Location Services for Geographic Routing

Geographic Routing

- Three major components of geographic routing:
  - Location services (dissemination of location information)
  - Forwarding strategies
  - Recovery schemes

Problem

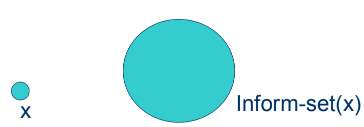
- Construct and maintain a location database.
- Who keep track of whom?
  - Some for Some
  - Some for All
  - All for Some
  - All for All

Location Database

- Distributed Information Structure
- Two major operations
  - Information Inform/Update (write)
  - Information Request/Query (read)
- Question
  - Inform set = ?
  - Query set = ?

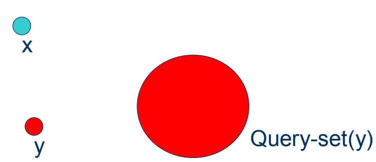
Inform Set

- X stores its location information in every node in Inform-set(x).



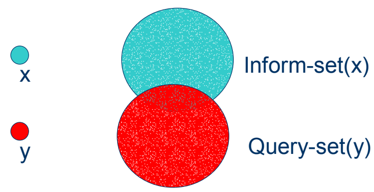
Query Set

- y consults nodes in Query-set(y) for location information.



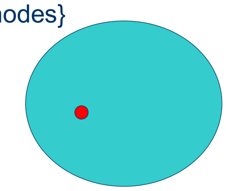
Requirement

Inform-set(x) ∩ Query-set(y) ≠ Φ for all x, y



A Brute-Force Scheme

- All for All
- Inform-set(x) = {all nodes}
- Query set(y) = {y}
- Example: DREAM

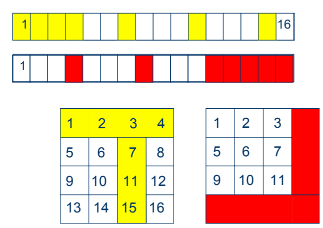


Quorum-Based Location Database: Basic Idea

- U = A set of nodes
- Quorum: any subset of U
- Quorum system: a collection of pair-wise intersecting quorums
- Inform-set(x) = any quorum
- Query set(y) = any quorum
- Design Issues: construction and maintenance of U and a quorum system

Example Quorum System

Quorum = one row + one column

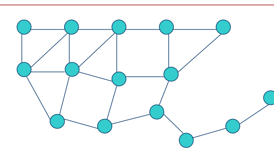


A Quorum-Based Location Service

- Zygmunt J. Haas and Ben Liang, *"Ad Hoc* Mobility Management With Uniform Quorum Systems," ACM/IEEE Trans. On Networking, April 1999.
- Ben Liang and Zygmunt J. Haas, "Virtual Backbone Generation and Maintenance in Ad Hoc Network Mobility Management," INFOCOM 2000.

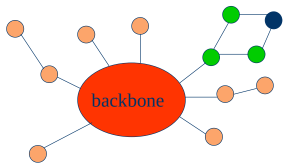
U = ?

- U = a set of nodes such that every node is within r-hops of U.
- Called r-virtual backbone.
- Called dominating set if r = 1.



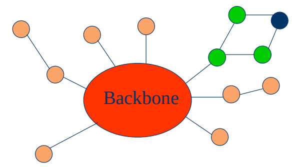
Backbone Generation

- Similar to domination set generation
- Nodes: Backbone, Bridge, Covered, Uncovered



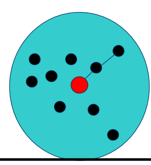
Constructing a backbone: basic idea

- Initially, Backbone contains a single node.
- Bridges keep entering Backbone until no more bridges.



Required for the algorithm

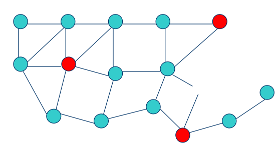
- r-zone: the neighborhood within r hops
- Every node maintains its r-zone
  - e.g. using the link-state algorithm.



Maintenance of Backbone

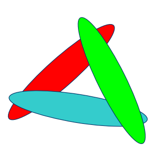
- If network topology changes

  Backbone changes



Quorum-Based Location Database

- U = {nodes in backbone}
- Quorum: a subset of U
- Quorum system: a collection of pair-wise intersecting quorums
- Inform-set(x) = any quorum
- Query set(y) = any quorum



Maintenance of Quorum System

- Backbone may change
- Quorum system may change
- Maintenance of quorum system is nontrivial

Conclusion

- Location services based on Quorum Systems seem too complicated.
- So?

  Probabilistic/Randomized Quorum Systems

Randomized Quorum Systems

- Malkhi et al., "Probabilistic Quorum Systems, " Information and Computation **170**, 184–206 (2001). Also, PODC 1997.
- Zygmunt J. Haas and Ben Liang, "Ad-Hoc Mobility Management with Randomized Database Groups," ICC 1999.
- Jiandong Li, Zygmunt, J. Haas, and Ben Liang, "Performance Analysis of Random Database Group Scheme for Mobility Management in Ad hoc Networks," ICC 2003.
- S. Bhattacharya, "Randomized Location Service in Mobile Ad Hoc Networks," MSWiM'03.
- H. Lee, J.L. Welch, N.H. Vaidya, "Location Tracking Using Quorums in Mobile Ad Hoc Networks," Aug 2003.

Basic Result [Malkhi et al., 1997]

- U = a given set of nodes.
- $0 \le p \le 1$.
- Random quorum of size k: any randomly selected subset of U of size k.
- Given U and p, it is possible to choose k such that for any two random quorums, A and B, of size k, $\text{Prob}(A \cap B \neq \Phi) \ge p$.

Random-Quorum-Based Location Database [Haas and Liang, ICC 1999]

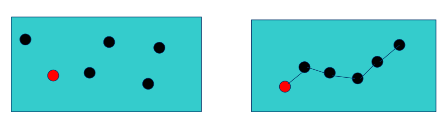
- U = {nodes in a virtual backbone}
  - These nodes serve as location servers
- Inform set(x) = any random quorum of size k.
- Query set(x) = any random quorum of size k.
- Access strategy: repeat queries until success.
- Performance studies

Performance Analysis [Li et al, ICC 2003]

- Update cost
- Query cost
- Total cost
  - Update rate (λ_u)
  - Query rate (λ_q)
  - How does total cost depend on (λ_q / λ_u) and quorum size?
- Optimum quorum sizes
- Different query strategies

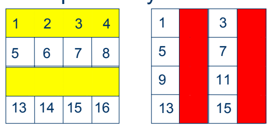
Randomized Location Service [S. Bhattacharya, MSWiM'03]

- Two schemes for constructing a quorum:
  - 1: randomly choose k nodes
  - 2: randomly choose a path of k nodes
- Similar accuracy for dense networks.
- Quorum size: depends on desired accuracy.
- Lower communication cost and query delay time for scheme 2.



Location Tracking Using Quorums [Lee, Welch, Vaidya, Aug 2003]

- Comparing three kinds of quorum systems
- U = {1, 2, …, n}
- Biquorum system
  - Update quorums
  - Query quorums
  - Update quorum ∩ Query quorum ≠ Φ
- Traditional quorum system
- Random quorum system
- Random-quorum-based performs best



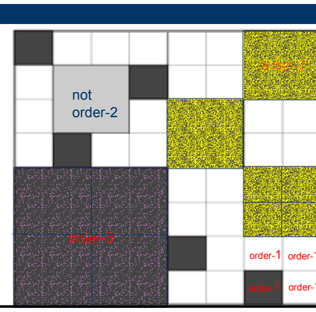
GLS: Grid location Service

- "A Scalable Location Service for Geographic Ad hoc Routing"
- J. Li, J. Jannotti, D. Couto, D. Karger, R. Morris
- MIT
- Mobicom 2000

Model

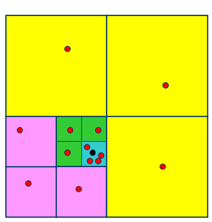
- Dense deployment of nodes in a rectangular area
- Nodes have unique ID
- Nodes know their own location information

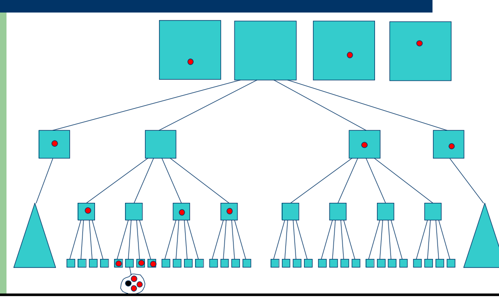
Geographic Hierarchy of the Network



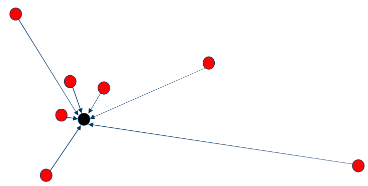
Location Servers

- Every node maintains its current location in the following location servers:
  - Every node in the same order-1 square
  - One node in every sibling order-1 square
  - One node in every sibling order-2 square
  - One node in every sibling order-3 square, etc.



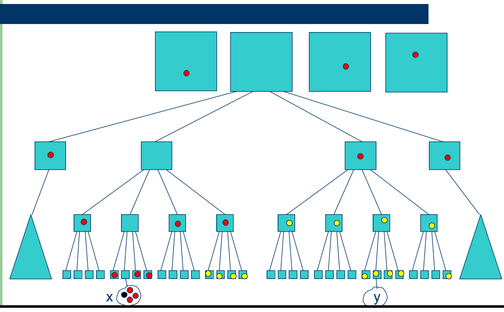


Who knows whom?



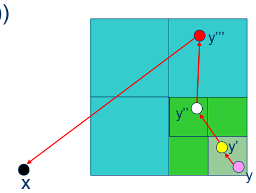
Selecting Location Servers

- If node x has a server in an area A, it is:
  f(x, A) = the node in A whose ID is circularly closest to x from the above.



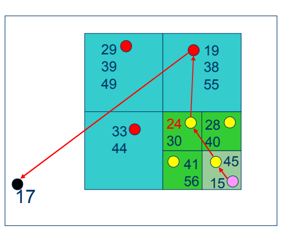
Query: where is x?

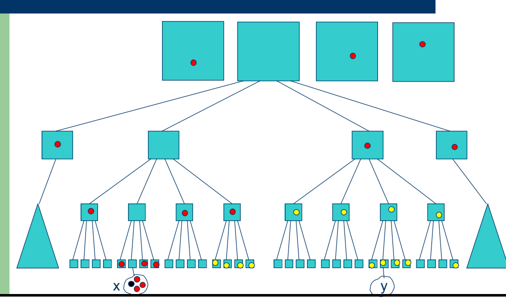
- y asks y' =f(x,order-1(y))
- y' asks y''=f(x,order-2(y'))
- y'' sks y'''=f(x,order-3(y''))



- Does y know y'?
- Does y' know y''?
- Does y'' know y'''?
- Answer: yes

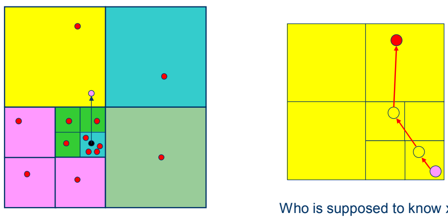
Query: where is x?





How does x update Location Information?

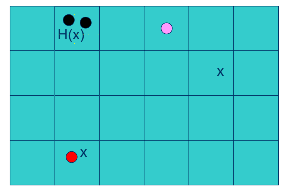
- x sends an update message to each area.



Who is supposed to know x?

Home-Region Based Location Service

- Each node x is assigned a home region H(x). Every node in H(x) serves as x's location server.

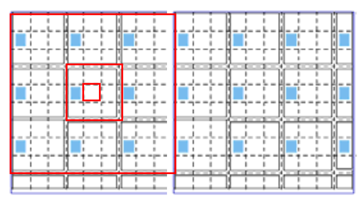


Combining GLS with Home-Region

- **"A Scalable Location Management Scheme in Mobile Ad-hoc Networks," LCN 2001**
- **"SLALoM: A Scalable Location Management Scheme for Large Mobile Ad-hoc Networks,"** WCNC 2002

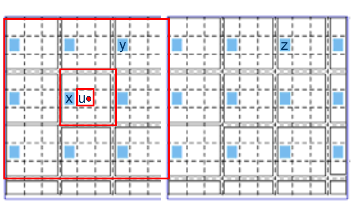
SLALoM

- Unit square, order-1 square, order-2 square, etc.



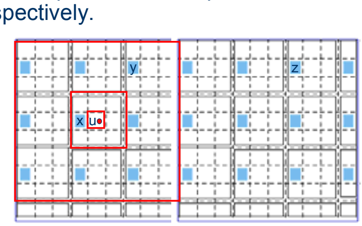
SLALoM

- Each node has a home region (blue shade) in each order-1 square.



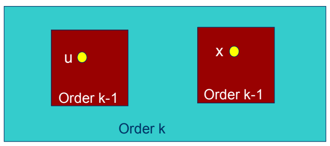
SLALoM

- x, y, z know of u's whereabouts to the accuracy of unit square, oder-1 square, order-2 square, respectively.



SLALoM

- x: a node in one of u's home regions
- If u and x are in the same order-k square, but different order-(k-1) squares, then x knows which order-(k-1) square contain u.



範文選讀 之一

見賢思齊
見不賢則內自省

是賢？ 是不賢？

- Jiandong Li, Z.J. Haas, and Ben Liang, "Performance Analysis of Random Database Group Scheme for Mobility Management in Ad hoc Networks," ICC 2003.

Random-Quorum-Based Location Database [Haas and Liang, ICC 1999]

- U = {nodes in a virtual backbone}
  - These nodes serve as location servers
- Inform set(x) = any random quorum of size k.
- Query set(x) = any random quorum of size k.
- Access strategy: repeat queries until success.
- Performance studies (by simulation)

Purpose of the Li-Haas-Liang paper

- To analyze the cost of the RDG scheme
- To propose different update and query schemes

The Randomized Database Group (RDG) Scheme under Analysis

- U = {all nodes in the ad hoc networks}
  - Every node is a location server
  - |U| = N
- Inform set(x) = any random quorum of size K.
- Query set(x) = any random quorum of size Q.
- Access strategy: repeat queries until success.

Question 1: Expected Query Cost?

- What's the expected total cost of Queries until success?
  - Denoted as C_Q
- Query: sending a message to every node in the Query set.
- query: sending a message to a single node.

- C_q = cost of query to a single server
- C_Q = Q * expected # of Queries until success * C_q

E(X) = Expected # of Queries until Success

- X = number of Queries until success
- E(X) = ?
- P(X=k)

  = P(failure)*…*P(failure)*P(success)

  = P(0,Q,K,N)*P(0,Q,K,N-Q)*P(0,Q,K,N-2Q)

  * … * (1-P(0,Q,K, N-(k-1)Q))
- P(0,Q,K,N) = C(N-K,Q) / C(N,Q)

E(X) = ?

$$E(X) = 1\cdot P(X=1) + 2\cdot P(X=2) + \ldots + (L+1)\cdot P(X=L+1)$$
$$= 1 + \sum_{l=1}^{L} \frac{(N-lQ)(N-lQ-1)\ldots(N-lQ-K+1)}{N(N-1)\ldots(N-K+1)}, \quad (6)$$

where L is the integer part of *(N-K)/Q*.

- Question is answered.
- Conclusion: C_Q is smallest when Q = 1.

Question 2: Total Update and Query Cost (per unit time)?

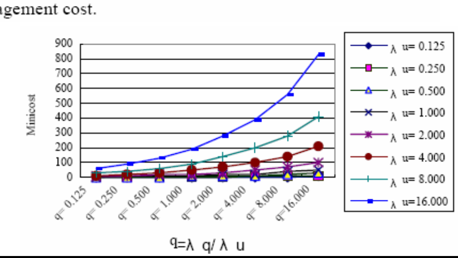
- Total cost (per unit time)
  - Update rate (λ_u)
  - Query rate (λ_q)
  - How does total cost depend on (λ_q / λ_u) and quorum size?
  - What quorum sizes would minimize the total cast?

$$C_{total} = \Lambda_u K C_u + \Lambda_q Q E(X) C_q = \Lambda_u (C_u K + (\Lambda_q/\Lambda_u) C_q Q E(X)) \quad (8)$$

$$C_{total} = \Lambda_u K C_u + \Lambda_q Q E(X) C_q = \Lambda_u (C_u K + (\Lambda_q/\Lambda_u) C_q Q E(X)) \quad (8)$$

From Equation (8) and numerical results, we learn that the minimum total cost is dominated by the average query cost, and the contribution of the update cost to the minimum total cost can be neglected for $\Lambda_q/\Lambda_u > 4$ and $K<5$, and that the minimum total cost is dominated by the update cost, and the contribution of the query cost to the minimum total cost can be neglected for $\Lambda_q/\Lambda_u < 4$ and $K>9$. The total cost asymptotically tends to be linear in K for $\Lambda_q/\Lambda_u < 4$.

The minimum total cost for different $q = \lambda_q / \lambda_u$ and $N=40$ is shown in Fig. 4. From the figure, it is clear that the update rate dominates the minimum cost. If the network is highly dynamic, the update rate should be set to the same order of magnitude of the query rate to decrease the mobility management cost.



Question 3

- Wanted:
  - The first Query succeeds with high probability.
- What quorum sizes, K and Q, would guarantee this, while minimizing the cost?
- What cost?

- Cost(N,K,Q) = cost of a Query
  = K*Cq
- Prob(N,K,Q) = prob(success)
  = 1 – C(N,N-K)/C(N,Q)
- Problem: Minimize Cost(N,K,Q) subject to Prob(N,K,Q) ≥ a given p.

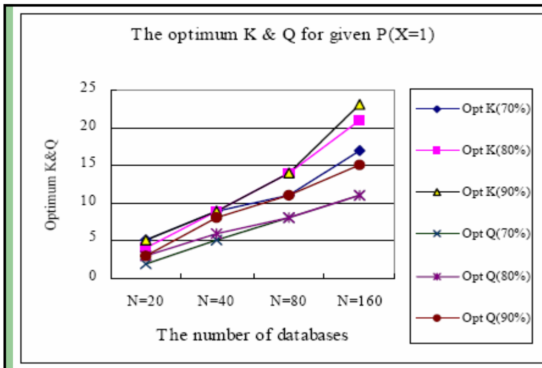


Fig. 5 Summary of the optimal values of K and Q for different N

Various query strategies

Strategy A: The first query uses a group size of *Q*; if the query is not successful, the second query uses a group size of *N-K-Q+1*. In this case, the maximum delay is two queries.

Strategy B: The first *n* queries use the same query group size of *Q*; if none is successful, the final query uses a group size of *N-K-n*Q+1*. In this, we need at most n+1 queries. (This is a generalization of the *Strategy A*).

Strategy C: The query group size is progressively increased; for example, the query group size could follow the sequence: *Q, 2Q, 3Q, ..., (N-K-n*(n+1)*Q/2+1)*.

Strategy D: The lowest-cost query group size of *Q=1*is used.

Another strategy

- Strategy E: use the optimal Q and K obtained from Figure 5.

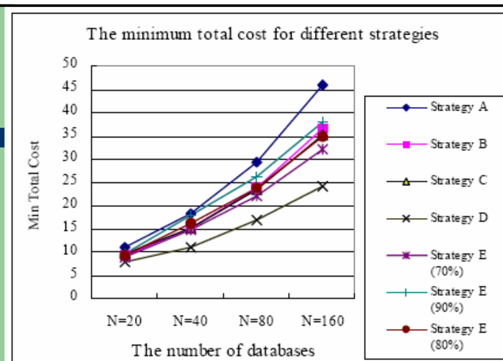


Fig. 6 The cost comparison of the different search strategy