Multi-level Hashing for Peer-to-Peer System in Wireless Ad Hoc Environment

Dewan Tanvir Ahmed and Shervin Shirmohammadi
Distributed and Collaborative Virtual Environments Research Laboratory
School of Information Technology & Engineering
University of Ottawa, Canada

Outline

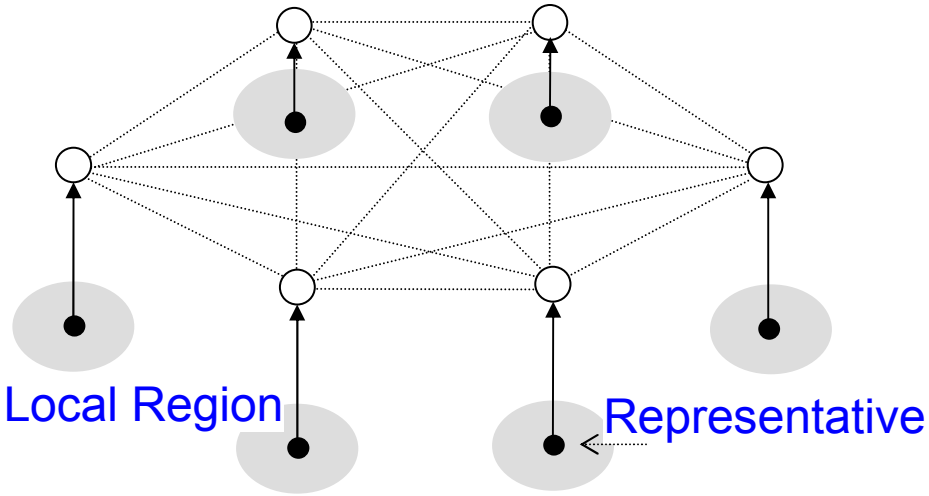
- Motivation
- Architecture
  - Node join
  - Content publishing
  - Content discovering
  - Important issues
- Simulation results
- Analysis & Conclusion

Motivation

- Heart of the p2p protocols
  - Smart content distribution mechanism
  - Efficient content discovery process
- Extra challenges in wireless ad hoc network
  - Routing cost
  - Resource limitations
- In this paper, our focus on to check
  - The effectiveness of multilevel hashing

Architecture

- The entire system consists of a set of home regions
- Peers in close proximity form home region



- Home region representative
  - plays a supporting role
  - functions as
    - **a rendezvous point for incoming requests and**
    - **provides useful information to local peers**
  - change over the time
  - identified by *Home-Region-ID*

Node identification

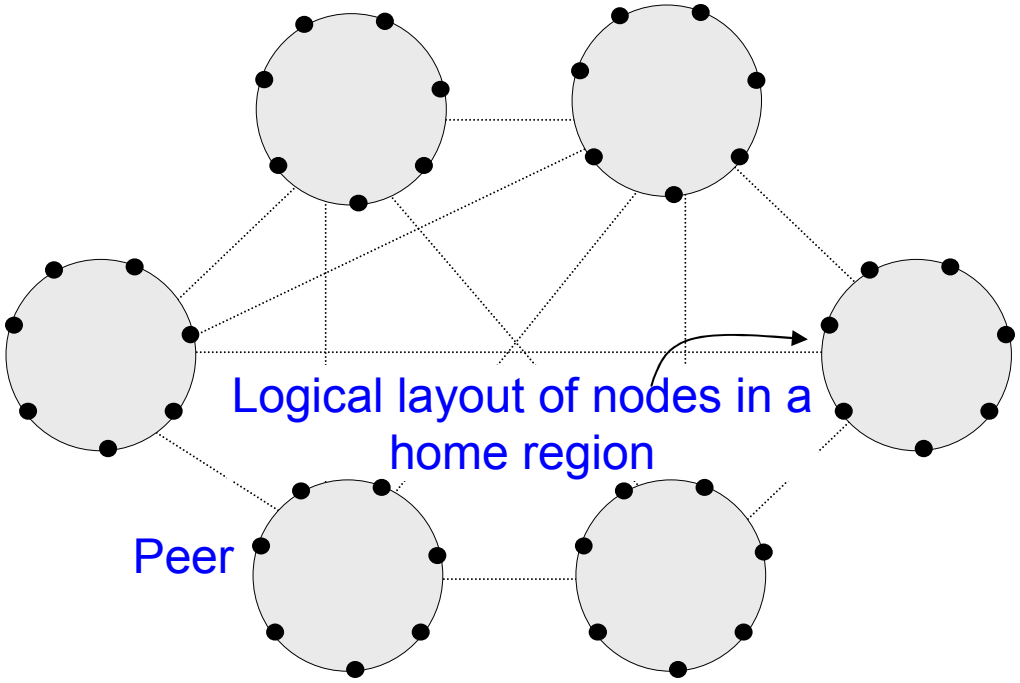
- Two keys::
  - *<Home-Region-ID, Node-ID>*
  - *Home-Region-ID*
    - **depends on its location**
    - **may change over time.**
  - *Node-ID = H_l(Node-Address)*

Learning Representative

- Periodic broadcast
  - home region representative
- Controlled flooding
  - not flooded to the entire system
  - Stop forwarding request when it has the answer
  - Keep minimum hops away response
- Peer gets the identity of its successor and predecessor from this representative

Logical Structure

- Chord like structure

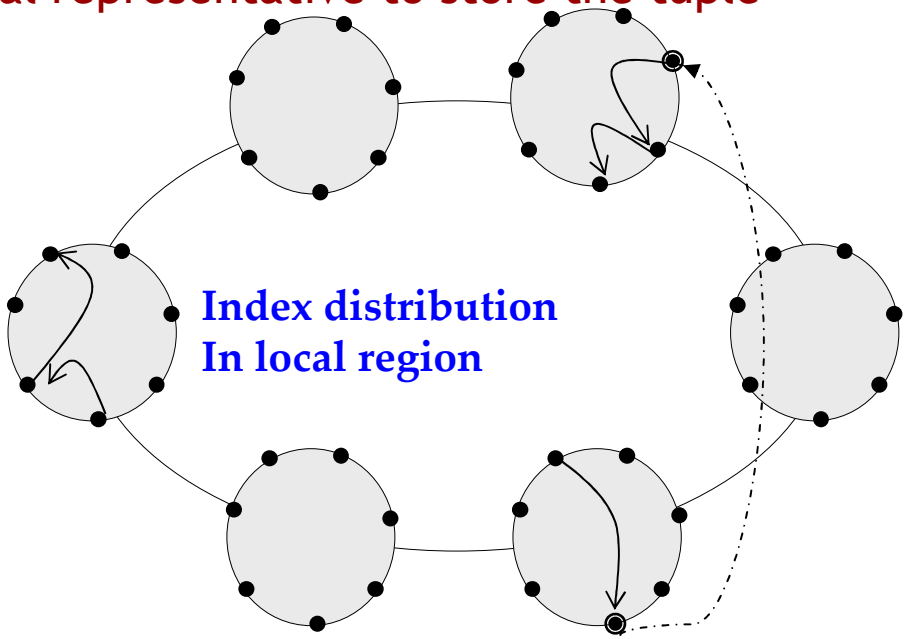


Content Publishing

- Contents & nodes
  - mapped to the same address space
- *Key*: an n-bit number
- two hash functions providing two keys
  - One key
    - **map the content name to a home region**
    - $key_r = H_r(Content\text{-}Name)$
  - Second key
    - **determine the successor in a particular region**
    - $key_s = H_l(Content\text{-}Name)$

Content Publishing

- The content publisher
  - builds a tuple <*Content-Name, Owner-Address*>
  - asks local representative to store the tuple



Index distribution in foreign region

- **Representative**

Content Discovery

- To discover content peer hashes *Content-Name*
  - $key_r = H_r(Content\text{-}Name)$
  - it represents the home region
    - **where the content information is stored**
- How does peer get id of foreign representative?
  - with the help of local representative
- What is the role of second key, key_s?
  - To discover the successor
    - **when the content discovering process has been shifted to the appropriate region**
  - chord like content searching process

Content Discovery (cont..)

- Chord uses Finger Table (FT)
  - to speed up searching process
- Let hash key is 128-bit long
  - address space: 3.4e38
  - Say, we have 1020 peers in the system
    - **average gap: 3.4e18 between two active nodes**
  - most entries of the FT point to the same successor
  - actually search continues in linear way
- Big issue in Ad hoc Networks
  - as routing is expensive

Content Discovery (cont..)

- The FT has
  - m entries 0 to m-1
- Divided into two halves
  - *0* to $\lfloor m/2 \rfloor$ and
  - $\lceil m/2 \rceil$ to (*m-1*)
- Each entry has two fields
  - start and the physical address of *successor(start)*

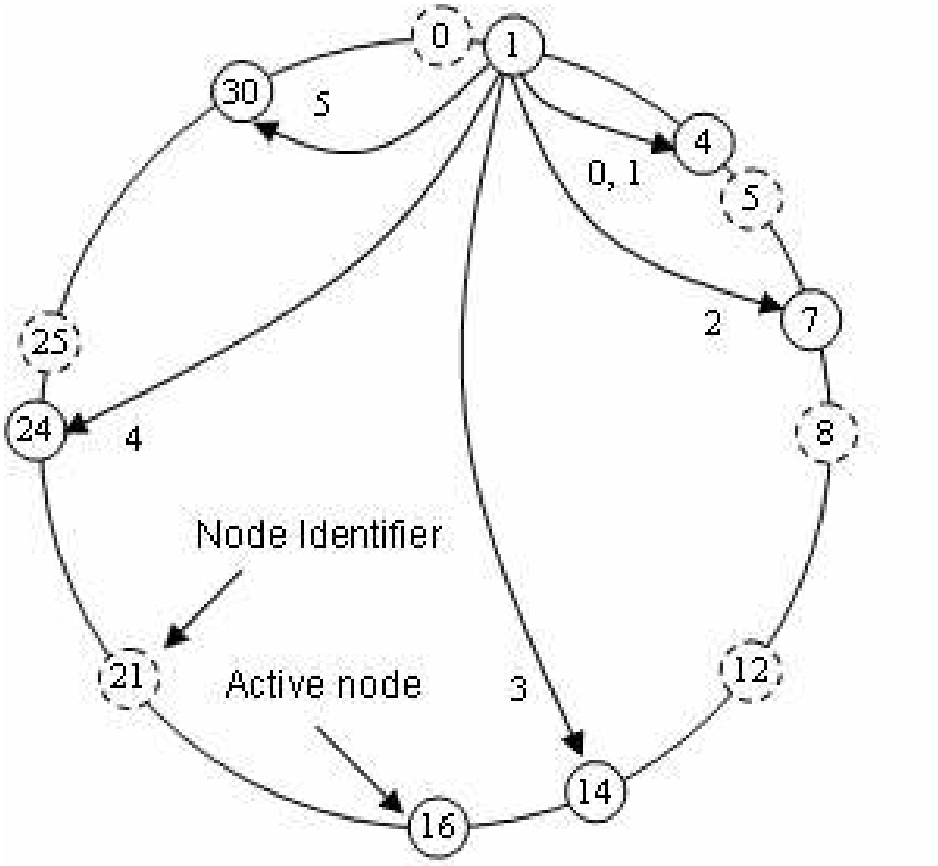
Start	Successor Address

Content Discovery (cont..)

- First half:
  - for entry *i* at node *k*
    - ***start* = *k* + 2^i *mod* 2^m, where *0* $\leq$ i < $\lfloor m/2 \rfloor$**
    - ***Physical address of successor (start*[*i*])**
  - first half table covers *k* to *k*+$2^{\lfloor m/2 \rfloor - 1}$ address space
- Second half:
  - define *step-size* as a big number to jump more rapidly
  - The entries are incremented by the step size starting from *k*+$2^{\lfloor m/2 \rfloor - 1}$

Content Discovery (cont..)

- Example



Node 1's Finger Table (Step size 8)	
Start	Physical Address of start
2	4
3	5
5	7
13	14
21	24
29	30

Content Discovery (cont..)

- Another approach
  - the start and the physical address of the successor
    - ***start = (k + a^i) mod 2^n ,where m/2 $\leq$ i< m***
    - ***Physical address of successor (start[i])***
  - *a* is a constant and we have n-bit hash value
  - possible choice for '*a*' is the index of finger table
  - it increase at a rate of i^i [1^1, 2^2, 3^3…]

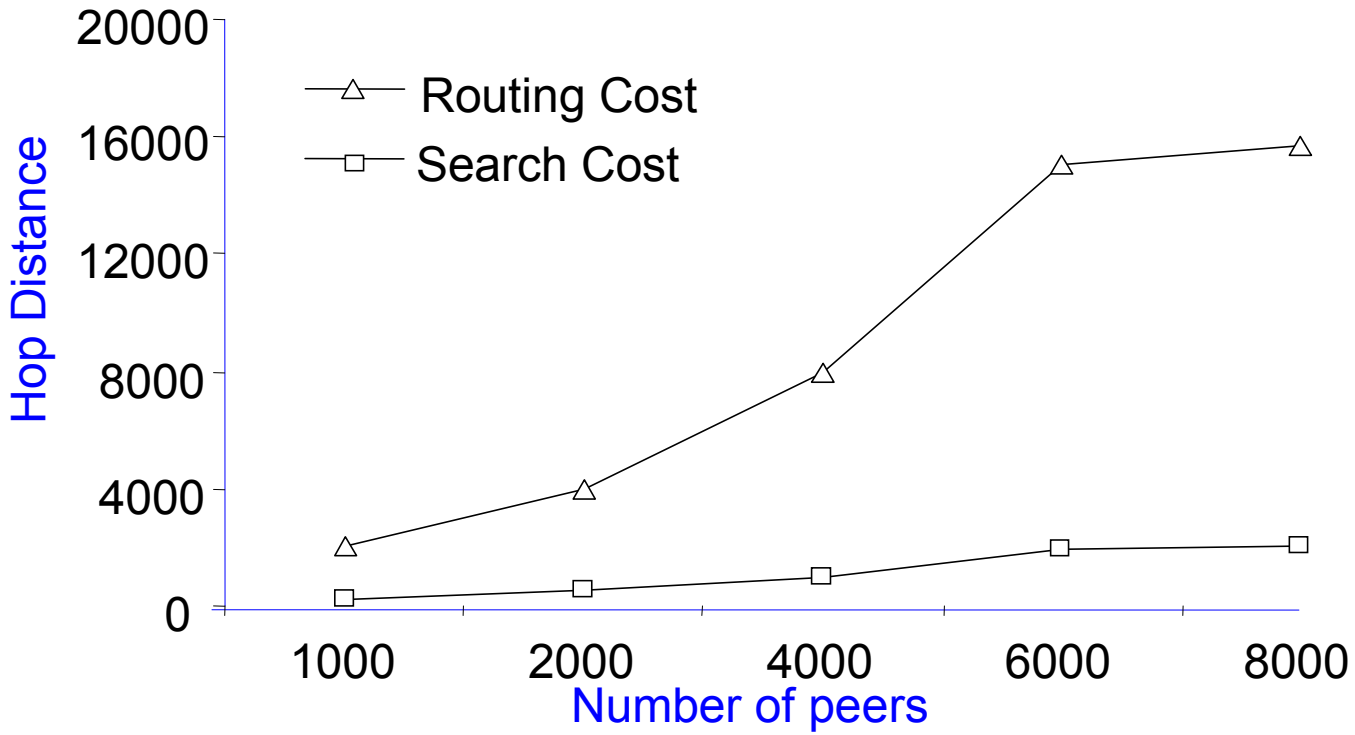
Simulation results

- Simulation setup

Parameter	Setup - 1	Setup - 2
Deployment area	100 × 100 square unit	10000 × 10000 square unit
Node's velocity	4 unit per second	4 unit per second
Content per node	3	3
Routing algorithm	Greedy Routing Algorithm	Greedy Routing Algorithm
Hash function	Message Digest 5	Modulo (32-bit Number)

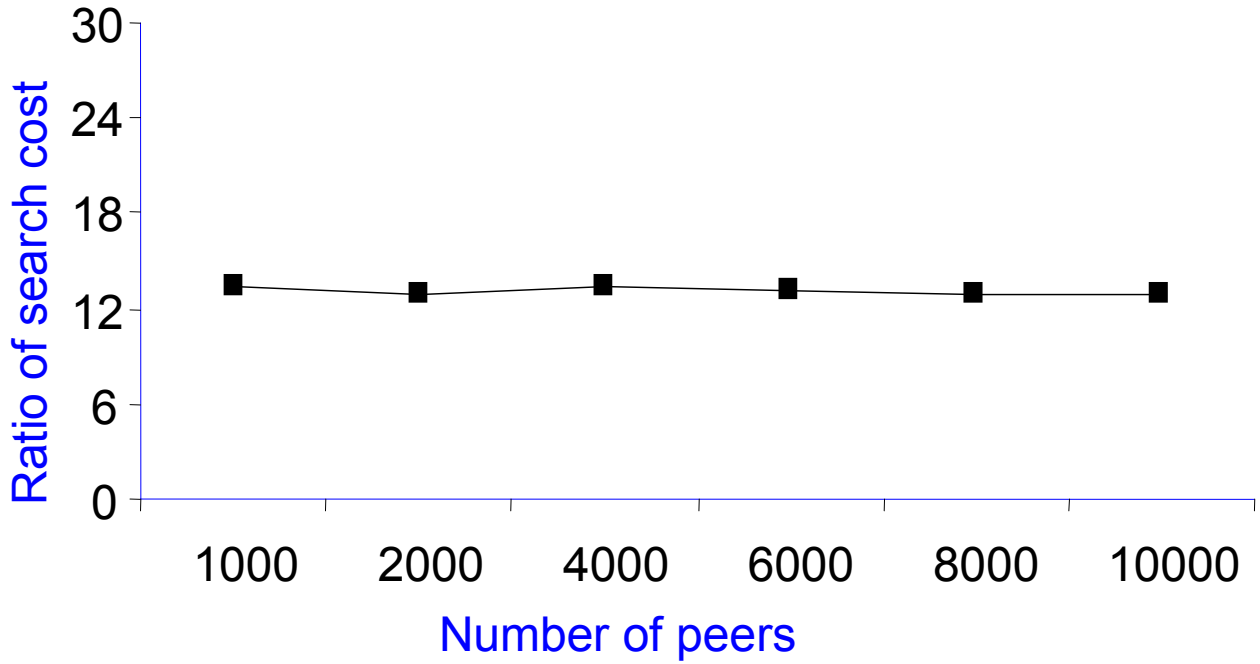
Simulation results

- Chord's performance in Mobile p2p



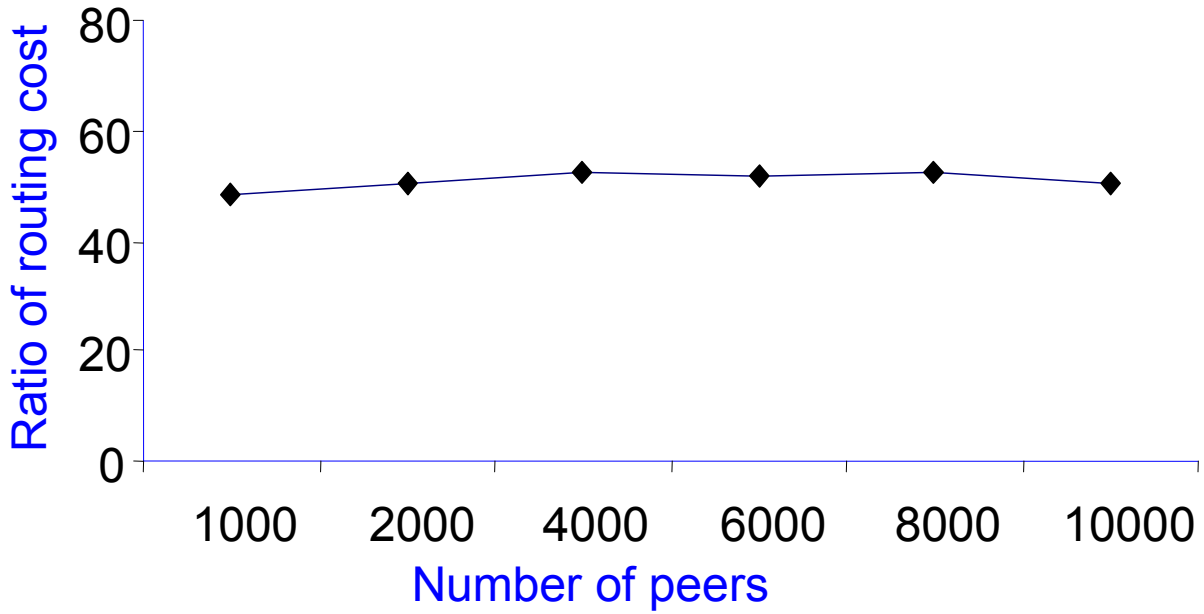
Simulation results (cont..)

- Improvement of search cost due to two level-hashing



Simulation results (cont..)

- Improvement of routing overhead due to two level-hashing



Conclusion

- We present a two-level hashing scheme for wireless ad hoc peer-to-peer systems
- Performance improvement due to multi-level hashing
  - in terms of search cost and routing overhead
- Modified finger tables works well when the number of peers is too small as compared to the hash address space

Thank You!