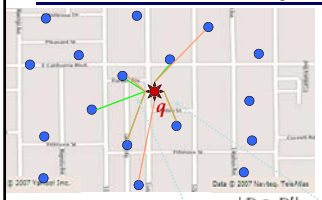
Performance Evaluation

- **Methodology:** issuing 1000 kNN queries with random origin
- **Datasets (10000 data points):**
  - Uniform Distribution
  - **Real-world**
    - Restaurants from NAVTEQ in a 26 by 26 mile area in Los Angeles
  - Skewed
    - Four clusters of points: 99% Gaussian with (σ=0.05 and Random μ) and 1% uniform
- **Evalu**
  - **Qu**
  - **Ap**
- **Param**
  - **Cu**
  - **Da**
- **Assu**

(a) Uniform (b) Real-world (c) Skewed

16/42

Accuracy Metrics



- *Actual Query Results*

  $R = \{o_1, o_2, \ldots, o_K\}$
- *Approximated Query Results*

  $R' = \{o'_1, o'_2, \ldots, o'_K\}$
- $R \cap R'$ = *Common Results*

Metric 1: The Resemblance: $\alpha = \frac{|R \cap R'|}{|R|}$

Metric 2: The Displacement: $\beta = \frac{1}{K}\left(\sum_{i=1}^{K} \|Q - o'_i\|\right) / \left(\sum_{i=1}^{K} \|Q - o_i\|\right)$

Accuracy vs. Displacement

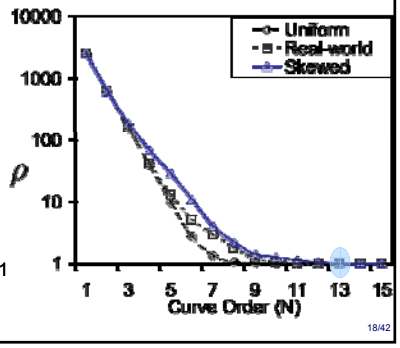
17/42

Effect of the Curve Order (N)

$$\rho = \frac{\sum_{i=1}^{N} O_i}{2^{2N}}$$

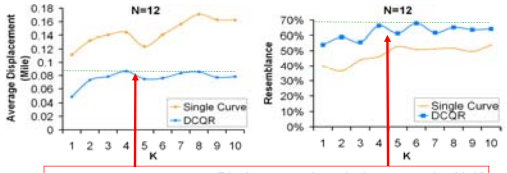
Ideally ρ≤1

Uniform (skewed):
First (last) to hit ρ=1



18/42

Single Curve Approach Vs. DCQR

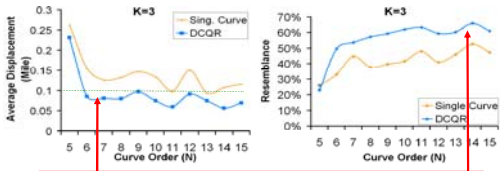


Displacement doesn't change much with K

Resemblance improves with K

DCQR improves the quality of overall results significantly:
Higher *Resemblance* (14% on average)
Smaller *Displacement* (0.06 mile/96 meters on average)

Single Curve Approach Vs. DCQR



Displacement reduces as N grows

Resemblance improves as N grows

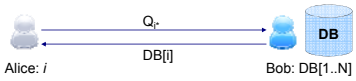
DCQR improves resemblance around 15% and displacement around 0.05 mile (80 meters) on average.

Location Privacy through Information Hiding

- Achieving Location Privacy by
- Hiding user identity
  - Who's accessing? (orthogonal to our work)
- What is being accessed?
  - Developing a secure and privacy aware spatial index
- Developing such privacy index reduces to
  - 1. secure index navigation
  - 2. private object retrieval

21/42

Private Information Retrieval



Examples:
- Patent Database
- Gold Mines
- Location Privacy

Variations:
- Information Theoretic PIR
  - Chor et al. 1998
- Computational PIR
  - Kushilevitz et al. 1997
- Hardware-based
  - Asonov et al. 2003

22/42

Discussion

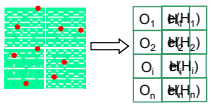
- Strengths?
  - Computation/Communication Cost
  - Lightweight client overhead
- Weaknesses?
  - Approximate
  - Prior Knowledge
    - Object distributions
    - Correlation queries

Privacy ⟷ Efficiency

23/42

Attacking SDK: Encrypting H-values

SDK_{trent}



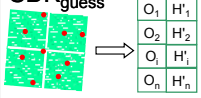
(Order Preserving) Encrypting H-values

II

SDK_{guess}=SDK_{trent}

$H = \ell(X_Q, Y_Q) \Longrightarrow Q$

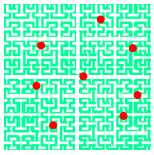
SDK_{guess}



24/42

Attacking SDK: Random Translation

- Before indexing, points are first translated using a random vector <ε,έ>
  - Analogous to the notion of salt in cryptography



25/42

Approximating SDK

- Assume LS knows precise values of N, Θ, Γ and X_0 and wants to guess Y_0 by Y'_0
- LS indexes objects with SDK_{guess} and compares DB_{guess} with DB

N

$|Y_0-Y'_0|$

	13	14	15	20	25	30
0.01	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
0.001	0.0%	0.02%	0.0%	0.01%	0.0%	0.0%
0.0001	0.02%	0.0%	0.0%	0.02%	0.0%	0.02%
0.00001	3.26%	0.29%	0.22%	0.0%	0.0%	0.0%

10^{-5} mile ~ 1.6cm

N

Γ/Γ'

	13	14	15	20	25	30
0.9	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
0.99	0.0%	0.02%	0.0%	0.01%	0.0%	0.0%
0.999	0.02%	0.0%	0.0%	0.0%	0.0%	0.0%
0.9999	0.48%	0.12%	0.03%	0.0%	0.0%	0.0%

26/42

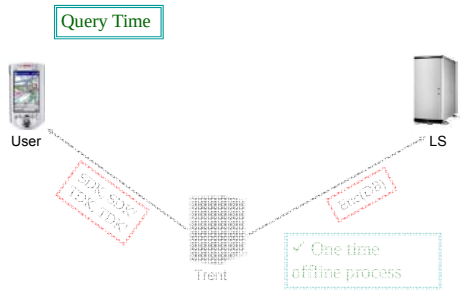
LS & External Adversary Collusion

We assume unmolested program execution on users' client devices that prevents adversaries from breaching into a client device

- Running code securely on an untrusted client is an open problem
- 100% utilization of server
  - Hard to map an H-value request to an external adversary's location
- Using SALT, makes it impossible for the attacker and LS to find the entire mapping

27/42

End to End Architecture

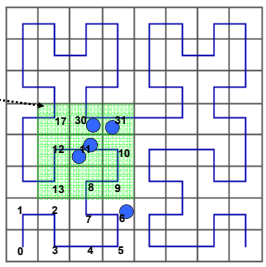


28/42

Range Queries

A window (range) query

Querying objects O such that ℓ(O) belongs to the set RS={8, 9, 10, 11, 12, 13, 17, 30, 31}



POI

29/42

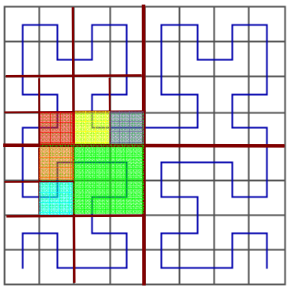
Steps to Answer a Range Query

The Hilbert space is recursively decomposed into each piece is fully contained in the range.

Result: maximal quad-tree blocks

Property: H-values inside a maximal block form a continuously increasing sequence.

Tsai et al. A strip-splitting-based optimal algorithm for decomposing a query window into maximal quadtree blocks, ICDE'04



30/42

Steps to Answer a Range Query

8-11, 17, 13, 12, 30, 31

Sort

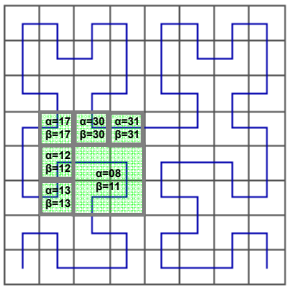
8-11, 12, 13, 17, 30, 31

Merge

8-11, 12-13, 17, 30-31

Each sequence is called a run

Chung et al. Space-filling approach for fast window query on compressed images, Transactions on Image Processing'00



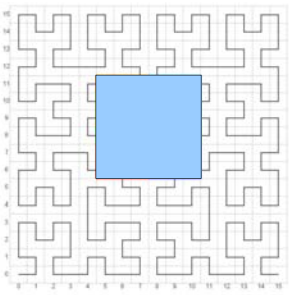
31/42

Example

A range query is decomposed Into its maximal quad tree blocks (each of the colored squares is a maximal quad tree block)

The final runs:
Each colored part is a single run (7 runs total)

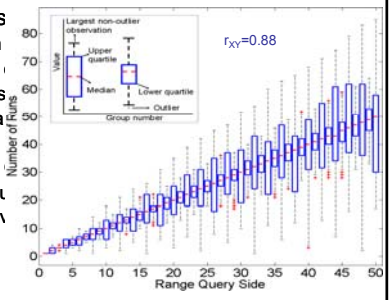
Range Query Result Set Is Exact but May Contain Excessive Objects



32/42

Privacy Aware Range Query Search

- Packing all runs
  - The server can
  - The cardinality
  - Number of runs
  - Server learns a |RS| points
- Query runs are
  - If each run is qu
  - r_{XY} (run length



33/42

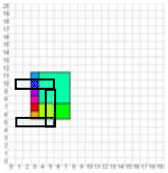
Algorithm Complexity

- Range algorithm takes $O(n_l \log T)$ time where $n_l = max(n_1,n_2)$ for a query of size $n_1 * n_2$ and $T = 2^N$ (N is the curve order).
  - $O(n_l)$ for decomposition
  - $O(n_l * N)$ for finding α and β
  - $O(n_l * \log n_l)$ for sorting sub runs
  - $O(n_l)$ for merging runs
- Search:
  - Alice performs quadtree decomposition on both curves and chooses the one with fewer runs and sends runs to LS
  - LS returns the encoded result set to Alice

34/42

Curves Translation & Range Search

- A range query maps into many runs
  - It is desirable to minimize the number of runs (quadtree blocks)
- Indexing the data with a second shifted curve can achieve this



35/42

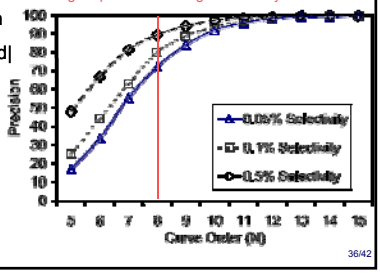
Range Query Processing

- Range queries are exact
- Include excessive objects
- Measuring precision

  |relevant|/|returned|

Higher precision for larger selectivity

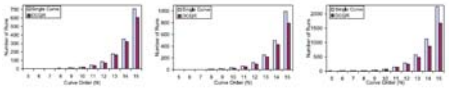
Precision reaches 100% for N≥13 for real-world data



36/42

Range Query Processing

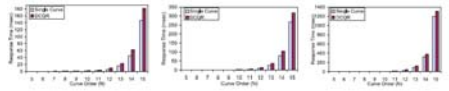
Average of 21% reduction in number of runs



(a) 0.05% Selectivity (b) 0.1% Selectivity (c) 0.5% Selectivity

Average number of runs linearly proportional to query side length

Number of Runs Vs. Hilbert Order for a Query with Different Selectivity.



(a) 0.05% selectivity (b) 0.01% selectivity (c) 0.5% selectivity

Marginal DCQR Overhead (around 6 ms on average)

Running Times Vs. Curve Order (N) for Different Query Selectivity.

37/42

Attacks on Cloaking and Anonymity

- Center of the cloaked region
- Single point of failure and attack
- Cloaking failure under certain distributions
- Availability of all user locations to LS in anonymity approaches
- Huge performance penalty for privacy-paranoid users.

38/42

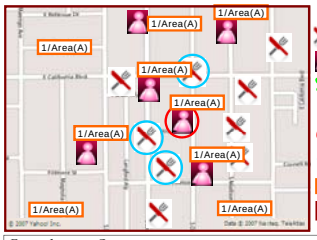
u-anonymity

Points of Interest o_i
Users u_i
Data S = $\{o_1, o_2, \ldots, o_n\}$
Users U = $\{u_1, u_2, \ldots, u_M\}$
Q = Query Point
Result Set RS
$P_Q(u_i)$

For each query Q:
Definition 1. u-anonymity: $P_Q(u_i) = 1/M$
$P_Q(u_i)$ is the probability that query Q is issued by a user u_i

39/42

a-anonymity



Points of Interest o_i
Users u_i
Data S = $\{o_1, o_2, \ldots, o_n\}$
Users U = $\{u_1, u_2, \ldots, u_M\}$
Q = Query Point
Result Set RS
$P'_Q(l_i)$
A

For each query Q:
Definition 2. a-anonymity: $P'_Q(l_i) = 1/area(A)$
$P'_Q(l_i)$ = probability that query Q is issued from any point inside A

40/42

Result Set Anonymity

Points of Interest o_i
Users u_i
Data S = $\{o_1, o_2, \ldots, o_n\}$
Users U = $\{u_1, u_2, \ldots, u_M\}$
Q = Query Point
Result Set RS
$p_Q(o_j)$

For each query Q:
Definition 3. Result set anonymity: $p_Q(o_j)$ = k/n for j = 1 … n and k=|RS|
$p_Q(o_j)$ =probability that o_j is a member of the result set for query Q

41/42