Deployment of Collaborative Web Caching with Active Networks

Laurent Lefèvre, Jean-Marc Pierson, Sid Ali Guebli

Laurent.lefevre@inria.fr Jean-Marc.Pierson@insa-lyon.fr

IWAN2003, Kyoto, 12th dec 2003

Agenda

- *Introduction & Motivations*
- *Collaborative caching…*
- *…and active networks*
- *Framework for web caching services in AN*
- *Implementation issues and experiment*
- *Discussion and conclusion*

Introduction

- At the border of two worlds…
- Collaborative web caching within active nodes
- To :
  - Reduce network traffic
  - Reduce server load
  - Reduce client latency
  - Rapidly test new features
  - Deploy dynamically new services in the network
- Add some intelligence in the nodes for the transfer of web traffic, considering pros and cons of active networks

Collaborative caching

- ✓ Distributed caches cooperates to efficiently share resources (files)
- ✓ What is to communicate between caches ?
  - ✓ Data
  - ✓ Summary/digest of data (tables)
- ✓ How to organize the collaboration ?
  - ✓ Geographic based
  - ✓ Hierarchical
  - ✓ Flat (construct adaptive meshes)

and active networks

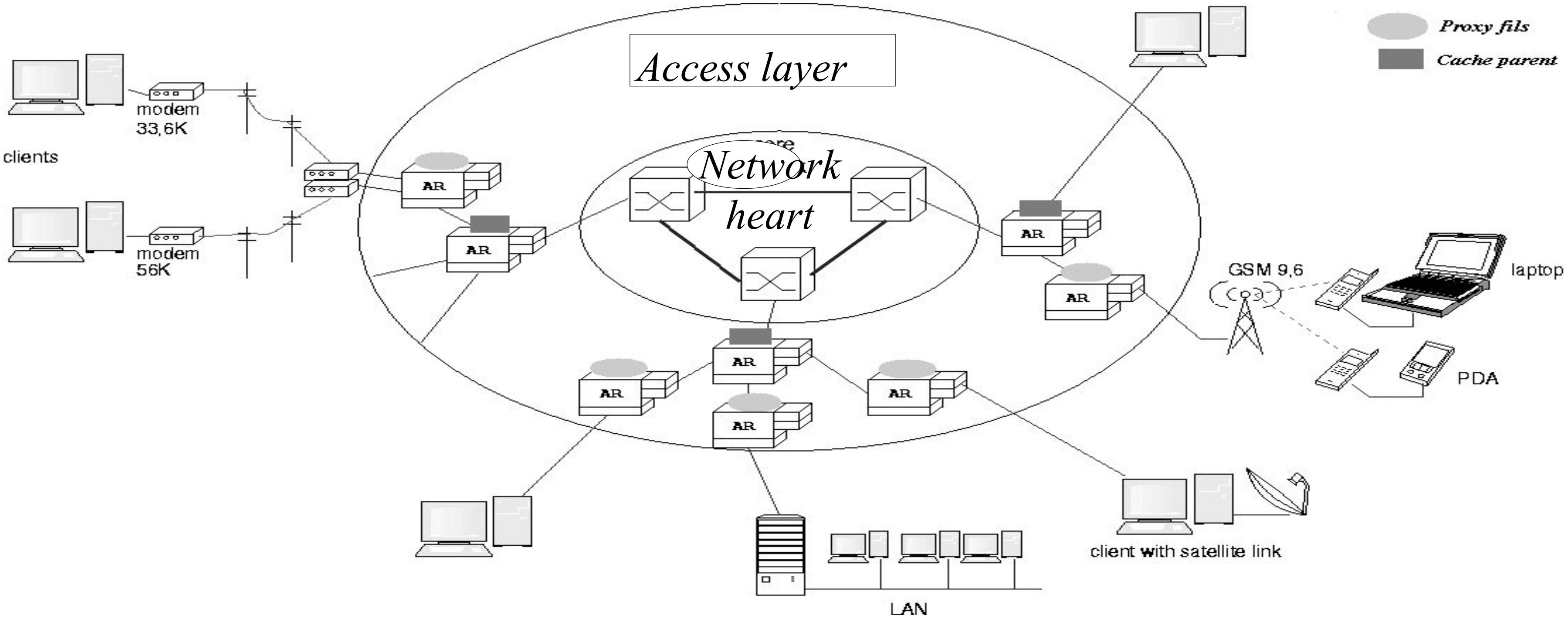
- Pros :
  - Deploy rapidly new protocols
  - Catch and process « web packets » on the fly
- Cons :
  - Limited storage capacity for caching in active routers
  - Active routers are firstly routers, they must do the rest at best effort

Our proposal

- ✓ A hierarchical caching
- ✓ Managing summaries of data that are cached (mirror table)
- ✓ Inter-cache communication protocols
- ✓ Based on services permitting the collaboration between caches
- ✓ Two main services :
  - ✓ Localize the cached data
  - ✓ Deliver the data to neighborhood

A two-level hierarchy is **enough**!

- Children caches : close to the clients
- Parent caches : group a community of children caches



Which information to exchange ?

Mirror tables

- used to localize the documents
- reflects the content of the caches : which document is on which children cache ?
- a compact representation, using the Bloom filter technique (IP of children cache, array for BF F, size of array, number of false hits…)

Bloom filter technique

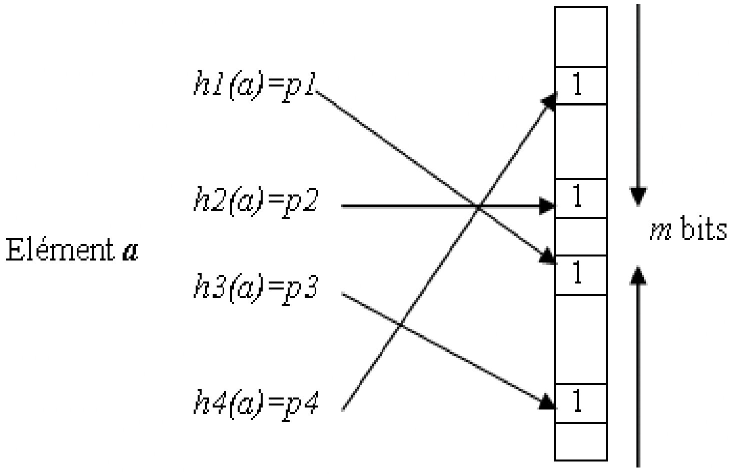
✓ Represent a set of *n* elements E = {*a1,a2, ... ,an*} in an array of bits (filter) *v* of size *m*.

✓ *k* independent hash functions $h_1, h_2, \ldots, h_k \rightarrow [0..m\text{-}1]$

✓ For each ***a*** ∈*E,* compute{ *h1(a)*, *h2(a)*, . . , *hk(a)* }

✓ Set the corresponding bits in *v* to 1

v filter



✓ Compact coding

✓ False hit problem :

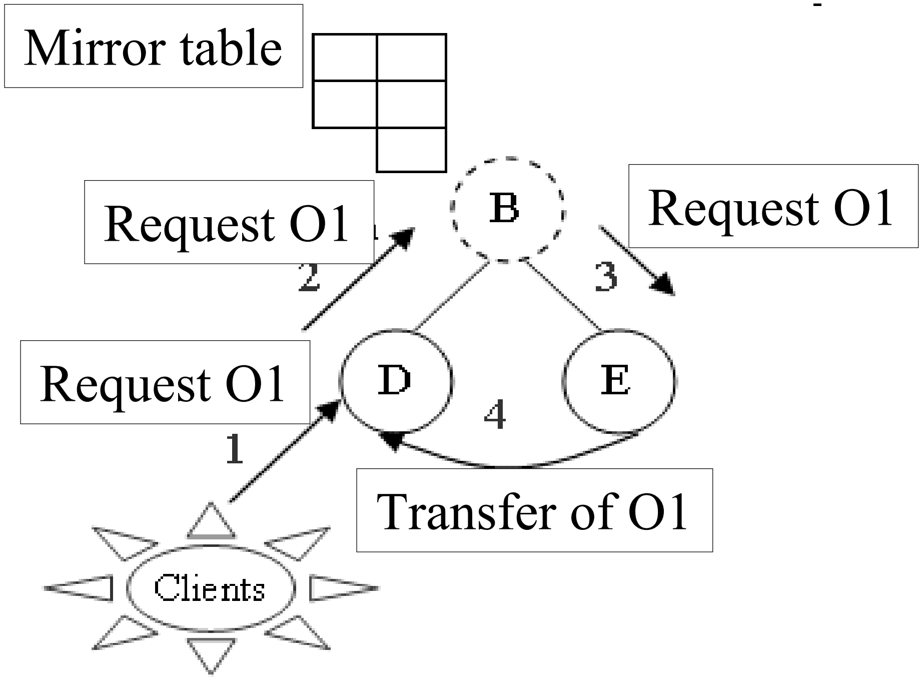
$$p_{err} \approx \left(1 - e^{-\frac{kn}{m}}\right)^k$$

Inter-cache communication protocol

- ✓ Under the responsability of a parent cache (supervisor)
- ✓Based on mirror tables
- ✓Localise :
  - ✓Keep a snapshot of children cache contents in mirror tables
  - ✓Look up mirror tables to redirect requests
  - ✓Load balance the workload amond children caches
- ✓Deliver
  - ✓Peer to peer transfer : less workload for parent caches

Inter-cache communication mechanism

Example :



✓ Consistency of mirror tables ? How to update those ?

Mirror table updates

Two complementary methods :

Parametrized with two validation thresholds :

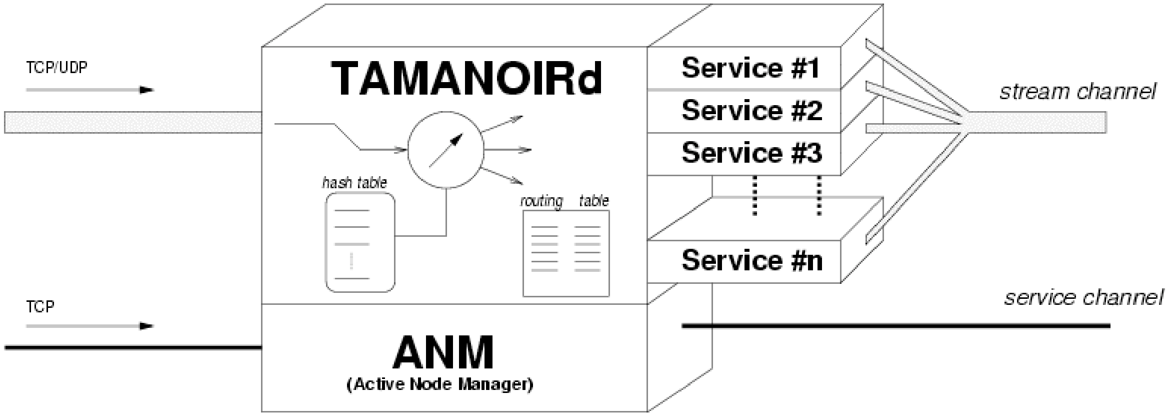
□ □ : parent side : number of False Hit

□ □ : children side : number of data movement in the cache

Deployment

- based on TAMANOIR execution environment (RESO-INRIA).
  - ✓ TAMANOIRd : deamon on TAN (Tamanoir Active Node).
  - ✓ ANM (Active Node Managment)

✓Development in user space



Deployment

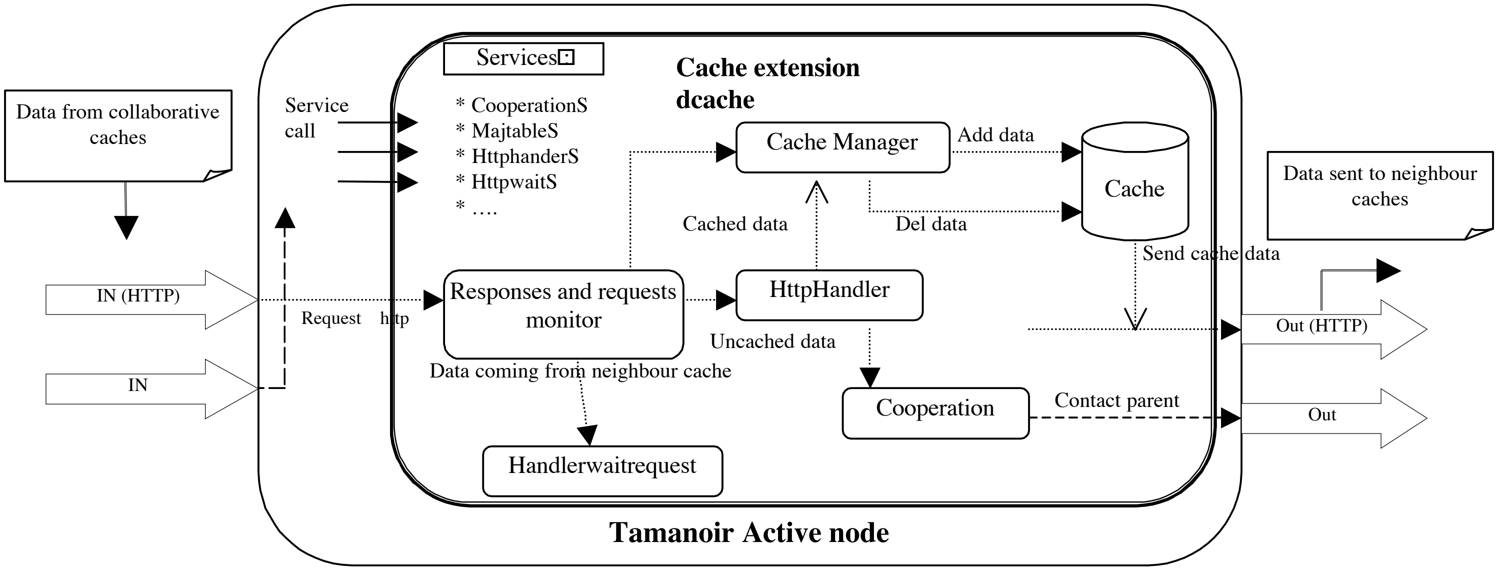
- AN acts as a proxy
- Explicit requests on dedicated ports (no active packets)
- Clients configure one AN (children cache) like a proxy in web browser
- AN exchange data and take tuning decisions independently of data streams (proactive approach)

Collaborative cache service in active node

✓Children cache

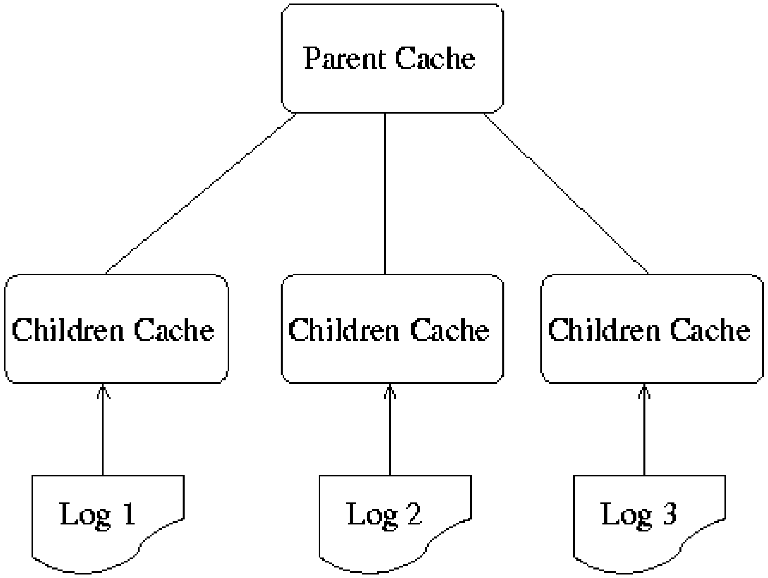
✓ based on composition of small services (CooperationS ,…).

✓ rely on TCP and Java



Experiments

- ✓ Platform :
  - 1 parent cache
  - 3 children caches with associated log files



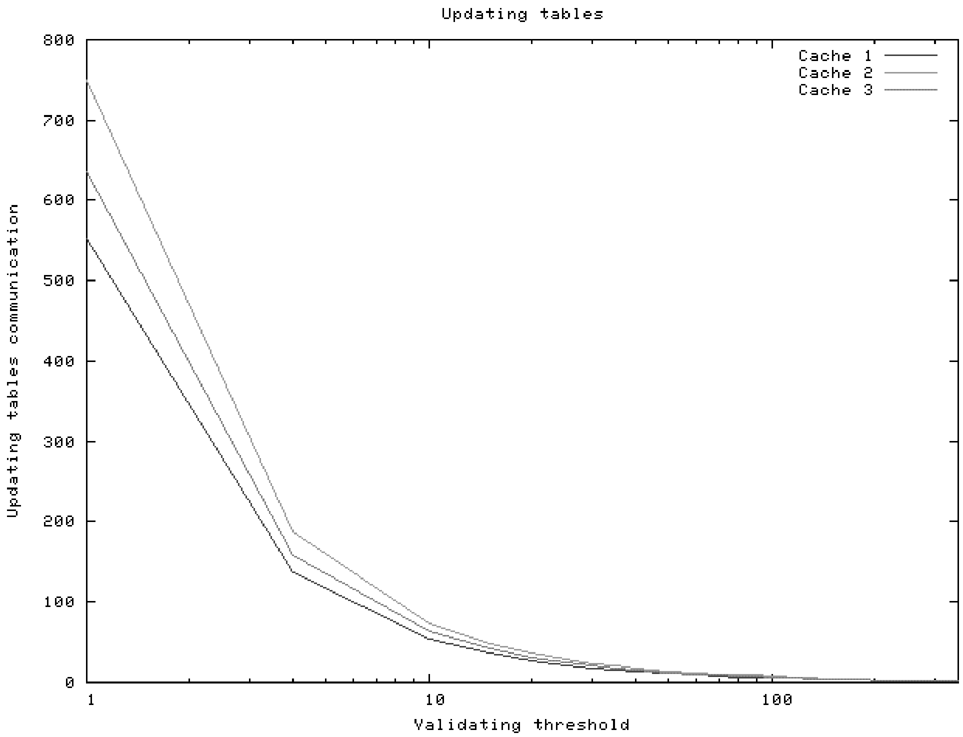
- ✓ Used data (real life proxy)

- 4 hash functions
- filter size *m=8000 bits*
- validation tresholds □ = □.
- 1000 requests from each proxy cache
- 1766 different documents

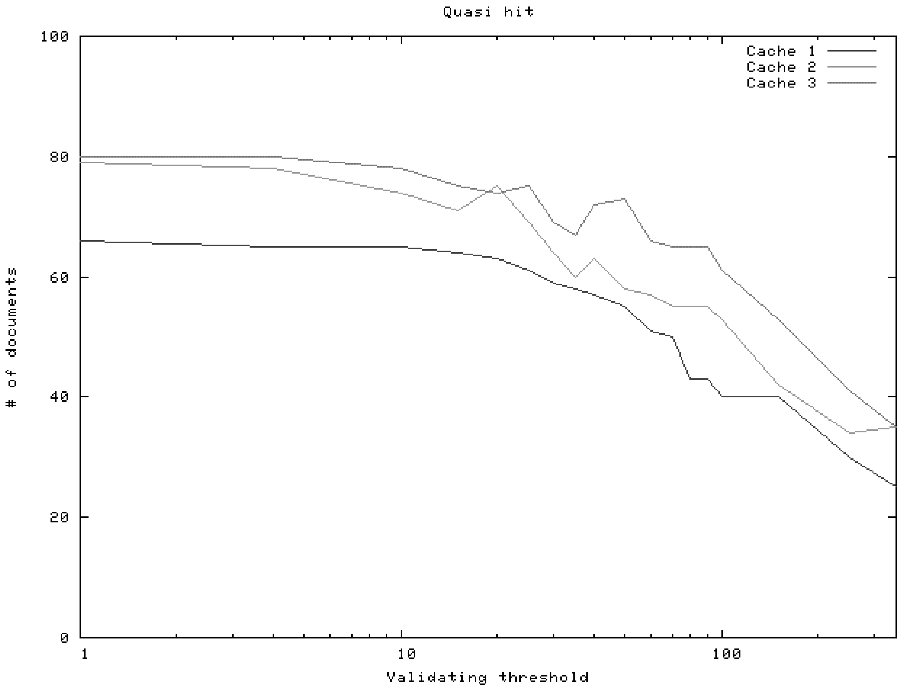
Quasi hits

- Number of hits when a children cache is able to get a requested document from one of its neighbor

Experiments : « Infinite » caches



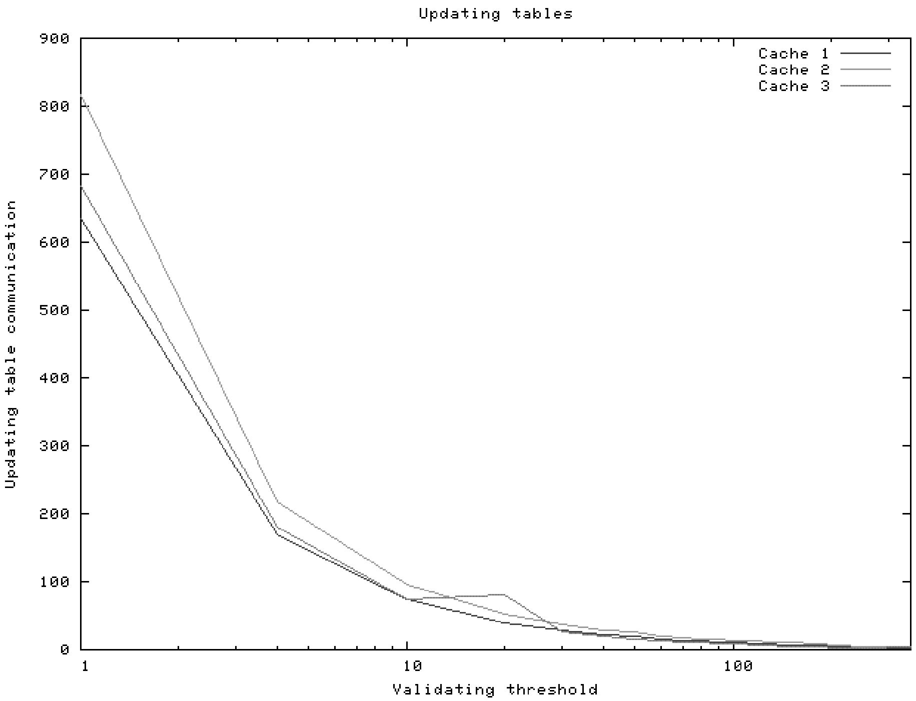
- Losing consistency of tables



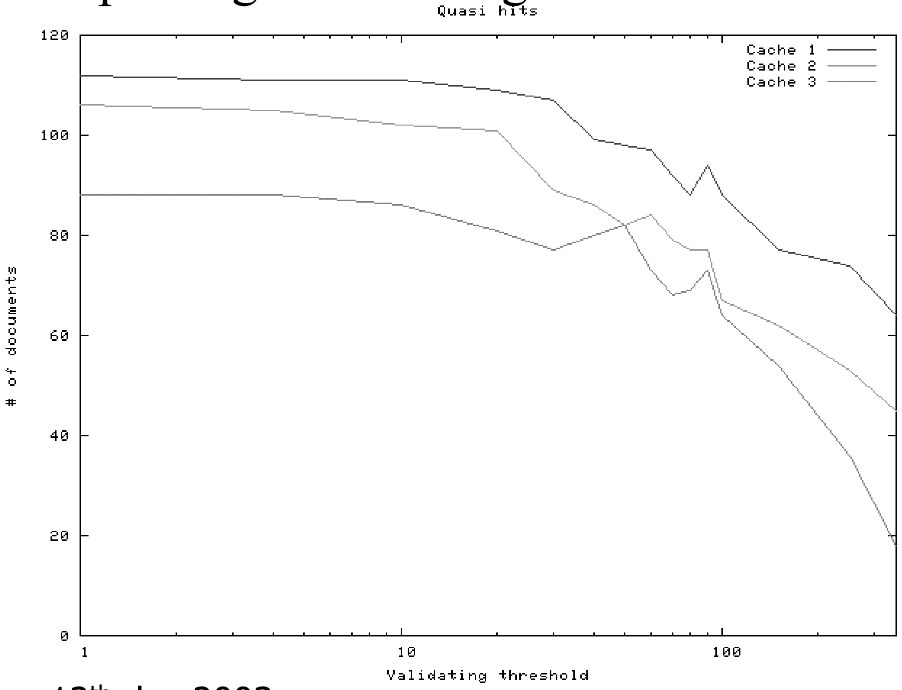
- LH : 439/221/339

Experiments : Limited caches

- 10% of requested space
- LRU policy
- Updating table during doc. removal



- LH : 355/149/291



Conclusion

- ✓ Active intelligent nodes allow a fast localization of documents in a community of cooperative caches
- ✓ Proposal of a cooperative caches model respecting requirements of active nodes
- ✓ Real high level active services with two main features : *Localization and delivery.*
- ✓ Hierarchical organization and definition of an inter-caches communication protocol
  - ✓ Limiting number of exchanged messages between caches to get a document
  - ✓ Opportunistic communication steps / proactive tuning
- ✓ Load balancing facilities between children caches : children caches can work in a « best effort » way
- ✓ TAMANOIR experimental platform

Lessons learned

- End users want a high performance plug and play active node
- Want to use AN like a proxy : no active packets !
- They need TCP
- They want to design active services with the same language they use for their application
- Need of a simple and easy API

Future works

- ✓ Evaluating performance aspects
- ✓ Providing reliable functionalities to IBP depots to support data caching
- ✓ Using intelligence in the node to cache active pages : « active cache » / link with web services

Questions ?