Plaxton Routing

- From a peer - to - peer network point of view

Lars P. Wederhake

Ferienakademie im Sarntal 2008
FAU Erlangen-Nürnberg, TU München, Uni Stuttgart

September 2008

Overview

The Basic Problem...

We want to:

- Share content with others
- Make content unavailable to others
- Search for content

→ And that in fact as fast as possible. . . but . . .

Desirable Properties of a P2P - Net

There's more than mere performance

When evaluating a P2P - Net it is a good idea to consider the following criteria:

- Scalability
- Maintenance
- Reliability
- Security
- Anonymity

Overview

Origin and original intention
Part I

The W-Questions:

- Who: C. Greg Plaxton, Rajmohan Rajaraman, Andrea W. Richa
- Where: @ University of Texas in Austin
- What: Accessing Nearby Copies of Replicated Objects in a Distributed Environment
- When: April 1997

Origin and original intention

Part II

- Why:
  - (i) High speed networks,
  - (ii) large numbers of geographically dispersed computers
  - → Cooperation & Sharing Content
  - (iii) Rapidly growing demand of users
    - . . . overloading network resources
  - → Main goal: Efficient usage of network resources
  - → Not intention: A P2P-Net

Overview

Overlay Network

PR defines a routing mechanism and a virtual network structure:

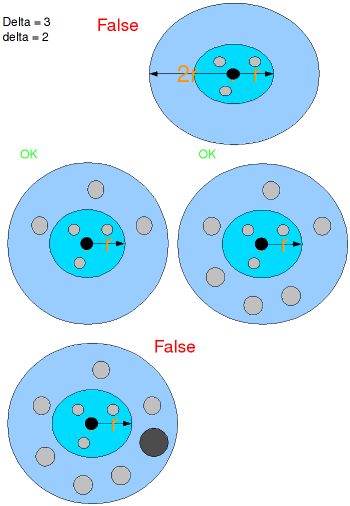
- A virtual network with
  - Neighbor Concept
  - Messaging via TCP/IP
  - Routing between nodes
    - on application level
    - Nodes usally are no routers in the underlay network

The Underlying Network

To make their network model work:

- Cost function $c : N \rightarrow R^{+}, c(1) = 1$ non-decreasing
- Symmetry: $c(u, v) = c(v, u)$
- Triangle inequaltiy: $c(u, \omega) \leq c(u, v) + c(v, \omega)$
- ball with radius r around u is $M(u, r)$
- Regularity of metric:
  $min\{\delta |M(u, r)|, n\} \leq |M(u, 2r)| \leq \{\Delta |M(u, r)|\}$

Regularity of metric: Visulization

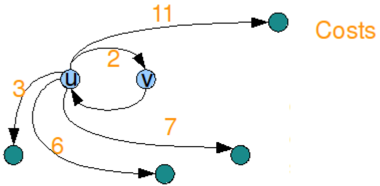


Overview on the Plaxton Model
Part I

The Plaxton Mesh is a distributed data structure with these properties

- **Naming:** random fixed length and unique bit-sequence. Typically hash of host/object name → names independent of location and semanctics
- Structuring: Each node owns a
  - **neighbor list** which is separated in a
    - (i,j)- primary neighbor
    - (i,j) secondary neighbors
    - reverse neighbors
  - **pointer list** of entries like:
    - (A, Y, k), A: some object, Y: node Y, k: costs on accessing A on Y.
  - Tree structure: every object is rooted at node with the bit-string nearest to object's one.

Neighconcept: Visualization



d = 4
(# sec - neigh.)

V is primary neighbor of u
and u is reverse neighbor of v
all others are secondary neigh.
Of u. Besides, if u is an i-leave of w,
then v would be and i+1-leave of w

Overview

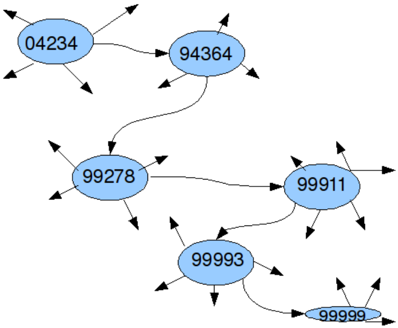
1. Introduction
   - Motivation for studying PR
   - The Background of PR
2. General Ideas and Concepts
   - Plaxton Mesh under the Overlay
   - The Routing Algorithm
   - Content Related Operations
3. Results
   - Advantages and Drawbacks of PR

The Routing Procedure

- Message M
- Destination address D
- A Plaxton mesh routes M to the node whose name is numerically closest to D.

Routing illustrated

5 hex digits Namespace: 2^{20} Routing from 04234 to 99999



- Each node has a neighbor map
- Multiple levels $log(n)$
- level i - matching prefix entries for i digits
- Number of entries per level = the ID base = b
- $\rightarrow$ Table Size: $b \cdot log(n)$
- At each entry the closest prefix matching node

Routing Mechanism
Part II

- Route to node x e.g.: ($A5F42$)
- Shared prefix i
- Look at level $i + 1$
- Match the next digit in destination
- Send message

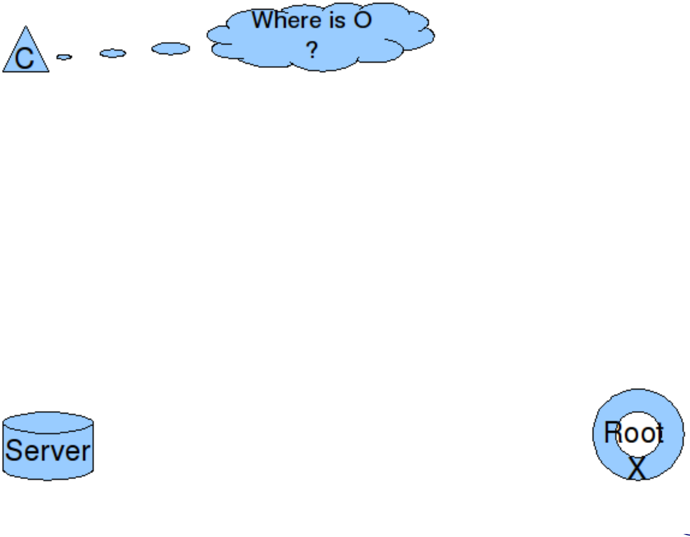
0642	x042	xx02	xxx0
1642	x142	xx12	xxx1
2642	x242	xx22	xxx2
3642	x342	xx32	xxx3
4642	x442	xx42	xxx4
5642	x542	xx52	xxx5
6642	x642	xx62	xxx6
7642	x742	xx72	xxx7

Overview

H(O) := Hash of an object O S := Host of file (Server)

- A client C wants to find object O.

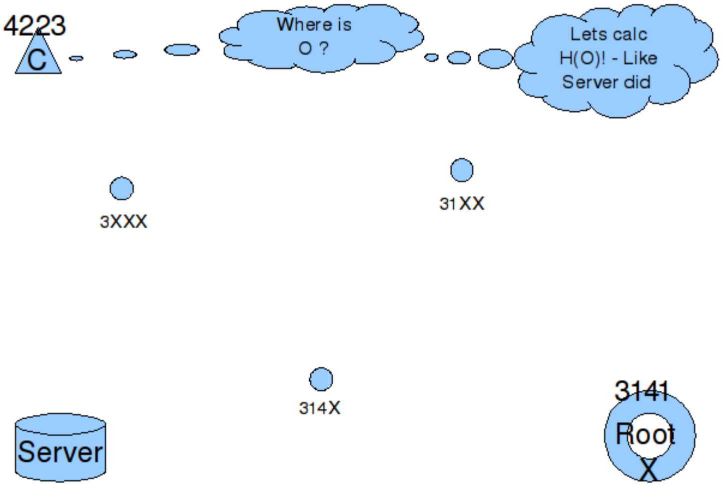
Accessing Visulization



H(O) := Hash of an object O S := Host of file (Server)

- A client C wants to find object O.
- Evaluates H(O), using same algorithm as the server

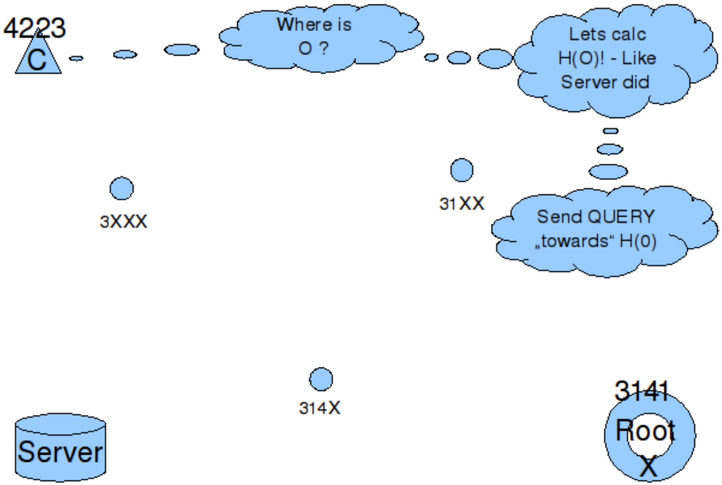
Accessing Visulization



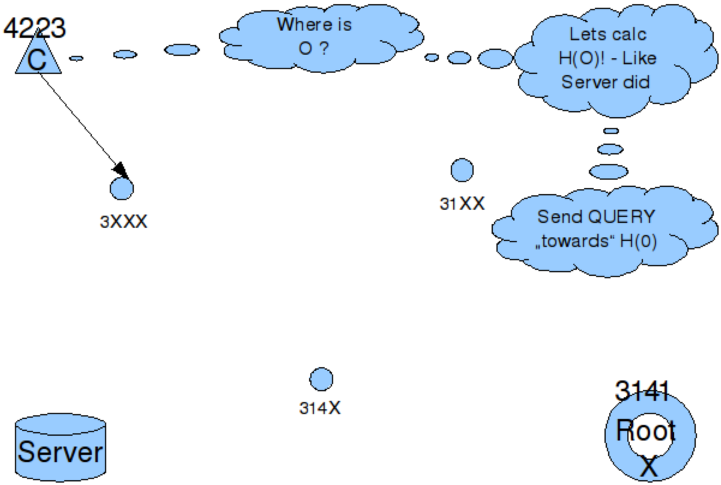
H(O) := Hash of an object O S := Host of file (Server)

- A client C wants to find object O.
- Evaluates H(O), using same algorithm as the server
- Sends a QUERY message to H(O)

Accessing Visulization



Accessinging Visulization



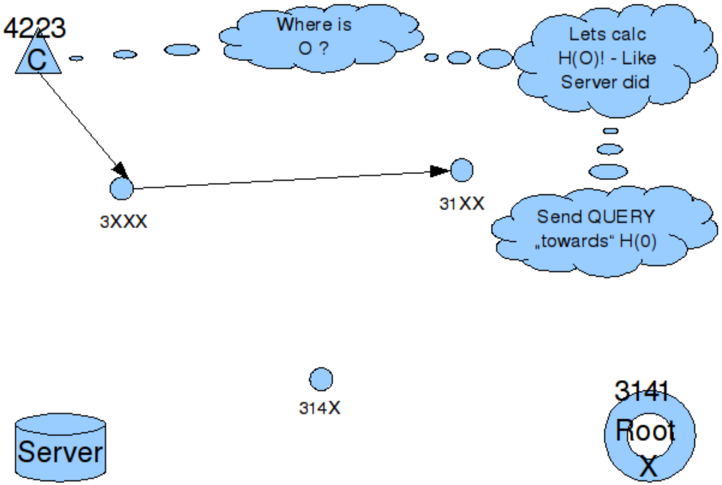
Content Accessing

Locating

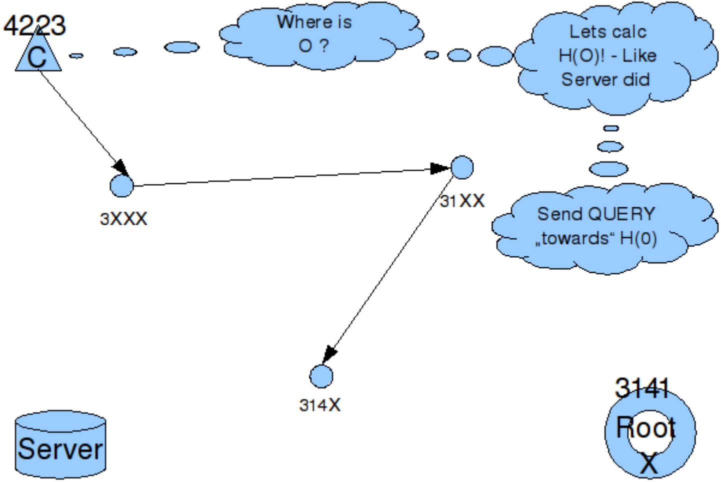
H(O) := Hash of an object O S := Host of file (Server)

- A client C wants to find object O.
- Evaluates H(O), using same algorithm as the server
- Sends a QUERY message to H(O)
- Forwarded to X, the object root.

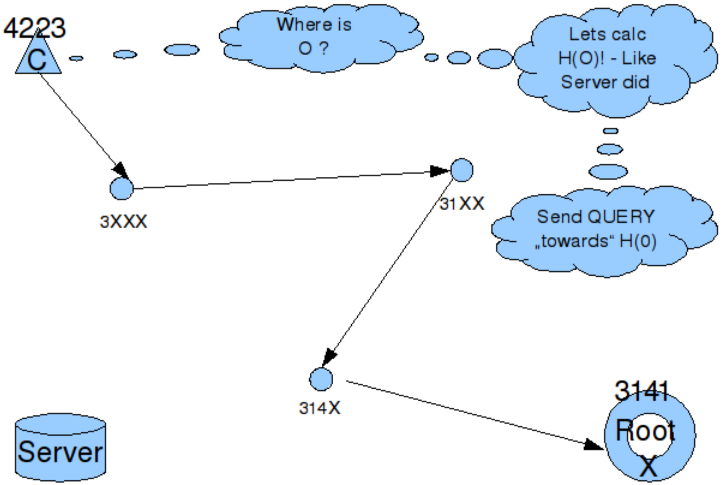
Accessing Visulization



Accessing Visulization



Accessing Visulization



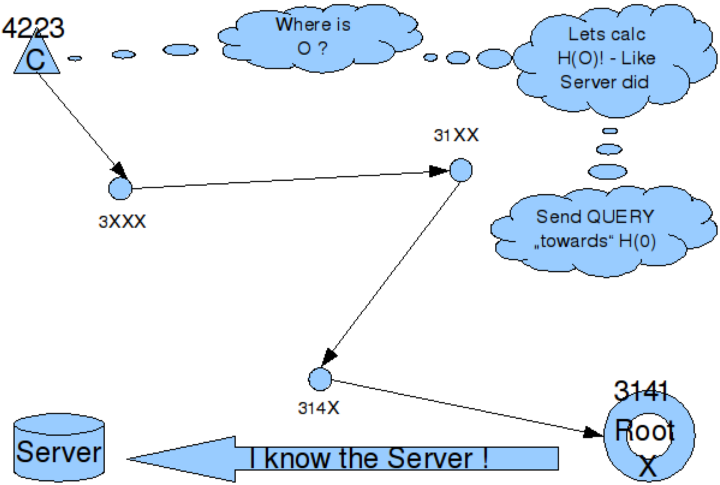
Content Accessing

Locating

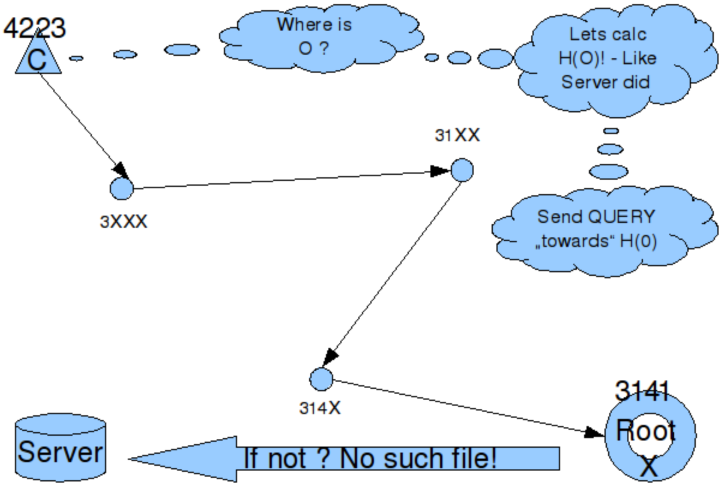
H(O) := Hash of an object O S := Host of file (Server)

- The object root forwards the message to S.

Accessing Visulization



Accessing Visulization



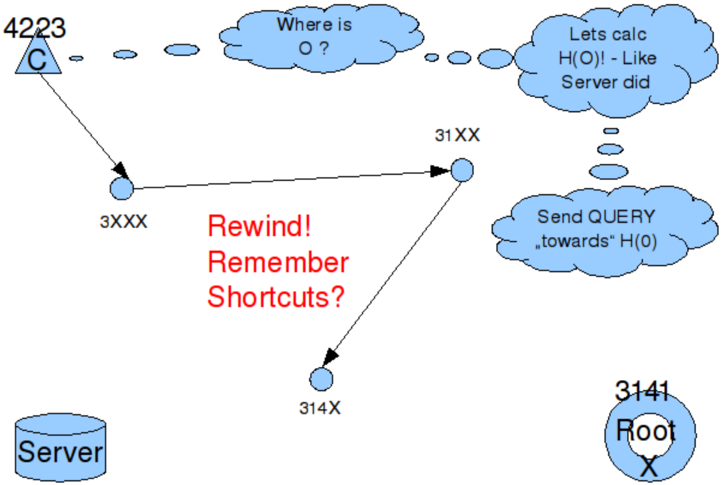
Content Accessing

Locating

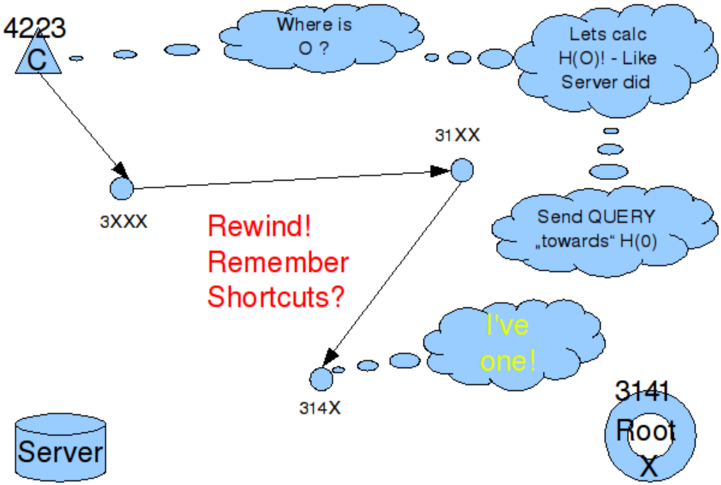
H(O) := Hash of an object O S := Host of file (Server)

- Shortcut - Nodes in the tree of O also know location of O
- If QUERY message reaches one of these nodes, it is forwarded directly to S.

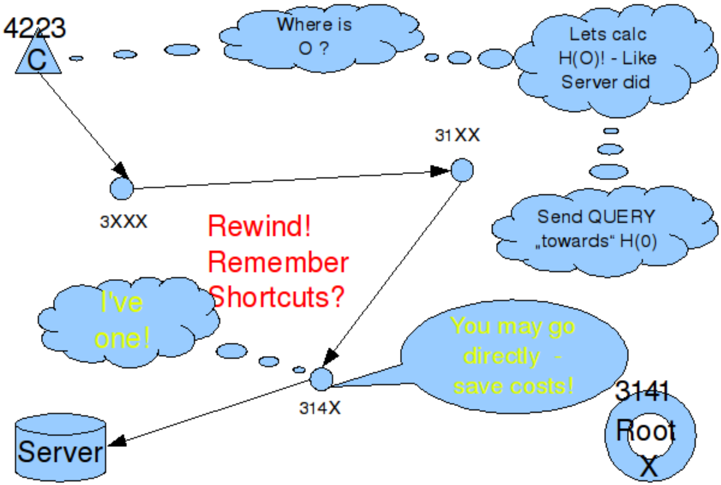
Accessing Visulization



Accessing Visulization



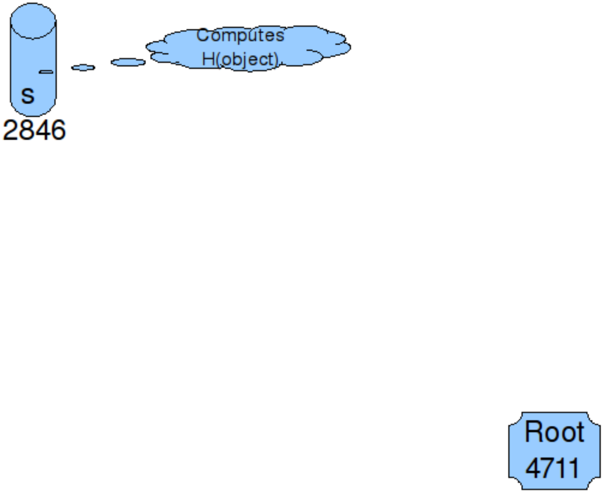
Accessing Visulization



H(O) := Hash of an object O S := Host of file (Server)

- Server S wants to publish object O
- S computes H(O), the object ID of O (hash of the name).

Publishing Visulization

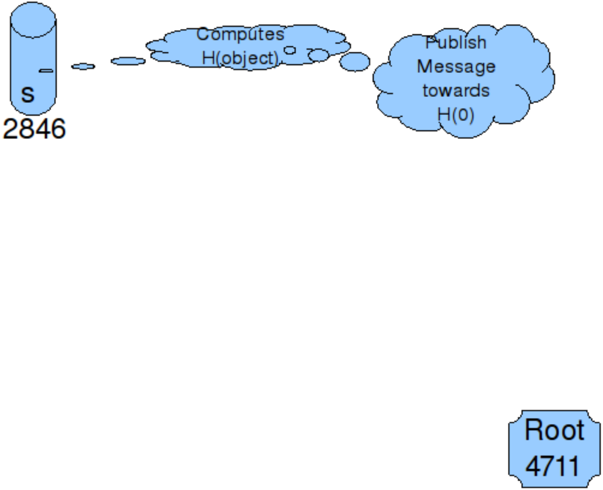


Publishing & Sharing Content

Exemplifying the algorithm

- Server S wants to publish object O
- S computes H(O), the object ID of O (hash of the name).
- S sends a PUBLISH message to node H(O).

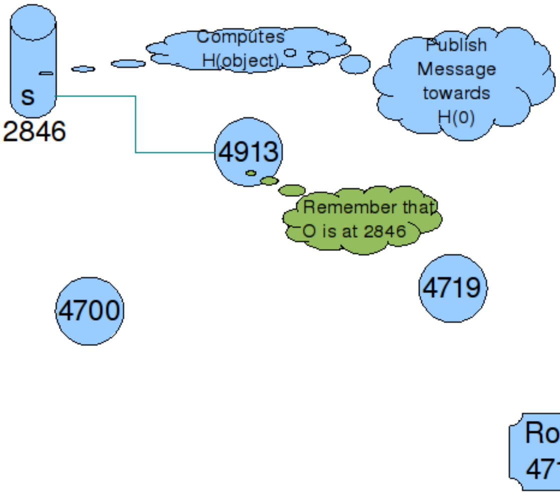
Publishing Visulization



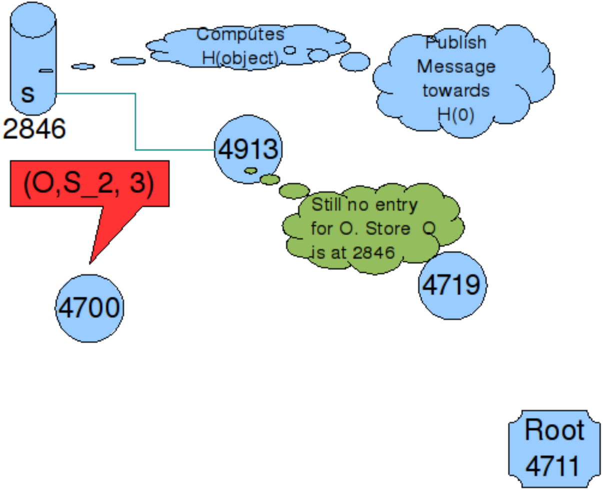
Publishing & Sharing Content
Exemplifying the algorithm

- Server S wants to publish object O
- S computes H(O), the object ID of O (hash of the name).
- S sends a PUBLISH message to node H(O).
- Nodes visited by the PUBLISH message associate H(O) with physical address of S (including root node)
- These nodes are the tree of the object O

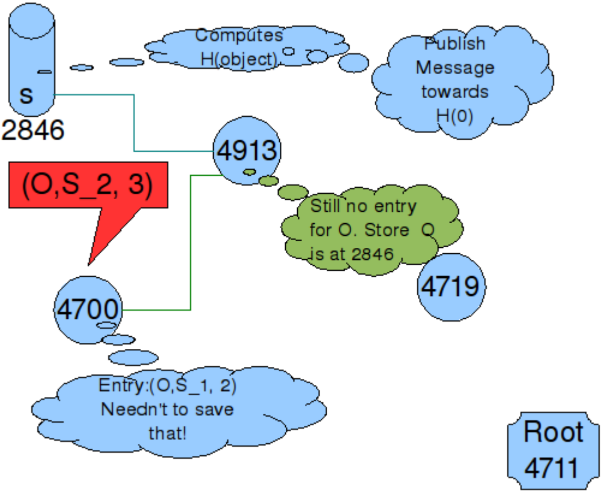
Publishing Visulization



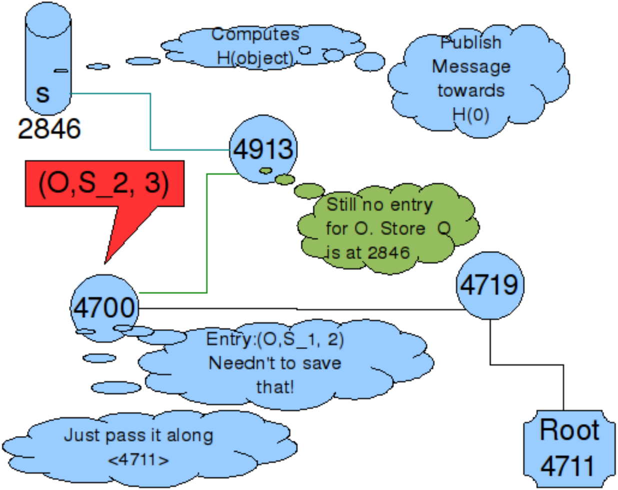
Publishing Visulization



Publishing Visulization



Publishing Visulization

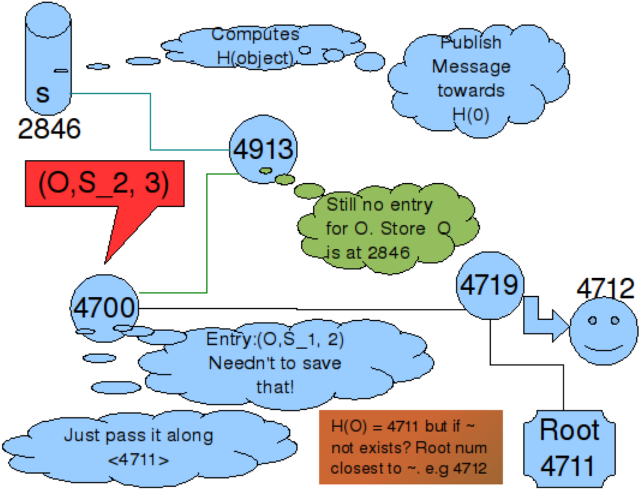


Publishing & Sharing Content

Exemplifying the algorithm

- Node H(O) might not exist. No problem. Message forwarded deterministically to node X with address closest to H(O).
- X - root node for object O

Publishing Visulization



Making Content Unavailable
Demonstrating the algorithm

- Server S wants to ’unshare’ object O

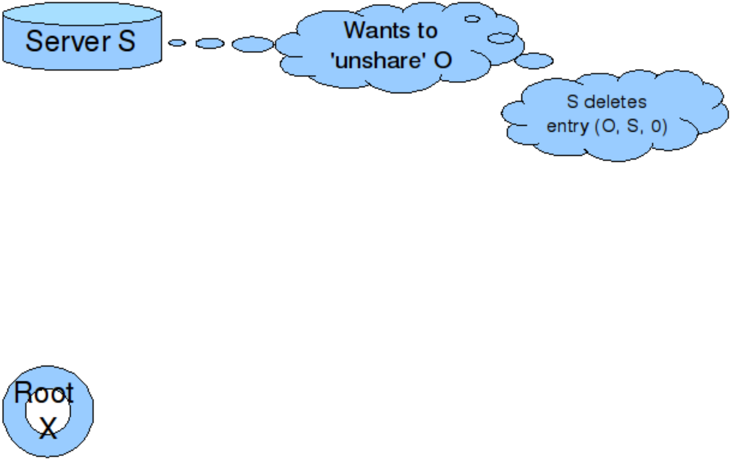
Publishing Visulization

Making Content Unavailable

Demonstrating the algorithm

- Server S wants to 'unshare' object O
- S deletes the entry $(O, S, 0)$ from its pointer list

Publishing Visulization

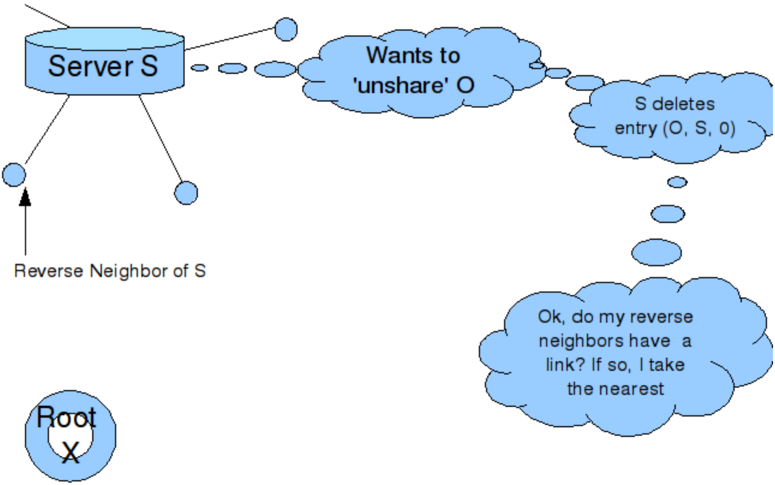


Making Content Unavailable

Demonstrating the algorithm

- Server S wants to ’unshare’ object O
- S deletes the entry $(O, S, 0)$ from its pointer list
- S asks its reverse neighbors for a copy of O
- $\rightarrow$ S chooses that node/entry with smallest costs

Publishing Visulization

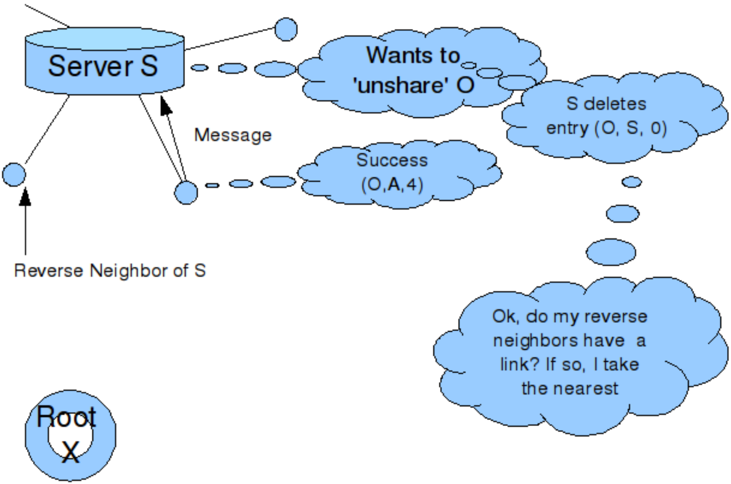


Making Content Unavailable

Demonstrating the algorithm

- Server S wants to 'unshare' object O
- S deletes the entry $(O, S, 0)$ from its pointer list
- S asks its reverse neighbors for the nearest copy of O
- If ∃ copy of O S receives success messages.

Publishing Visulization

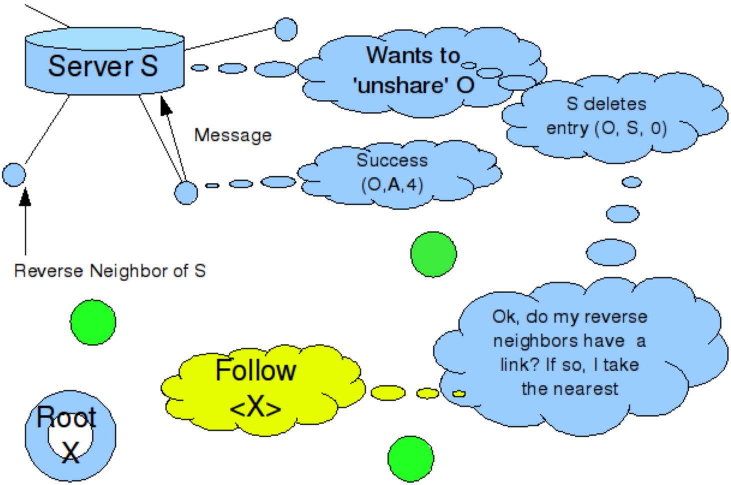


Making Content Unavailable

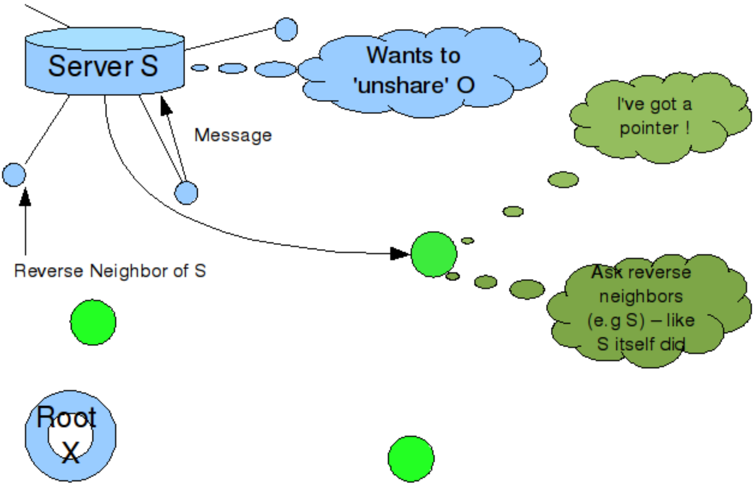
Demonstrating the algorithm

- S passes the delete request along the $\langle$ S $\rangle$
- Stops on not finding an entry or when on root.

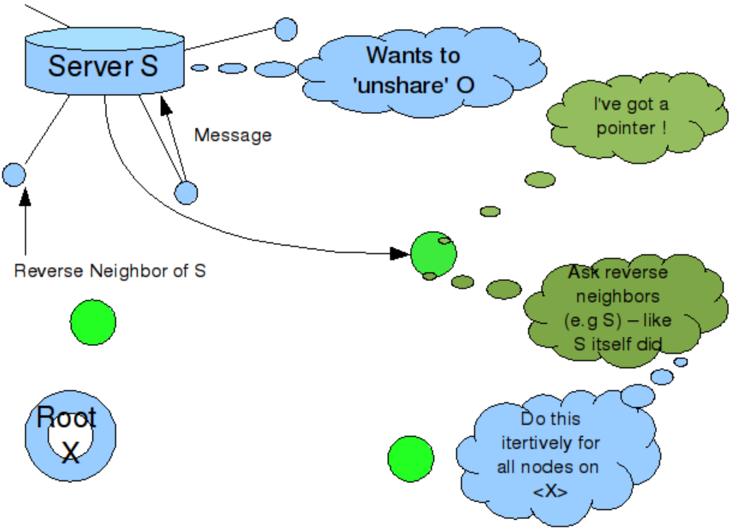
Publishing Visulization



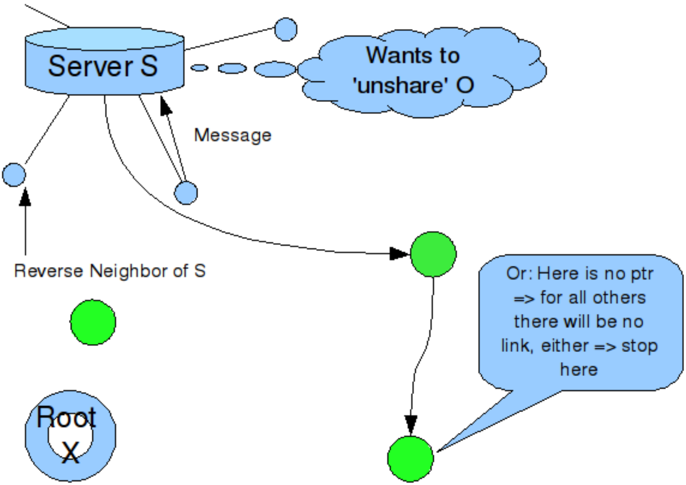
Publishing Visulization



Publishing Visulization



Publishing Visulization



Theorem 1: Let x be any node in V and let A be any object in $\mathcal{A}$. If y is the nearest node to x that holds a shared copy of A, then the expected cost of a read operation is: $\mathcal{O}(f(l(A)) \cdot c(x, y))$.

Sketch of Proof Theorem 1

Sketch of Proof:

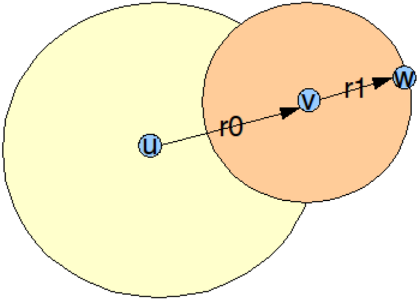
We will show something like a transitive closure:

1.) Show that if v is in N(u,k0) and w is in N(v,k1)
=> w is in N(u, $\Delta^2 k_0 \Delta k_1$)

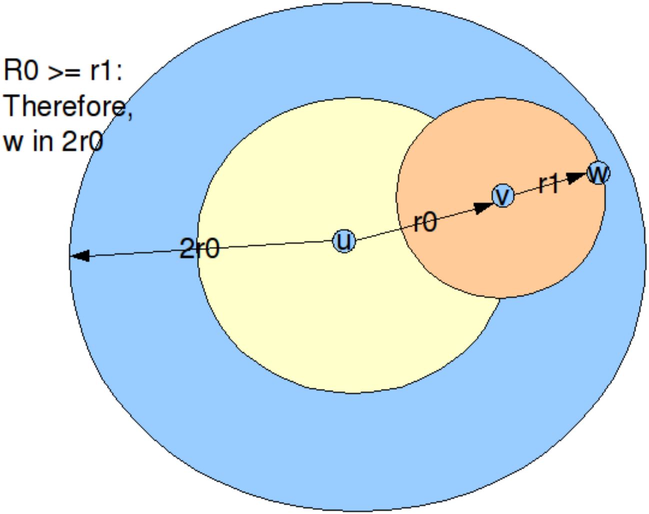
Let r0 denote the radius of N(u,k0) and r1 N(v,k1)
I.) r0>=r1 => |M(u,c(u,w))| <= | M(u,r0+r1)| <= Δ k0

Sketch of Proof Theorem 1

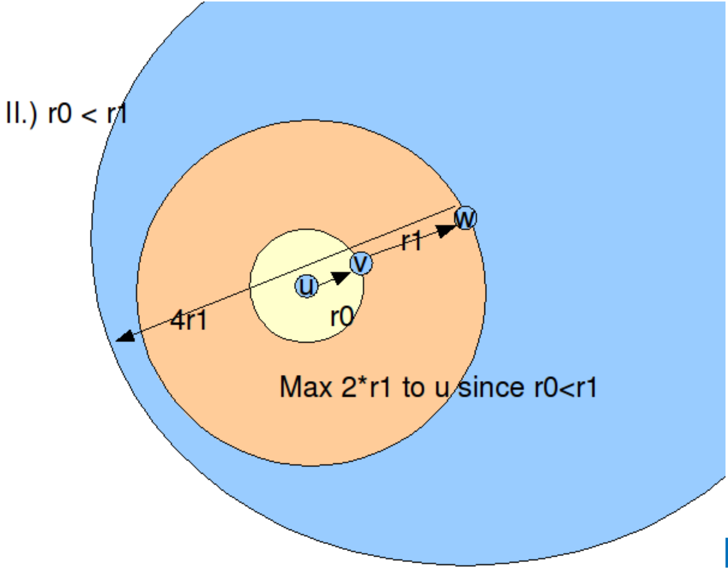
R0 >= r1:



Sketch of Proof Theorem 1



Sketch of Proof Theorem 1



Sketch of Proof Theorem 1

=> w is in N(u, $\Delta^2 k_0 \Delta k_1$)

Let k1 > Δ^2k0 and v in N(u,k0)

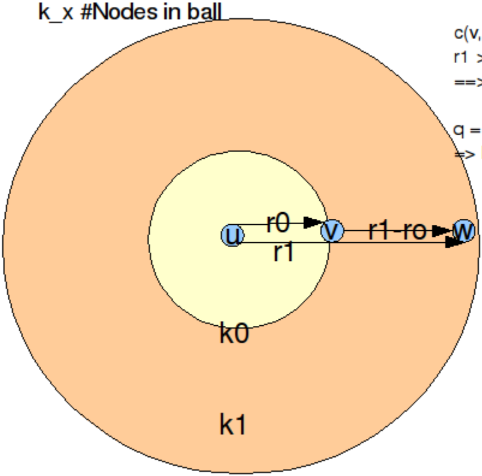
=> Hypo: q(v, N(u,k1) >= k1/Δ

=> Hypo: r(v, N(u,k1) =< Δk1

let q (u, S) denote the largest integer k such that N (u, k) is a subset of S

let r (u, S) denote the smallest integer k such that N (u, k) is a superset of S

Sketch of Proof Theorem 1



c(v,w) => r1-r0

r1 >= 4*r0 (cf. Slide before)

==> 2(r1-r0) >= r1+r0

q = |M(v,c(v,w))| >=|M(v,2c(v,w))| / Δ

=> k1/ Δ

Proof for r(v,N(u,k1) <= Δ k1

analog

Sketch of Proof - Theorem 1

- Let k be $= |M(u, c(u, v))|$.
- What is the probability that u is a primary (i,j)-neighbor of v, if distance is c(u,v)

- Cancel nodes in ball within that radius until only those two are left
- $Pr[P_j(u) holds] = (1/2)^{(i+1)b}$
- $\rightarrow (1 - (1/2)^{(i+1)b})^{NumberofNodesinarea}$
- Nodes in area are $k/\Delta - 2$. k minus v and u.
- upper bounded by $e^{-(k/\Delta - 2)/2^{(i+1)b}}$

Sketch of Proof - Theorem 1

- Corrollary: $N(u, 2^{ib}log(n))$ contains (i,j)-primary neighbor of v whp.

Sketch of Proof - Theorem 1

- We will proove
  $u \in V, i \in [log(n)/b] \rightarrow Numberofi - leaves(u) = \mathcal{O}(2^{ib} * log(n))$
- $v \in i - leaf(u) \rightarrow v \in N(u, c_0 \cdot 2^{ib} log(n)); c \in R$
- Corol. 1 $\forall j \in [i], v_j \in N(v_{j+1}, c_1 \cdot 2^{j+1} log(n))$
- Proof by Induction:

Sketch of Proof - Theorem 1

- Hypothesis: $\forall j \in [i+1], v = v_0 \in N(v_j, c_0 \cdot 2^{jb} log(n))$ whp
- Induction base j=0 trivial
- Induction step :
  - Assumption: $v \in N(v_j, c_0 \cdot 2^{jb} log(n))$
  - By Corol1 : $v_j \in N(v_{j+1}, c_0 \cdot 2^{ib} log(n)$
  - remember if $v \in N(u, k_0) and \omega \in N(v, k_1) \rightarrow w \in N(u, \Delta^2 k_0 + \Delta k_1)$ but set $u = v_{j+1}$, $v = v_j$ and $\omega = v_{j-1}$
  - $\rightarrow v \in N(v_{j+1}, (\Delta^2 k_0 + \Delta k_1) \cdot 2^{jb} * log(n)$

Sketch fo Proof - Theorem 1

- set $(\Delta^2 k_0 + \Delta k_1) = c'$, that is a constant factor
- finally $\rightarrow v \in N(v_{j+1}, c' \cdot 2^{(j+1)*b}) \rightarrow N(u, \mathcal{O}(2^{ib} \cdot log(n))$
- Since by reg. of metric we can derive that the radius increases only by a constant factor, too.
- only finite number of steps => $\mathcal{O}(c(x, y)$
- we won't proove: $E[\sum_{0 \leq i < \tau} c(x_i, x_{i+1}) + c(y_i, y_{i+1}))] \in \mathcal{O}(c(x, y))$

Theorem 2: The expected cost of an insert operation is $\mathcal{O}(C)$, and that of a delete operation is $\mathcal{O}(C \cdot logn)$.

Theorem 3: Let q be the number of objects that can be stored in the main memory of each node. The size of the auxiliary memory at each node is $\mathcal{O}(q \cdot log^2 n)$ words whp.

- **Definition 4.1:**

  Adaptability: The number of nodes whose auxiliary memory is updated upon the addition or removal of a node.

- **Theorem 4** The adaptability of our scheme is $\mathcal{O}(logn)$ expected and $\mathcal{O}(log^2 n)$ whp.

Overview

Benefits of PR

- Locality
- Simple fault handling
- Scalable

Shortcomings of PR

- Global knowledge required to build routing tables
  - Expensive to initialize the network
- Static network
  - Bad for P2P
- Root node vulnerability
  - Causes moderate maitenance

Conclusion

- PR provides a scalable, quite realiable, static network structure
- PR exploits locality

- Outlook
  - How does it acquit itself on arbitrary cost functions?
  - Is there any chance to improve the adaptability?
  - How about hotspots or taking network bandwith into account?