# Targeted Search: Reducing the Time and Cost for Searching for Objects in Multiple Server Networks

**Graciela Perera** and **Ken Christensen**
Computer Science and Engineering
University of South Florida
Tampa, FL 33620
{gpererao, christen}@csee.usf.edu

**Allen Roginsky**
NIST
100 Bureau Drive, Stop 8930
Gaintersburg, MD 20899-8930
allen.roginsky@nist.gov



IEEE IPCCC 2005 – April 8, 2005

USF UNIVERSITY OF SOUTH FLORIDA
COLLEGE OF ENGINEERING

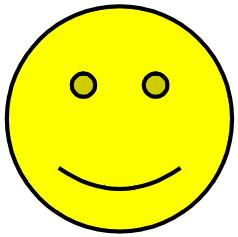
# Topics

- **This talk is about**

  **1) Traffic**

  **2) Flooding**

**Traffic and flooding!**



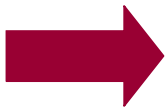
# Topics continued

- Introduction
- Targeted search method 

- Performance evaluation
- Ditella – Targeted search in Gnutella
- Summary and future work

# Introduction

- **Question: How to efficiently find an object in multiple servers?**
  - For example, in a P2P network
- **Efficient = fast with low overhead**
  - Overhead is query traffic on a P2P network
- **We have a trade-off between…**
  - Time to complete a search
  - Amount of overhead traffic



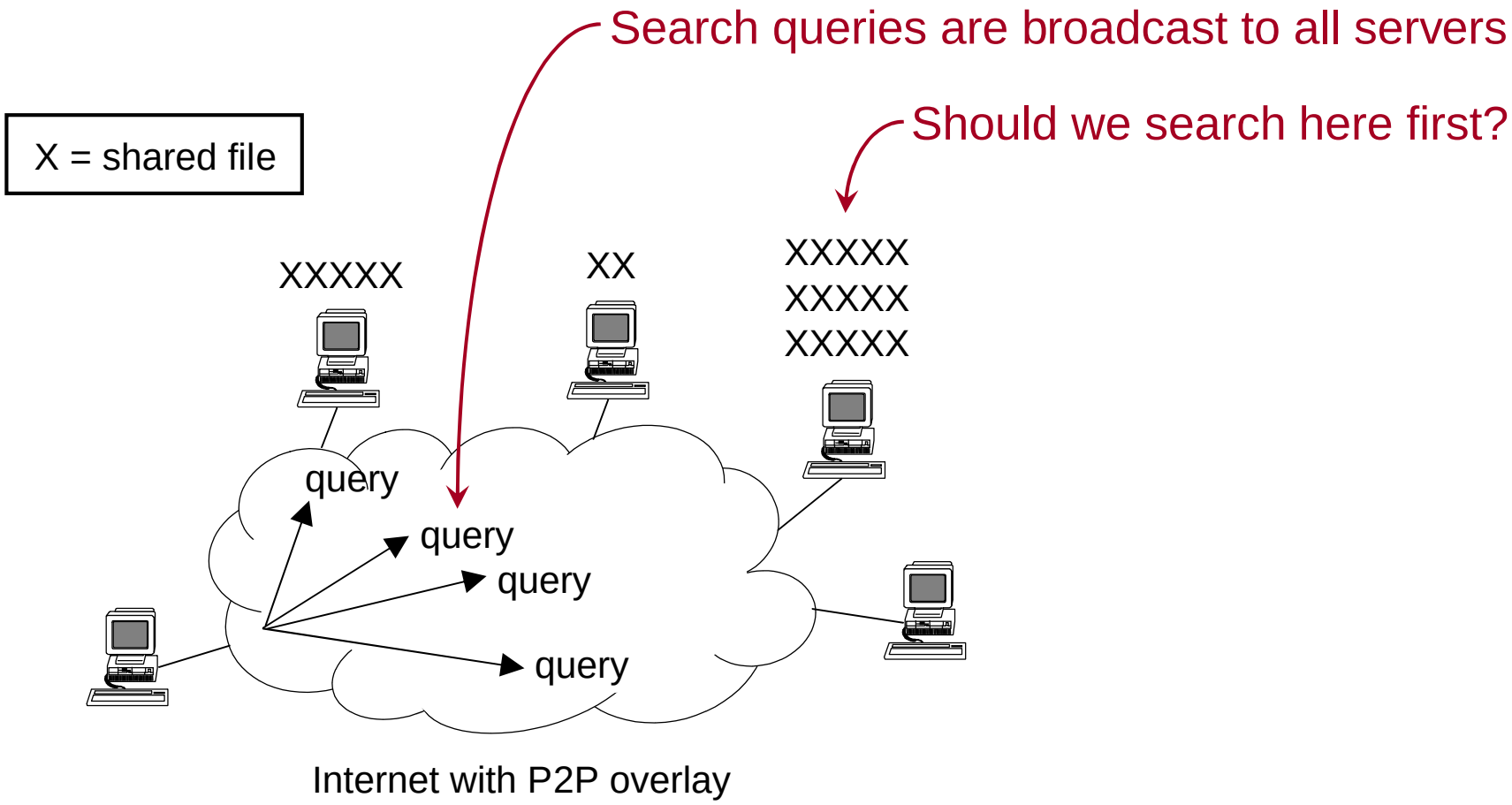
Much of the traffic on the Internet is P2P search queries

# Introduction continued

- **P2P networks are an example of multiple server networks**
  - Used for file sharing
- **P2P search methods**
  - Napster
    - Single master directory
  - Gnutella
    - No master directory – broadcast a query to all “servents” (nodes)
  - Kazaa
    - Hierarchical directory structure (many local directories)
  - Chord
    - Hash of file name determines where a file is stored

# Introduction continued

- **Power-law distribution of files is a reasonable assumption**
  - Few servers share most of files



# Introduction continued

- **Our hypothesis…**

**Individual nodes can "learn" the distribution of objects in servers and use this learned knowledge to improve searching.**

- **Search first in servers with the most files**
  - Highest probability of finding file
- **May not have to broadcast to all servers**
  - Thus, reduce the amount of query traffic

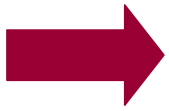
# Introduction continued

- Can prove that an optimal search exists
- Determine optimal search method for given time constraint *k*
- Have *N* servers and uniform probability of file being in a server
- Check at each step $s(1), s(2), \ldots s(k)$ such that $s(1)+s(2)+\cdots+s(k)=N$
- The mean cost is

$$E[C] = N/2 + \left(s(1)^2 + s(2)^2 + \cdots + s(k)^2\right)/2N$$

- The strategy that minimizes $E[C]$ is $s(1)=s(2)=\ldots=s(k)=N/k$ where

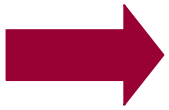
$$E[C] = N(k+1)/2k$$



For non-uniform distribution we show that weighted search is optimal (see paper for proofs)

# Targeted search method


- **A method to target high-probability nodes for a direct query**
- **Each node maintains a frequency list**
  - Tuples of **`<server_id, hit_count>`**
- **Frequency list is sorted by previous successes**
  - Sort variable is **`hit_count`**



Do a direct search to the $N_{top}$ highest ranked nodes and if the file is not found, then broadcast a query to all nodes

# Targeted search method continued

- **The algorithm**
  - Node searching for a file in *N* servers

**Step 1:** Send a direct query iteratively to the servers in the frequency list starting with the first listed server. This step terminates when $N_{top}$ servers have been queried or the object has been found.

**Step 2:** If Step 1 did not find the object then broadcast a query to all *N* servers.

**Step 3:** If Step 1 or Step 2 found the object in a server then download the object and update the frequency list.

# Performance evaluation

- **Performance evaluation using analysis and simulation**
  - Analysis with theoretical distribution of files
  - Simulation with traced (empirical) distribution of files
- **Control variable:**
  - Distribution of files in nodes
- **Response variables:**
  - Time = proportional to number of query steps
  - Cost = number of nodes queried

We seek closed form expressions for mean time and cost

# Performance evaluation continued

- **Define a peaked distribution for $\Pr[\text{file in server } i]$**
  - $N$ is number of servers
  - $K$ is a "tuning knob"

$$\Pr[\text{file in server } i] = \begin{cases} \dfrac{K}{N+K-1} & i = 1 \\ \dfrac{1}{N+K-1} & i = 2,3,\ldots N \end{cases}$$

- **$\Pr[\text{file in server } i = 1]$ is $K$ times greater than $\Pr[\text{file in server } i > 1]$**
- **As $K$ increases $\Pr[\text{file in server } i = 1]$ goes to 1**
- **As $K$ increases and $\Pr[\text{file in server } i > 1]$ goes to 0**

# Performance evaluation continued

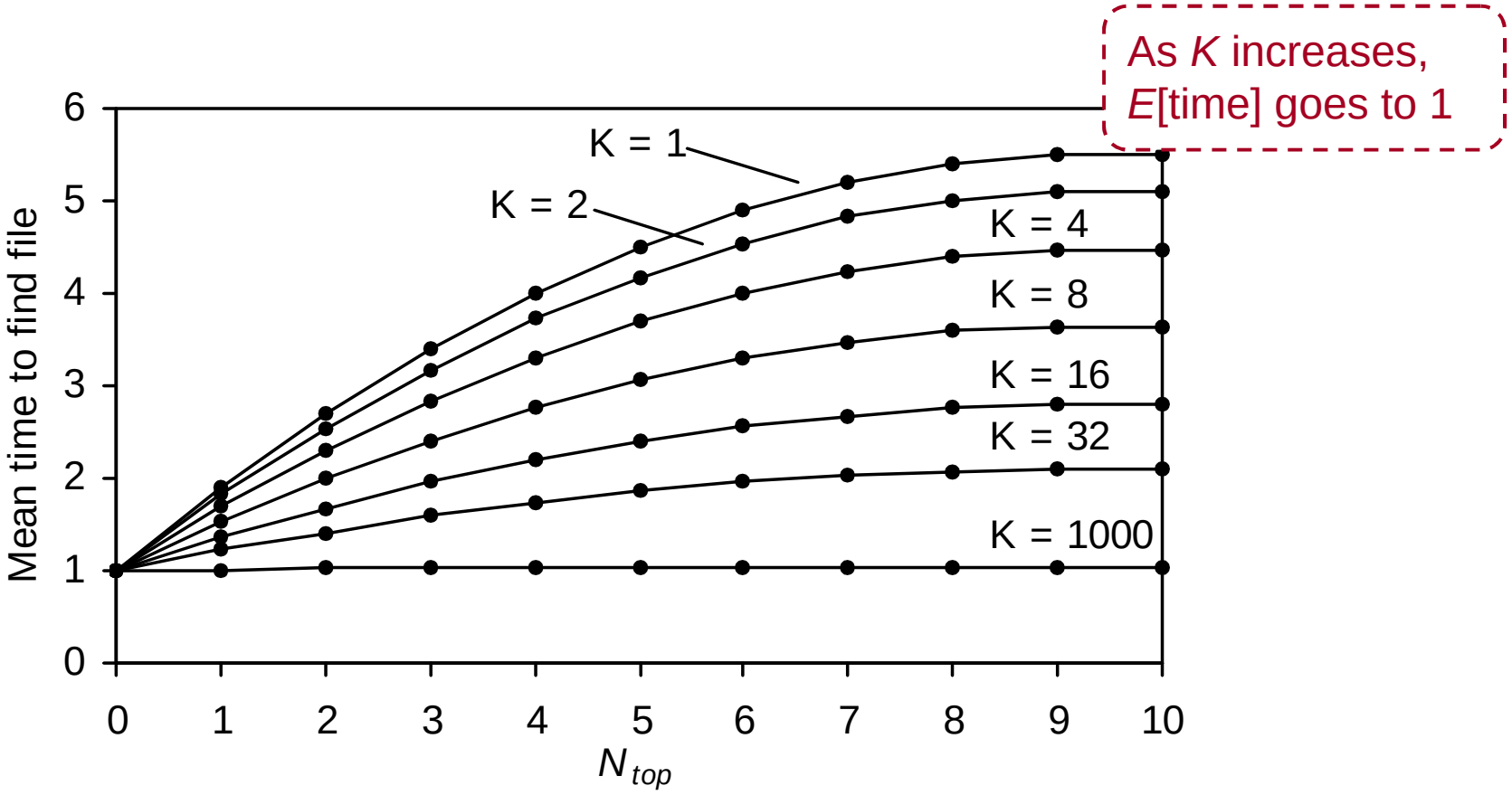
- **We wish to find an expression for**
  - The mean time to find an object (*E*[cost])
  - The mean cost to find an object (*E*[time])
- **For files distributed with the peaked distribution we find**

$$E[\text{time}] = \frac{2N_{top}N - N_{top}^2 - N_{top} + 2 \cdot N + 2K - 2}{2(N + K - 1)}$$

$$E[\text{cost}] = \frac{2N^2 - N_{top}^2 + N_{top} + 2K - 2}{2(N + K - 1)}$$

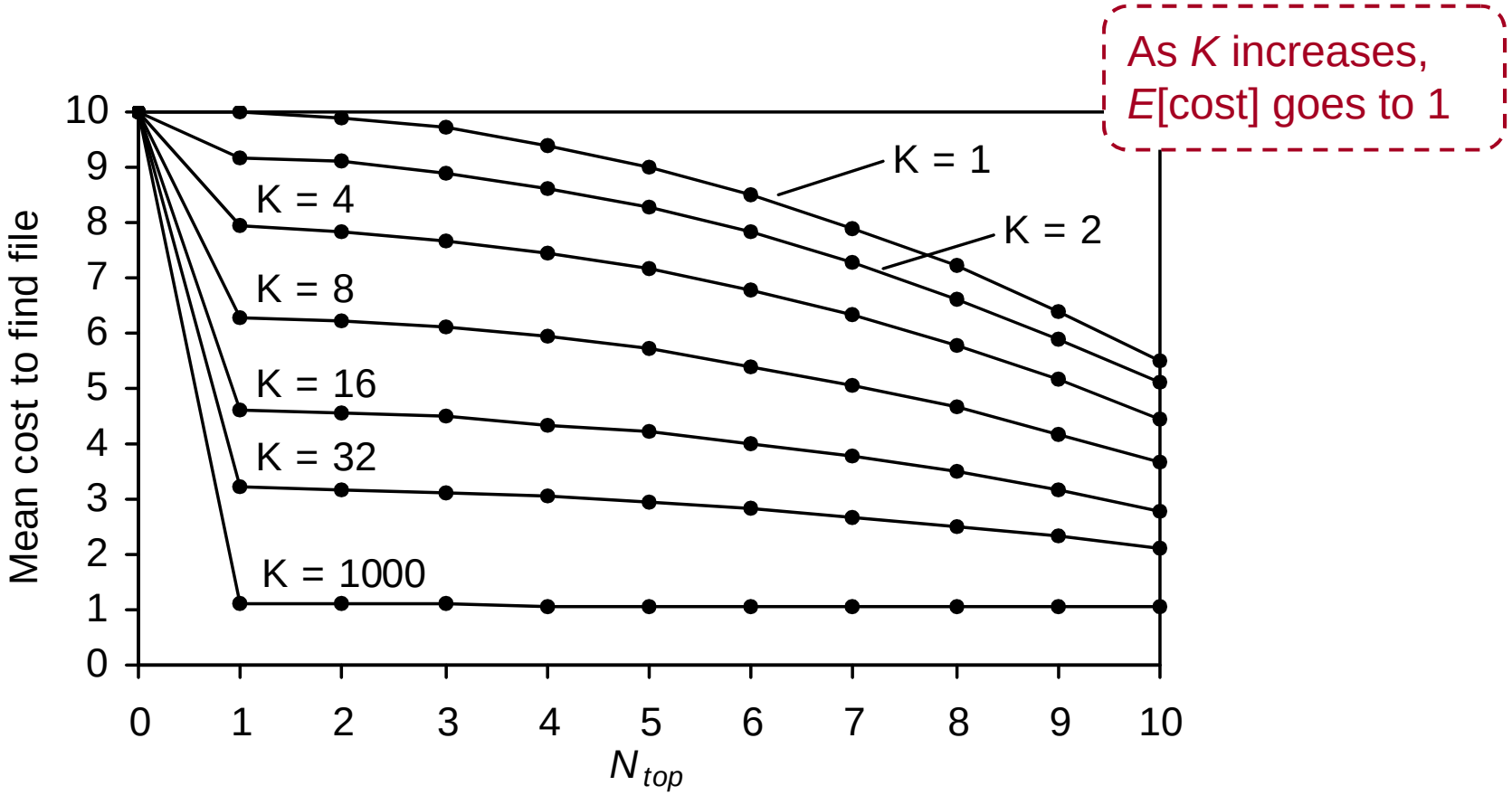
# Performance evaluation continued

- **Analytical mean time results for *N* = 10**



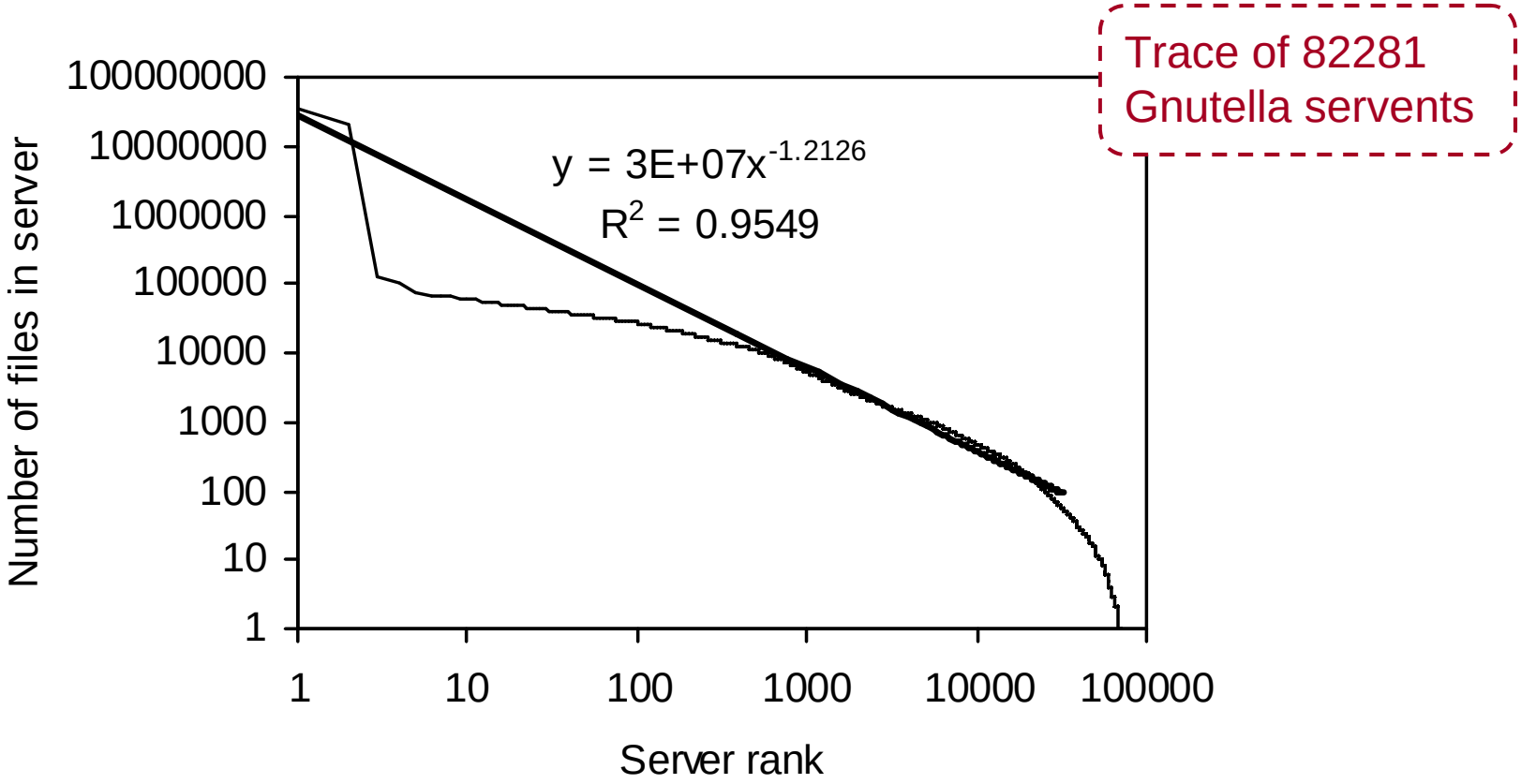
# Performance evaluation continued

- **Analytical mean cost results for *N* = 10**



# Performance evaluation continued

- **Simulation evaluation using Gribble et al. file trace for input**
  - See Power Law fit on log-log graph

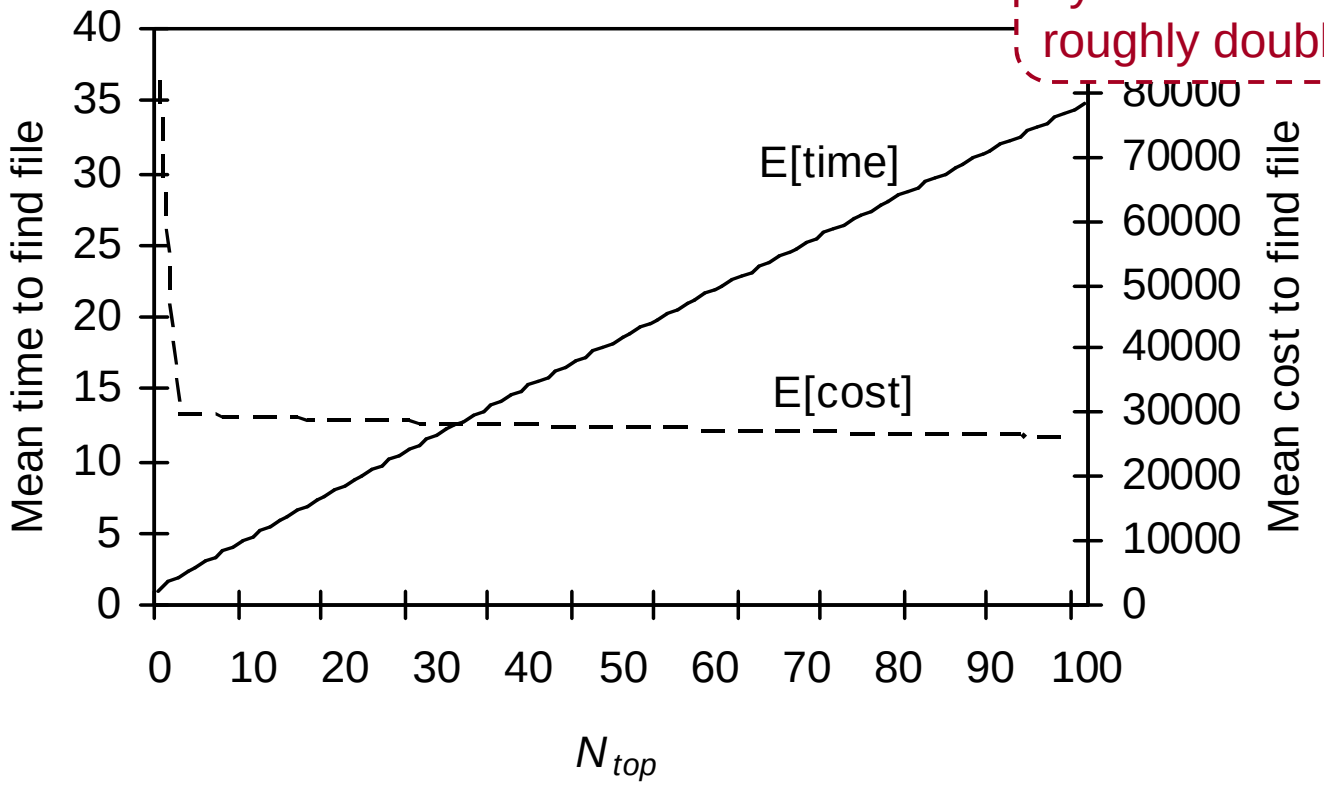



S. Saroiu, P. Gummadi, and S. Gribble, "A Measurement Study of Peer-to-Peer File Sharing Systems," *Proceedings of SPIE*, pp. 156-170, 2001.

# Performance evaluation continued

- **Simulation mean time and cost results**

For $N_{top}$ = 2 cost is reduced by 63% and search time is roughly doubled

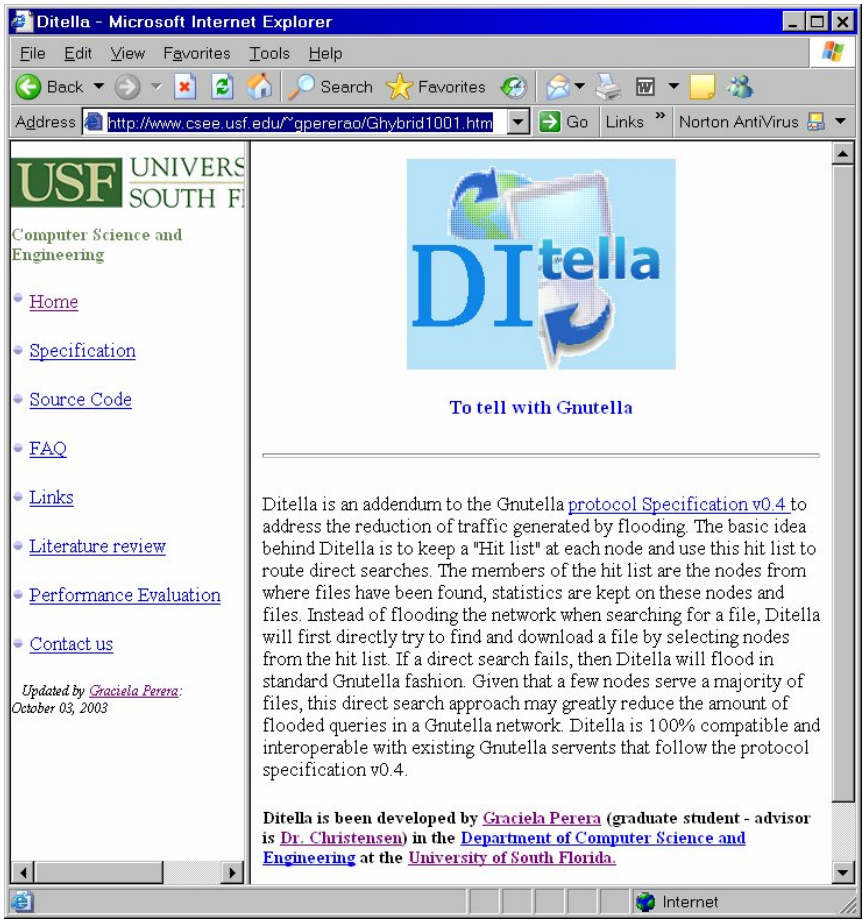


# Performance evaluation continued

- **See paper for…**
  - Comparison to random walk P2P search
  - Evaluation of convergence of frequency list to distribution

# Ditella – targeted search in Gnutella

- **Implemented targeted search**
  - In a Gnutella servant
  - For Microsoft Windows
- **Available at project website**




**http://www.csee.usf.edu/~gpererao/Ghybrid1001.htm**

# Summary and future work

- **Distribution of files to servers is often power law**
- **Can exploit this to reduce search time and cost**
- **Our new *targeted search* is: query first the most popular nodes**
  - Performance evaluation with analysis and simulation
- **Implemented targeted search in Gnutella**
  - We call this Ditella
- **Future work a better (more usable and portable) implementation**
  - Goal is widespread adoption and deployment

# Acknowledgement


- **The authors would like to thank…**
  - Stefan Saroiu and Steven Gribble at the University of Washington for sharing their trace data
  - Sasha Dos Santos at the University of South Florida for her assistance in developing the Ditella prototype
  - Shu-Jen Chang at NIST for her assistance with reviewing of the paper