# Exploiting Heterogeneity in Parallel and Distributed Systems

**Dick H.J. Epema**
**Delft University of Technology**
**Delft, the Netherlands**

**HeteroPar 2009**

**august 25, 2009**

# Heterogeneity (1): hardware

- Different hardware characteristics:
  - processor speeds and types
  - network bandwidth / asymmetric ADSL connections
  - …
- **Problem**: select suitable/optimal resources

# Heterogeneity (2): software

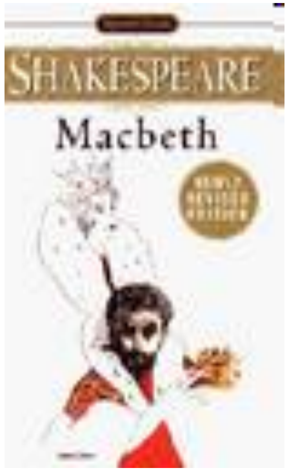
- Different software characteristics
  - operating systems
  - compiler versions
  - libraries
  - input files
- System configuration
- **Problem**: correct installation / resource selection

# Heterogeneity (3): management

- Systems management / ownership
  - authorization and access
  - usage rules (times of day, limits to sizes of jobs, priority to certain users)
  - system availability
  - level of system management
- **Problem**: resource description and selection / translation of requirements

# Heterogeneity (4): roles

- Different roles played by different machines
  - clients versus servers
  - peers, superpeers, trackers in P2P networks
  - social roles in P2P systems
- **Problem**: take into account different roles



# Case studies

1. **Grids**: processor co-allocation
2. **P2P systems**: measurements
3. **P2P systems**: cooperative downloading
4. **P2P systems**: semantic clustering

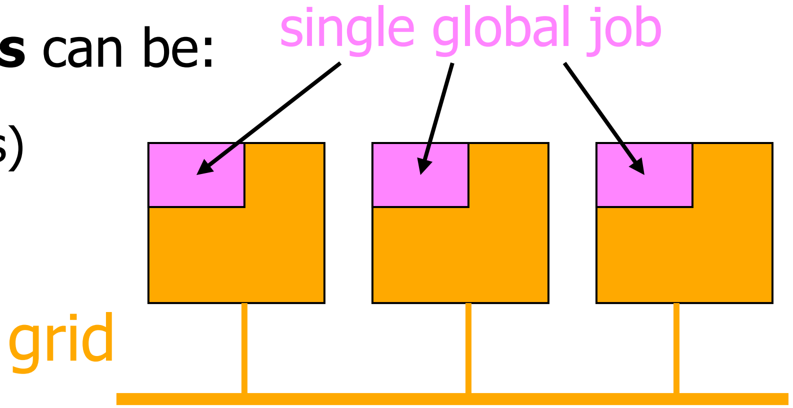
# The KOALA Grid Scheduler

## Processor and data co-allocation in grids

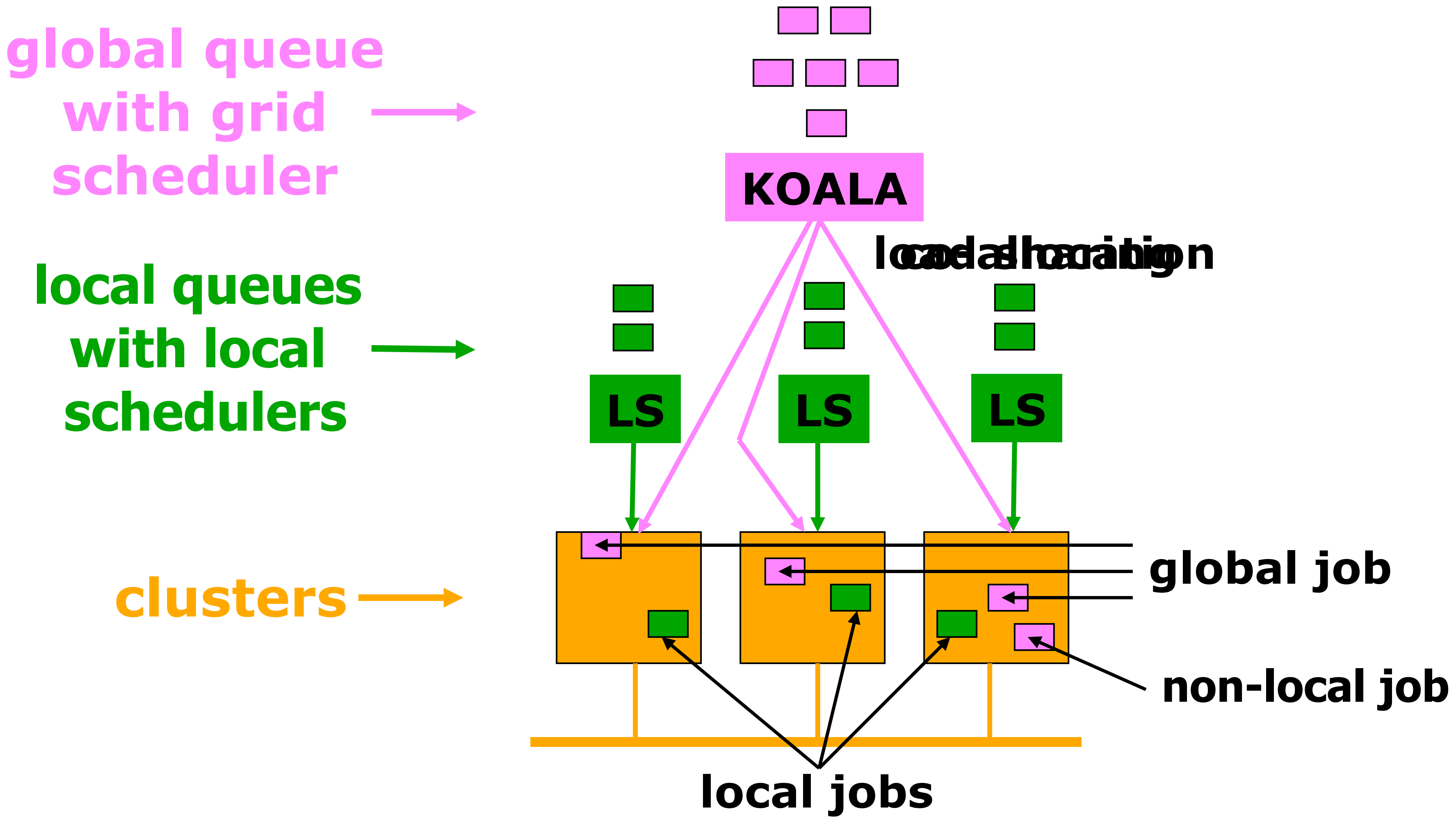
Dick Epema, Alexandru Iosup,
Hashim Mohamed, Ozan Sonmez

# Co-Allocation

- In grids, jobs may use multiple types of resources in multiple sites: **co-allocation** or **multi-site operation**
- **Reasons**:
  - to use available resources (e.g., processors)
  - to access and/or process geographically spread data
  - application characteristics (e.g., simulation in one location, visualization in another)
- Resource possession **in different sites** can be:
  - simultaneous (e.g., parallel applications)
  - coordinated (e.g., workflows)



# A model for co-allocation (1): schedulers



global queue with grid scheduler

local queues with local schedulers

clusters

KOALA

load sharing

co-allocation

LS

LS

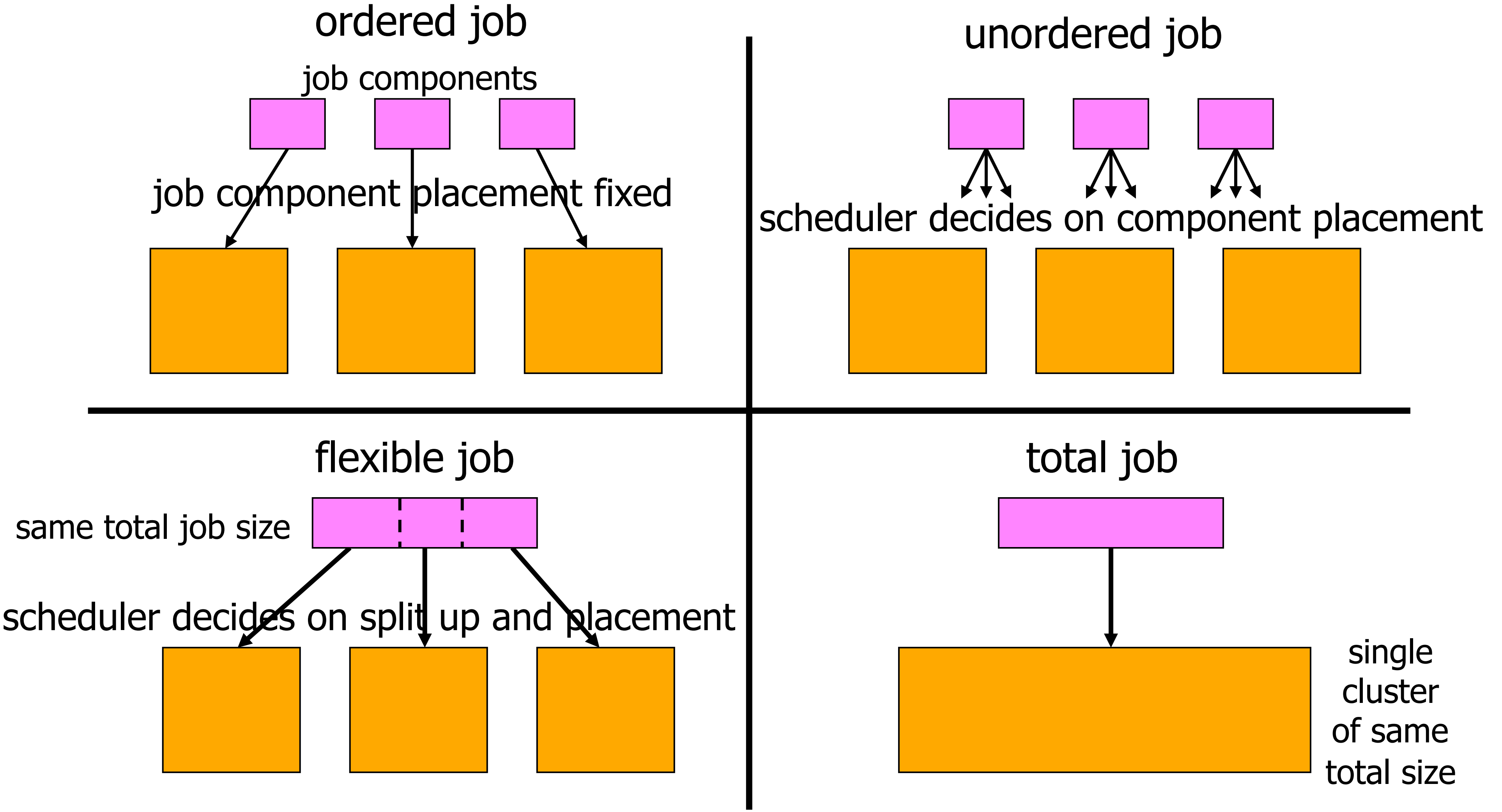
LS

global job

non-local job

local jobs

# A model for co-allocation (2): job types



ordered job

job components

job component placement fixed

unordered job

scheduler decides on component placement

flexible job

same total job size

scheduler decides on split up and placement

total job

single cluster of same total size

# A model for co-allocation (3): slowdown

- Co-allocated applications are **less efficient** due to the relatively slow wide-area communications
- **Extension factor of a job**:

$$\frac{\text{service time on multicluster}}{\text{service time on single cluster}} \qquad \textbf{(>1 usually)}$$

- Processor co-allocation is a **trade-off** between faster access to more capacity and shorter service times
- Communications libraries may be optimized for wide-area communication

# A model for co-allocation (4): policies

- **Placement policies** dictate where the components of a job go
- Placement policies for **unordered jobs**:
  - **Load-aware**: Worst Fit (**WF**)

    (balance load in clusters)
  - **Input-file-location-aware**: Close-to-Files (**CF**)

    (reduce file-transfer times)
  - **Communication-aware**: Cluster Minimization (**CM**)

    (reduce number of wide-area messages)
- Placement policy for **flexible jobs**:
  - **Communication- and queue time-aware**: Flexible Cluster Minimization (**FCM**)

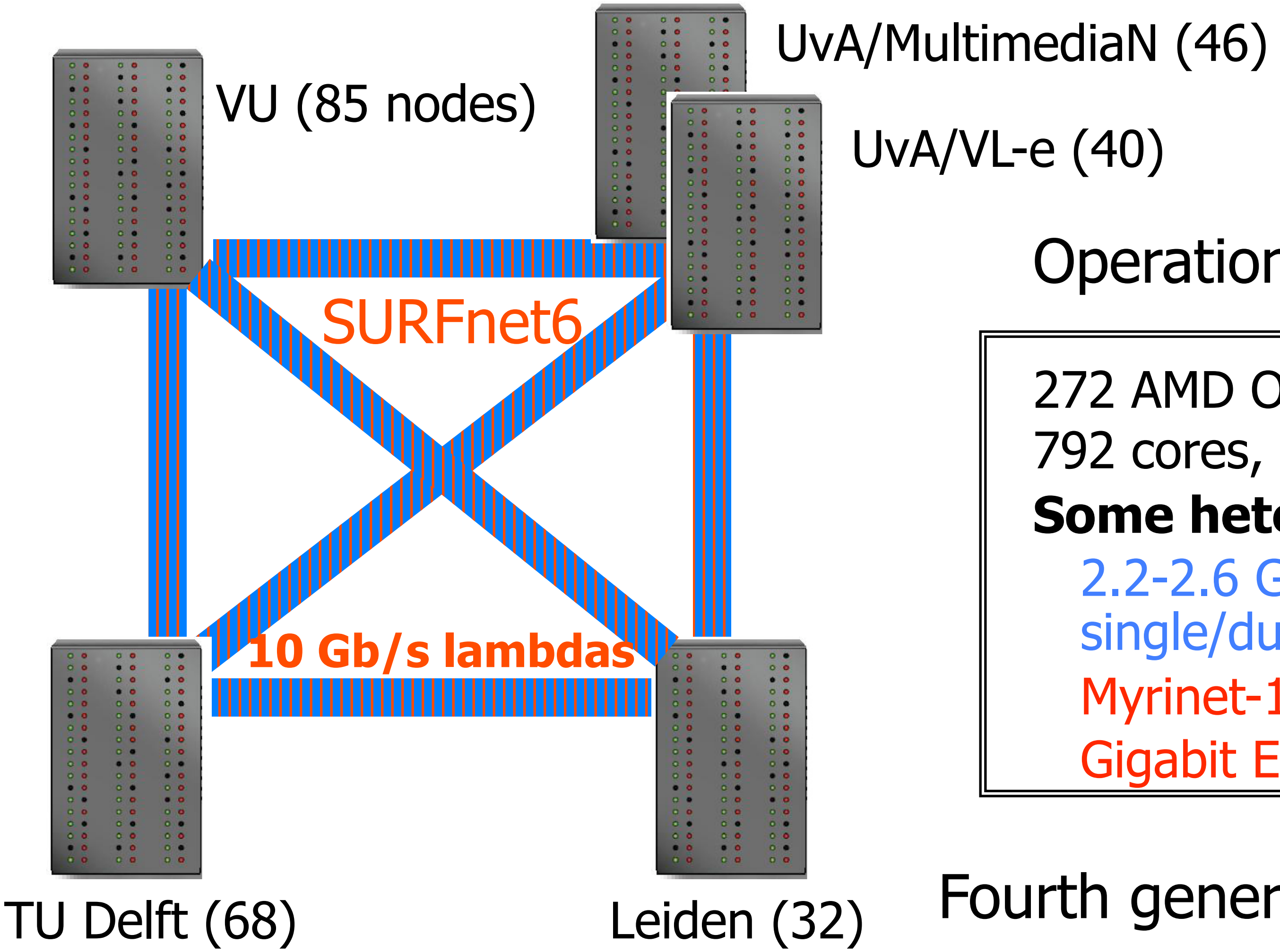
    (CM + reduce queue wait time)

# Simulations of co-allocation

- Processors only resource considered
- Model has a host of parameters
- **Main conclusions**:
  - Co-allocation is beneficial when the **extension factor ≤ 1.20**
  - **Unlimited co-allocation is no good**:
    - limit the number of job components
    - limit the maximum job-component size
  - **Give local jobs some** but not absolute **priority** over global jobs

See, e.g.: A.I.D. Bucur and D.H.J. Epema, "Scheduling Policies for Processor Co-Allocation in Multicluster Systems," *IEEE Trans. on Parallel and Distributed Systems*, Vol. 18, pp. 958-972, 2007.

# DAS-3



Operational: oct. 2006

272 AMD Opteron nodes
792 cores, 1TB memory
**Some heterogeneity:**

- 2.2-2.6 GHz
- single/dual core nodes
- Myrinet-10G (excl. Delft)
- Gigabit Ethernet

Fourth generation on the way!

# DAS3: Characteristics

| location | Nodes (#) | Speed (GHz) | interconnect |
|---|---|---|---|
| Vrije Universiteit | 85 | 2.4 | Myri-10G & GbE |
| Amsterdam (1) | 41 | 2.2 | Myri-10G & GbE |
| Delft | 68 | 2.4 | **GbE** |
| Amsterdam (2) | 46 | 2.4 | Myri-10G & GbE |
| Leiden | 32 | **2.6** | Myri-10G & GbE |

# DAS3: measured network performance

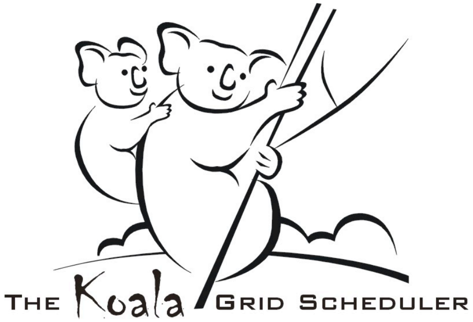
- Legend:

bandwidth (MB/s)
latency (ms)

| Cluster | VU | A'dam 1 | Delft | A'dam 2 | Leiden |
|---|---|---|---|---|---|
| VU | 561<br>0.03 | 185<br>0.4 | 45<br>1.15 | 185<br>0.4 | 77<br>1.0 |
| A'dam 1 | 185<br>0.4 | 526<br>0.03 | 53<br>1.1 | 512<br>0.03 | 115<br>0.6 |
| Delft | 45<br>1.15 | 53<br>1.1 | 115<br>0.05 | 10<br>1.45 | -<br>- |
| A'dam 2 | 185<br>0.4 | 512<br>0.03 | 10<br>1.45 | 560<br>0.03 | 115<br>0.6 |
| Leiden | 77<br>1.0 | 115<br>0.6 | -<br>- | 115<br>0.6 | 530<br>0.03 |

# KOALA: a Co-Allocating grid scheduler

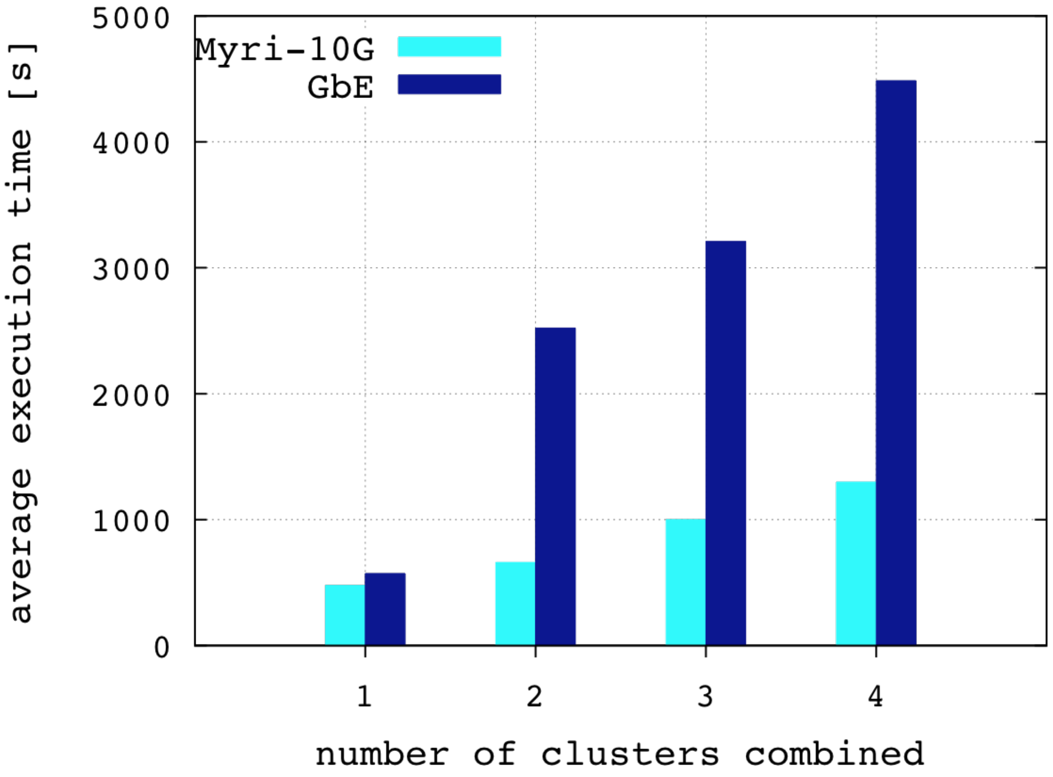
- Main goals:
  1. **processor co-allocation**: (un)ordered/flexible jobs
  2. **data co-allocation**: move large input files to the locations where the job components will run prior to execution
  3. **load sharing**: in the absence of co-allocation
  4. **run alongside local schedulers**
- **KOALA**
  - is written in Java
  - is middle-ware independent
  - has been deployed on the DAS2 and DAS3 since september 2005



See H.H. Mohamed and D.H.J. Epema, "The KOALA Co-allocating Grid Scheduler," *Concurrency and Computation, Practice and Experience Systems*, Vol. 20, pp. 1851-1876, 2008.

# Performance of Co-allocation: network

- Synthetic MPI application with all-to-all communication
- Fixed job requests
- Equal job component sizes



See O. Sonmez, H. Mohamed, and D.H.J. Epema, On the Benefit of Processor Co-Allocation in Multicluster Grid Systems, *IEEE Trans. on Parallel and Distributed Systems*, to appear.

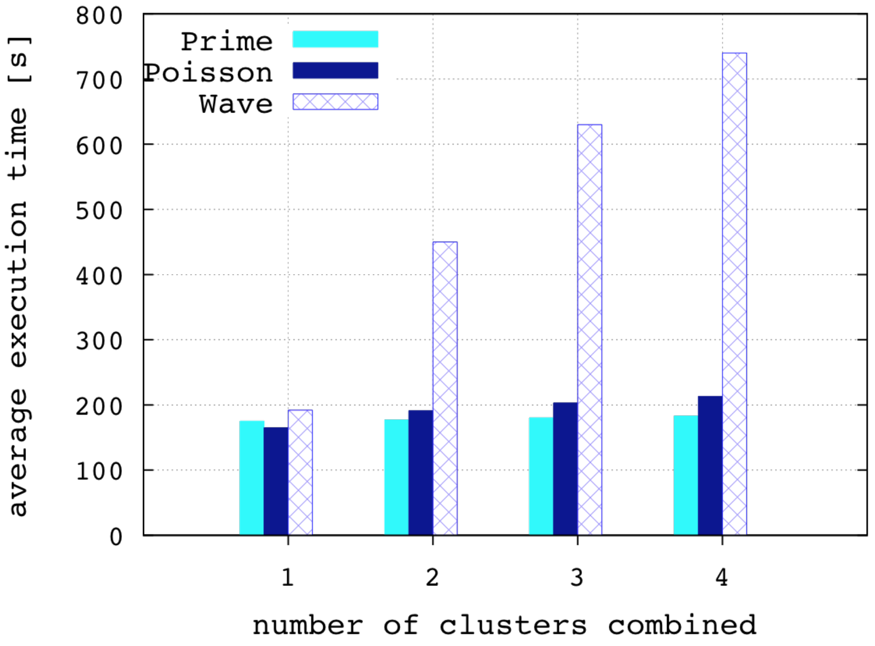
# Performance of Co-allocation: processor speed

- Synthetic application: MPI initialization plus floating point operations

| **clusters** | **Leiden** | **Leiden +VU** | **Leiden +Delft** | **Leiden +A'dam 1** | **Leiden +A'dam 2** |
|---|---|---|---|---|---|
| exec. time (s) | 30 | 32 | 32 | 32 | 35 |
| increase (%) | - | 7 | 7 | 7 | 17 |

# Performance of Co-allocation: communication

- Three applications:
  - Prime (hardly any communication)
  - Poisson (differential equation)
  - Wave (communication-intensive)
- Delft excluded, Myri-10G
- Fixed job requests
- Job components of equal size



TU Delft

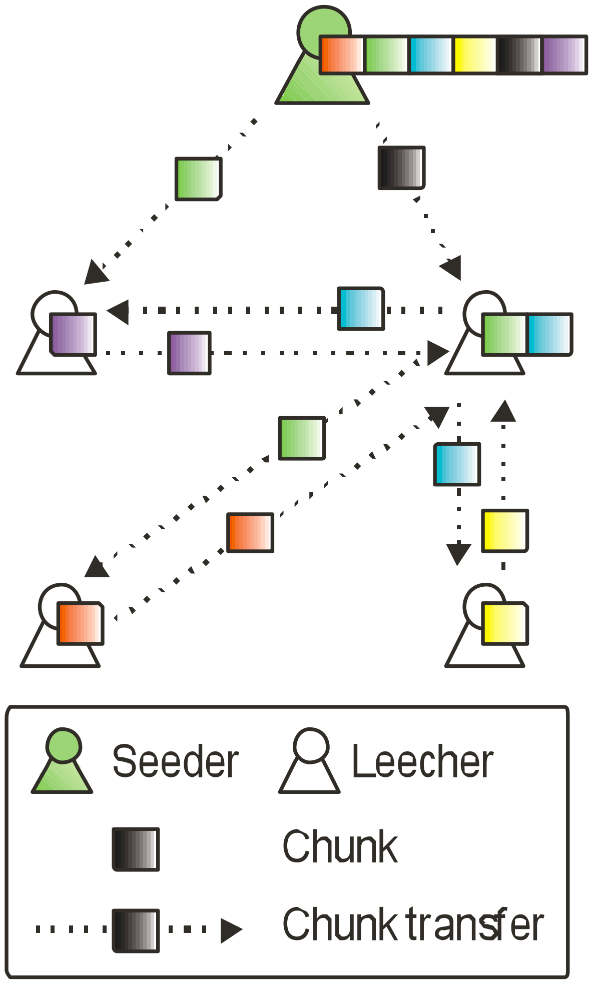
# **The Bittorrent P2P File Sharing System:** Measurements and Analysis

**Johan Pouwelse, Paweł Garbacki,**
**Dick Epema, Henk Sips**

See J.A. Pouwelse, P. Garbacki, D.H.J. Epema, and H.J. Sips, The BitTorrent P2P File-Sharing System: Measurements and Analysis, *4th Int'l Workshop on Peer-to-Peer Systems* (IPTPS'05).

**august 25, 2009**

# Data distribution model in BT



File divided into **chunks**

**Swarming** – groups of peers downloading the same file

**Seeders** – peers with the complete file

**Leechers** – peers whose download is in progress

Chunks exchanged between peers according to **tit-for-tat** strategy (rarest-first)

IP addresses of other peers obtained from a **tracker**

# BT web site: Suprnova.org

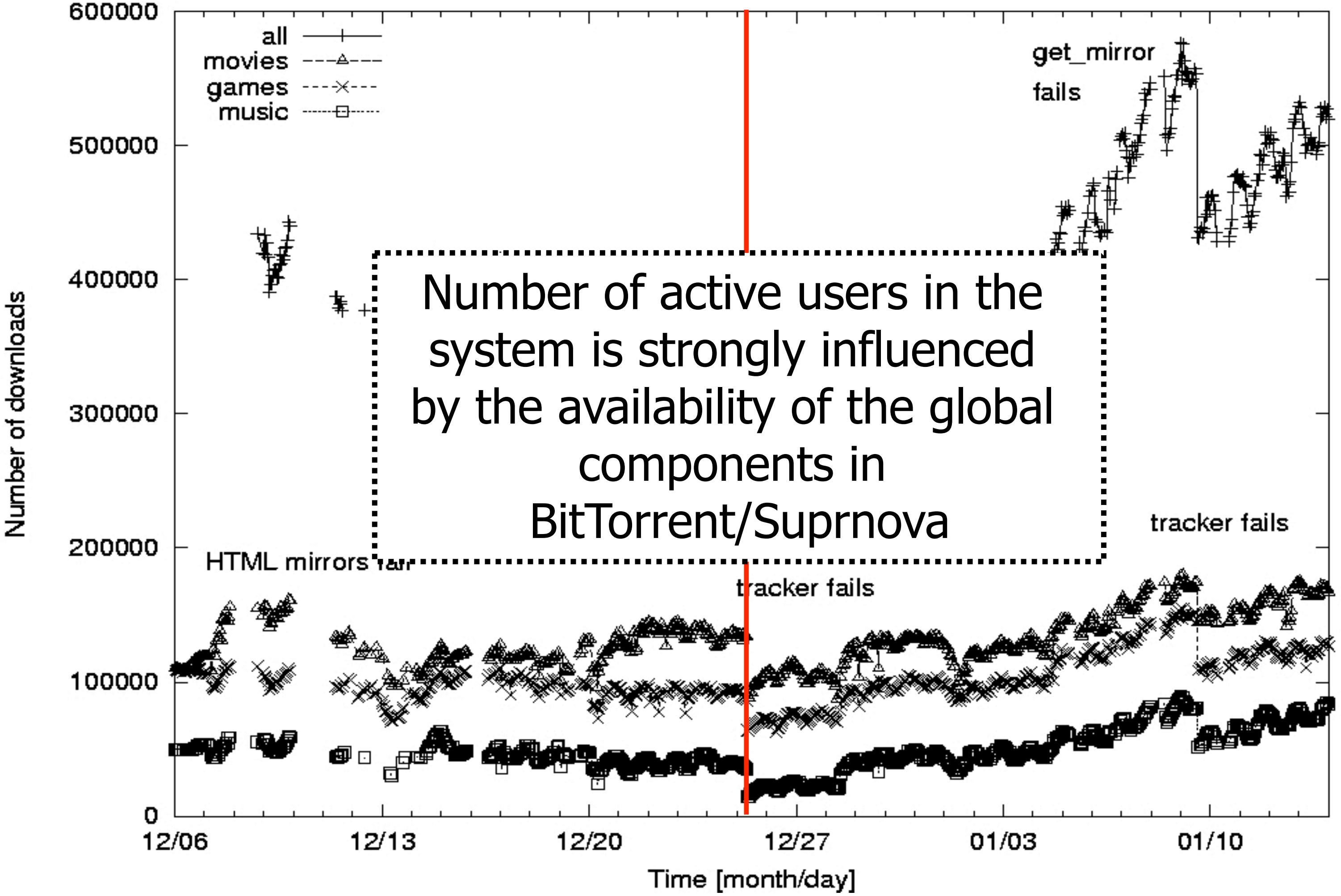
- At the time of performing the measurements the **most popular** .torrent distribution web site
  - 50,000 available files
  - 2,300,000 concurrent file transfers
- Used **mirroring** for load balancing
- **.torrent files** distributed among a number of **file servers**
- .torrent files point at **trackers**

- … went down in December 2004

# Some statistics of experiments

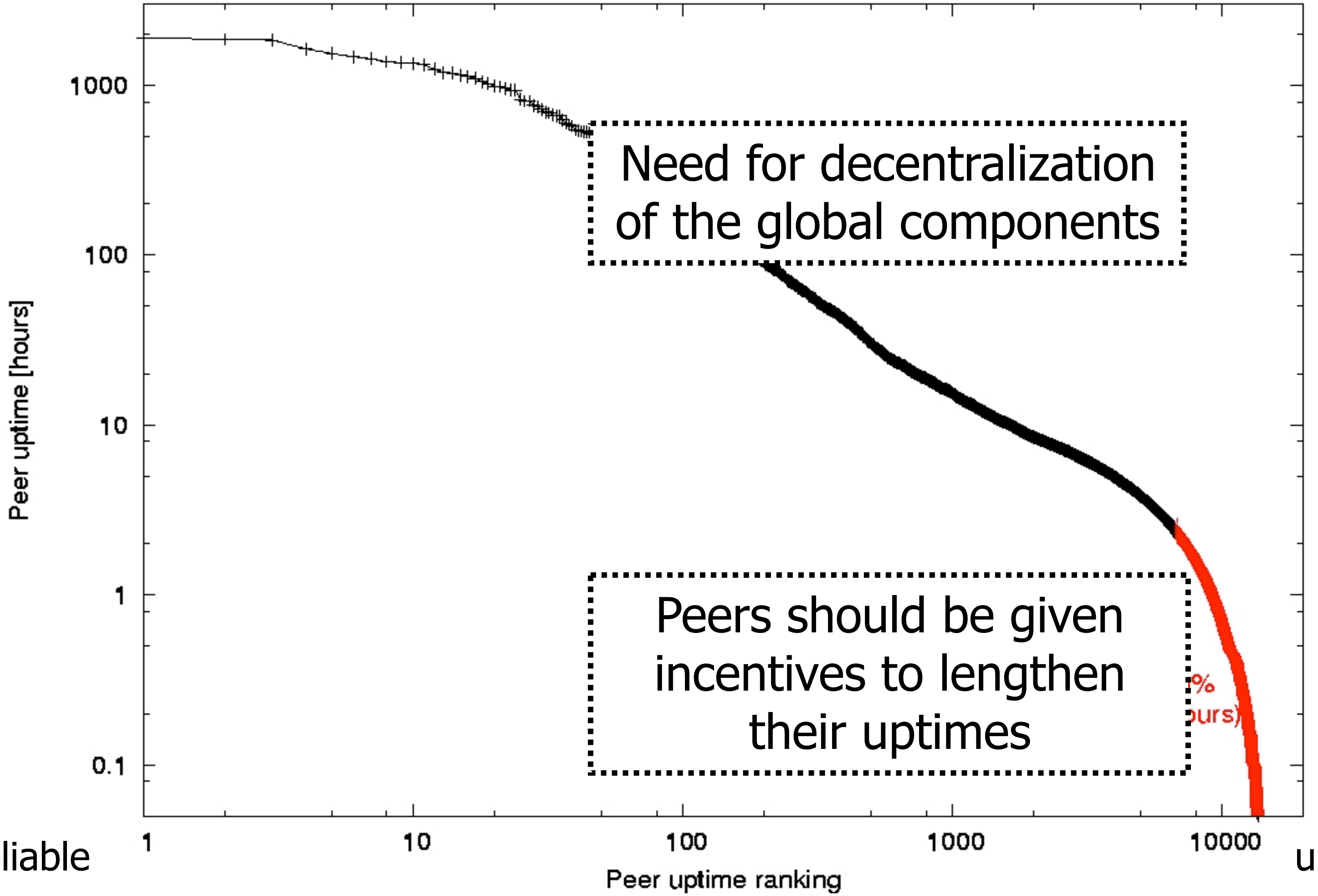
- **100** DAS2 nodes (1-Ghz Pentium-IIIs, 1 GB RAM)
- **8-month** traces of more than 2,000 global components
- **Complete lifetime** of a popular file (90,000 peers)
- Bandwidth measurement of **55,000 peers**
- **150 GB** of collected data

# Overall system activity



Number of active users in the system is strongly influenced by the availability of the global components in BitTorrent/Suprnova

# Uptime



Need for decentralization of the global components

Peers should be given incentives to lengthen their uptimes

reliable

unreliable

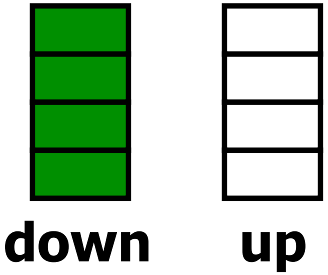
# 2Fast: Collaborative Downloads in File-Sharing Peer-to-Peer Networks

**Paweł Garbacki, Alexandru Iosup, Dick Epema,**
**and Maarten van Steen (VU)**

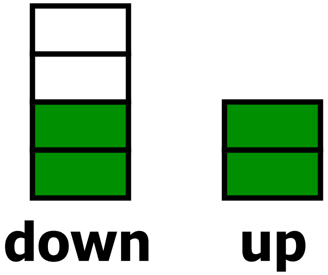
See P. Garbacki, A. Iosup, D.H.J. Epema, and M. van Steen, "2Fast: Collaborative Downloads in P2P Networks," *6-th IEEE International Conference on Peer-to-Peer Computing*, 2006.

# Peer-to-peer data transfer protocols

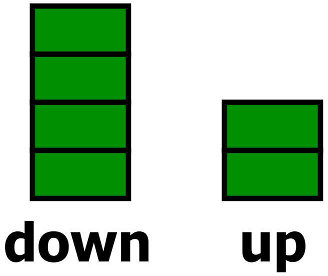
- Gnutella, Kazaa
  - no incentives for bandwidth sharing
  - free-riders sensitive
  - poor utilization of upload bandwidth



- BitTorrent (BT), Slurpie
  - tit-for-tat enforces fairness
  - temporal fairness cannot handle asymmetric links
  - poor utilization of download bandwidth

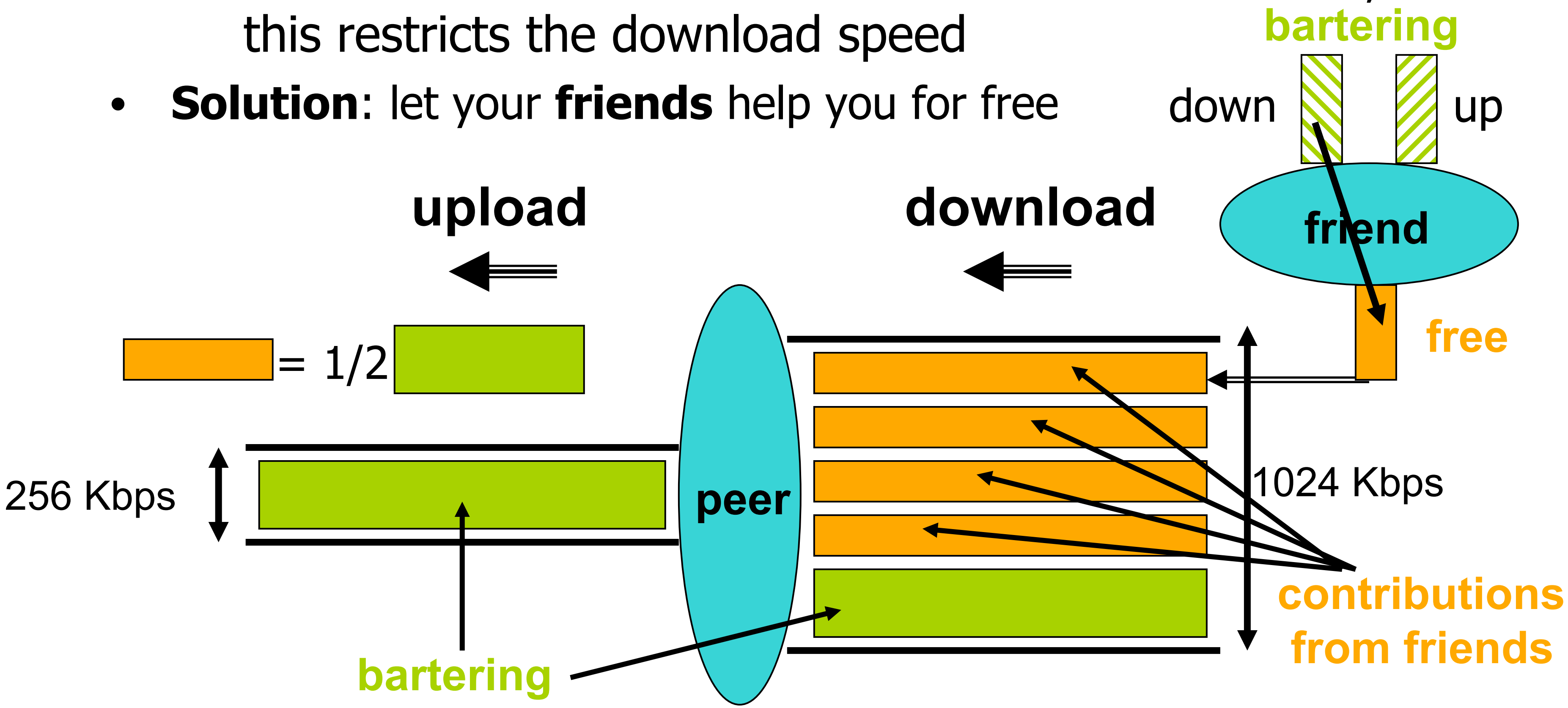


- **2Fast: BT+collaborative downloads**
  - no tit-for-tat within a single session
  - cross-session bandwidth sharing
  - full utilization of upload AND download links

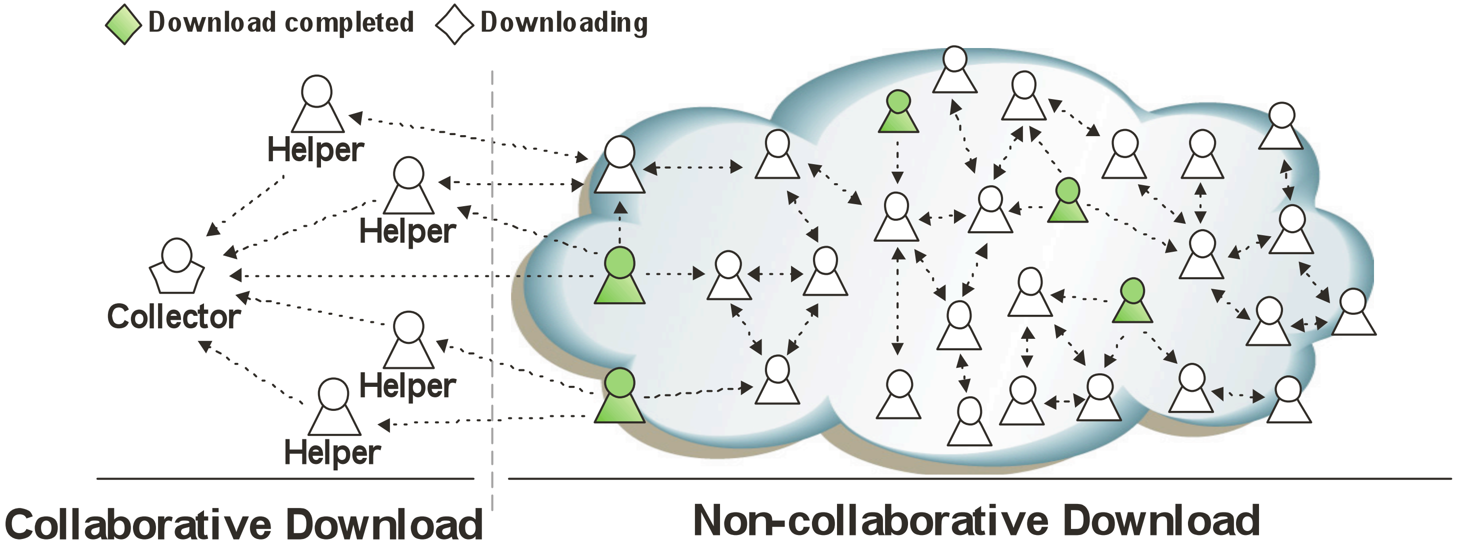


# Cooperative downloads: basic idea

- **Problem**:
  - most users have **asymmetric** upload/download links
  - because of the **tit-for-tat** mechanism of Bittorrent, this restricts the download speed
- **Solution**: let your **friends** help you for free



# Collaborative downloads: another view



- **Collaboration** established between collector and helpers
- **Collector** aims at obtaining a complete copy of the file
- **Helpers** download distinct chunks and send them to the collector, not requesting any other chunk in return

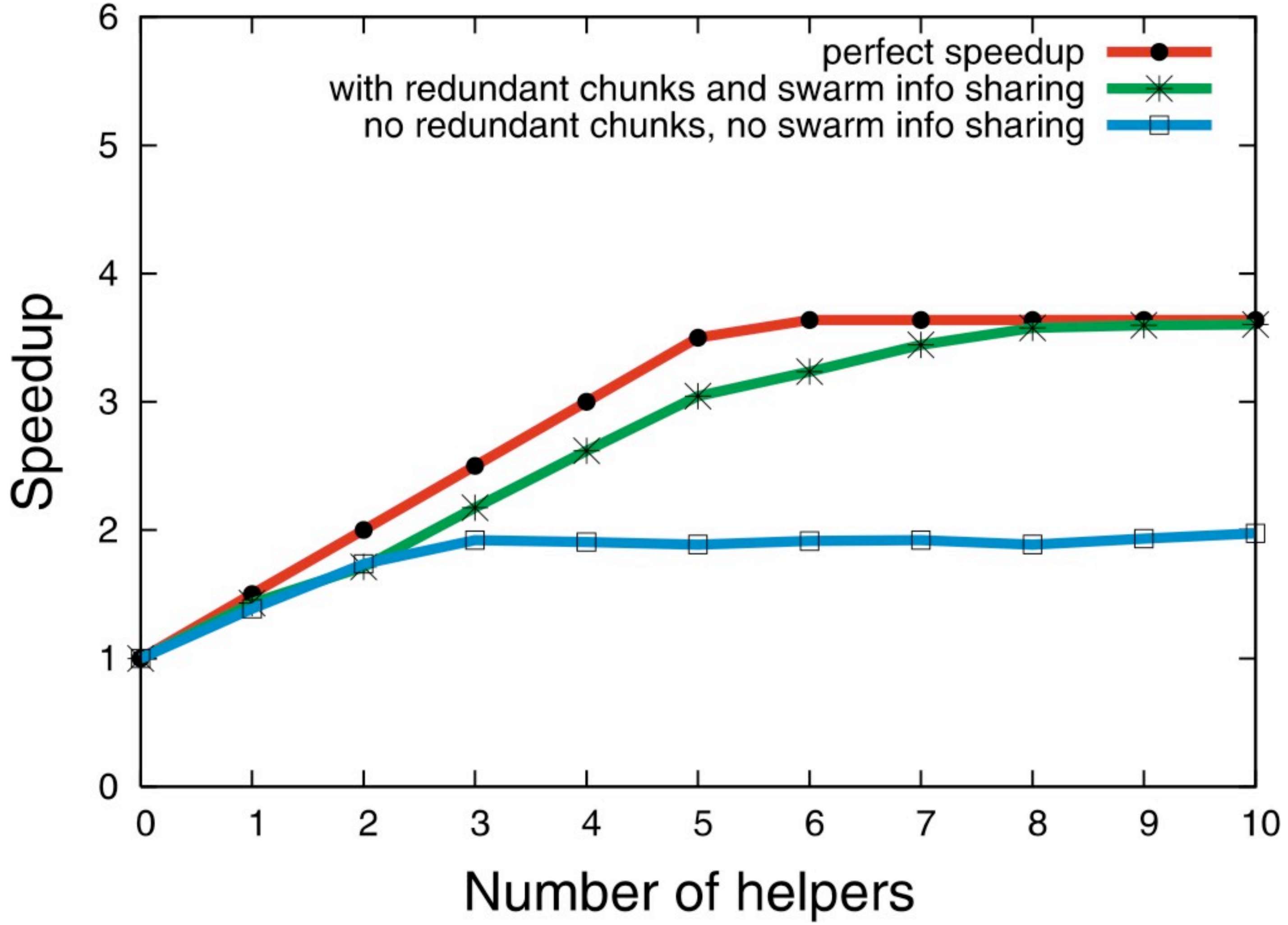
# Two protocol extensions

- **Redundant chunks download**
  - **problem**: helpers download different chunks; more restrictive chunk selection + fewer chunks to offer, so limited bartering possibilities
  - **solution**: the same chunk may be downloaded by different helpers
- **Sharing of swarm information**
  - **problem**: slow start; finding suitable bartering partners takes time
  - **solution**: collaborating peers exchange information on other peers in the swarm

# Experimental setup

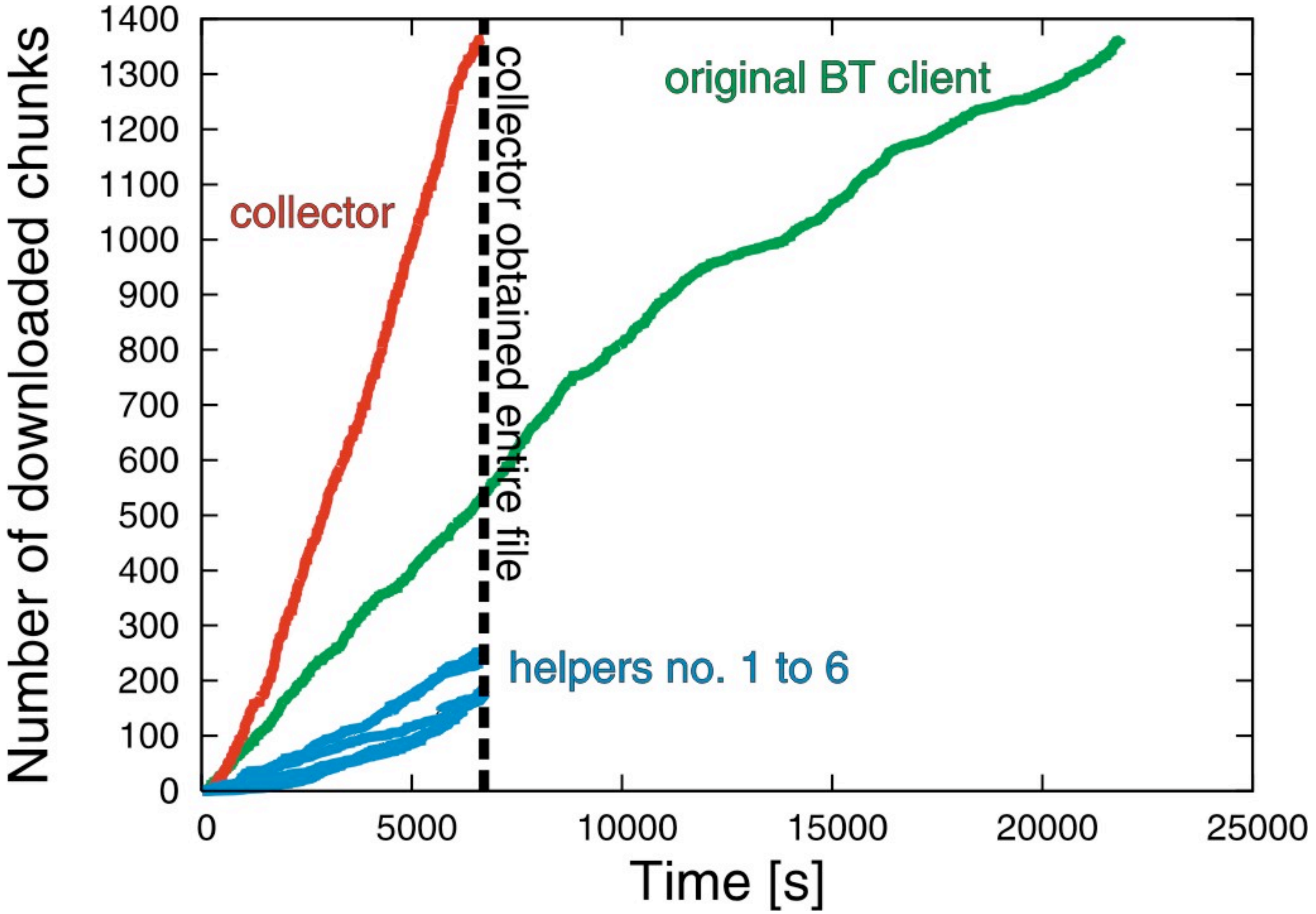
- Experiments performed in a real environment – collaborating peers connect to existing BitTorrent swarms
- Collaborating peers connected through ADSL links: 256kbps up / 1024kbps down
- Downloaded file size: 700MB
- Swarm size: 100 leechers, 10 seeders

# Speedup

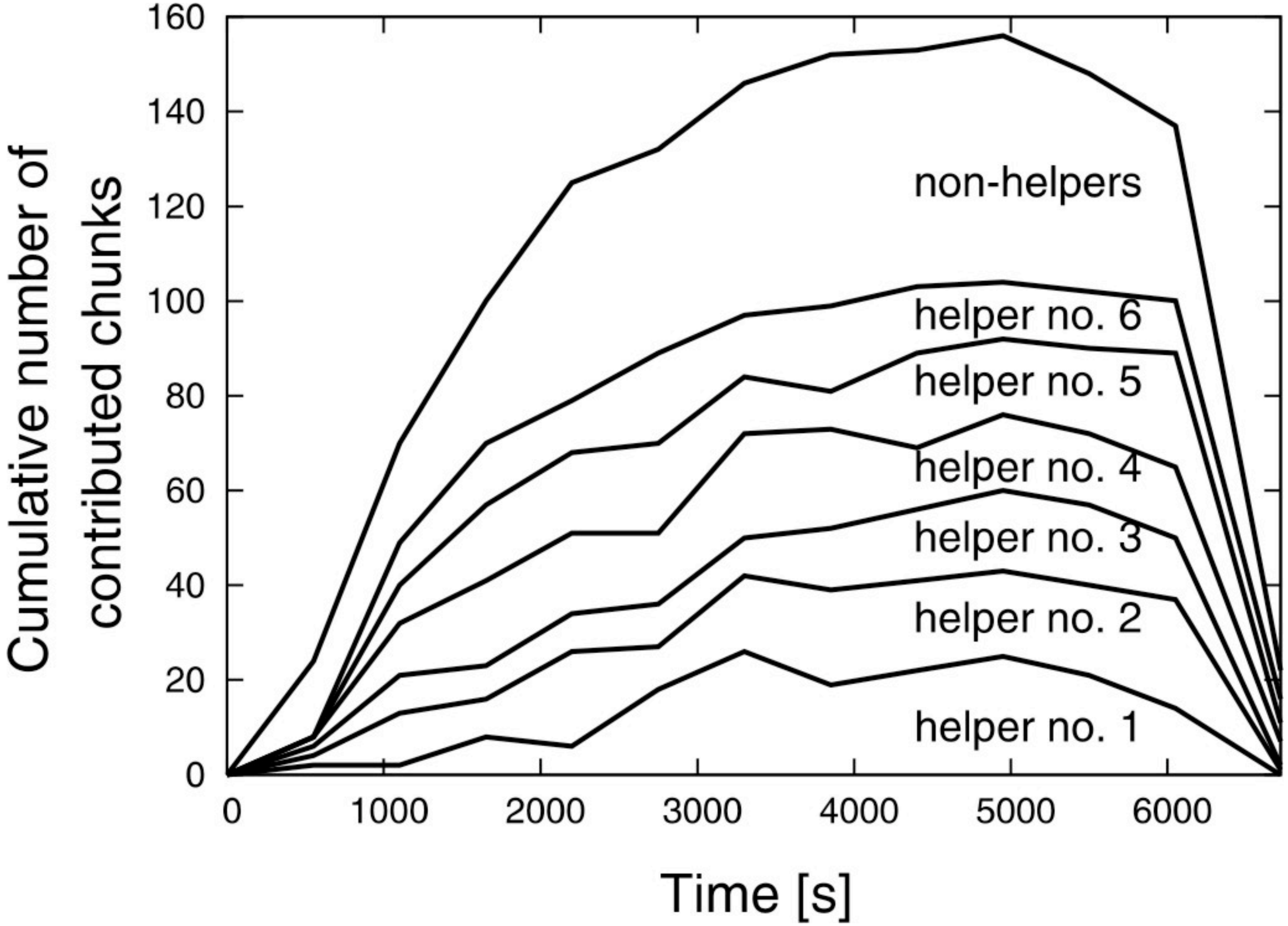


perfect speedup
with redundant chunks and swarm info sharing
no redundant chunks, no swarm info sharing
Speedup
0
1
2
3
4
5
6
0
1
2
3
4
5
6
7
8
9
10
Number of helpers

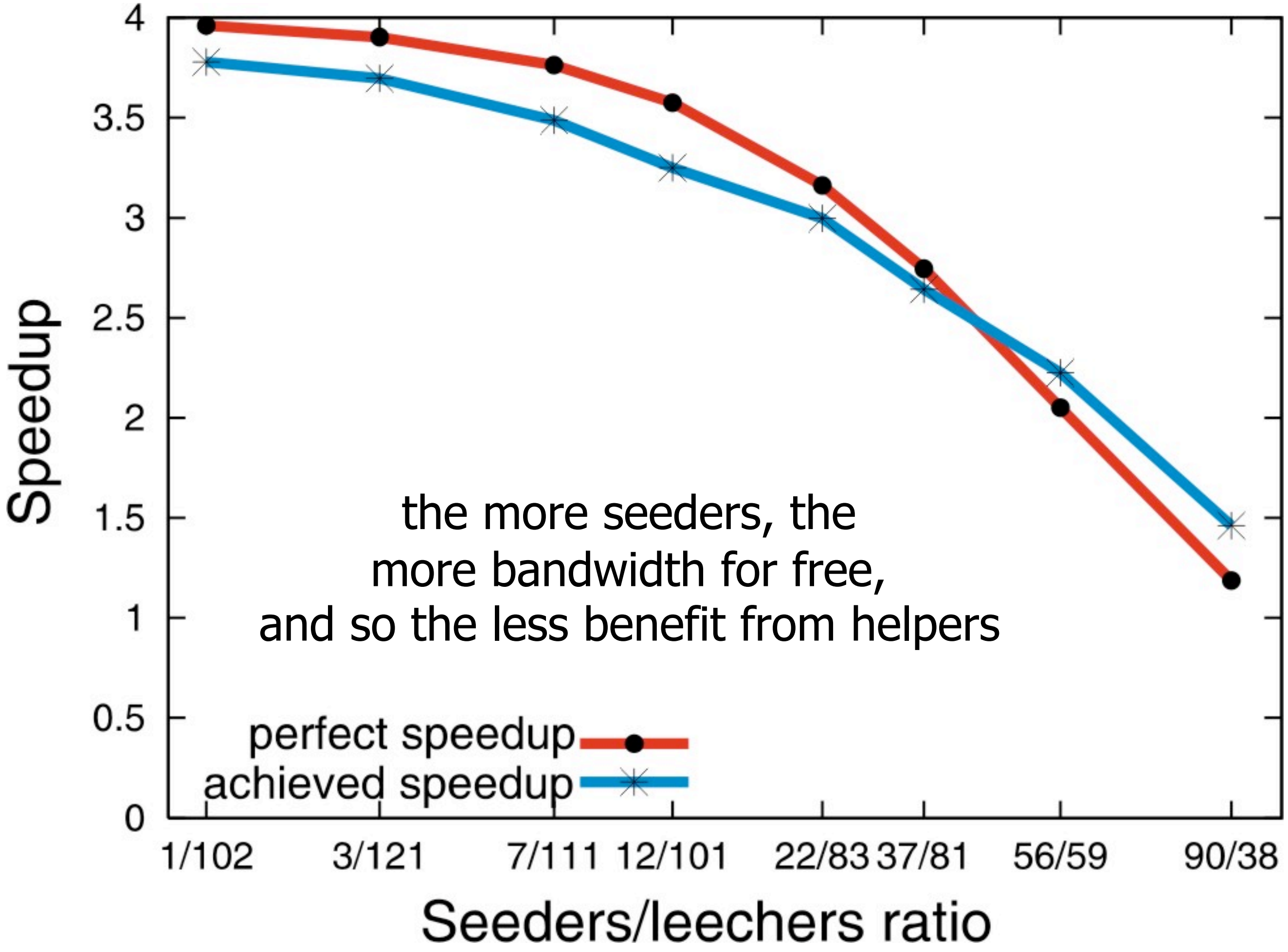
# Download progress



# Peer contributions



# Seeders/leechers ratio



the more seeders, the
more bandwidth for free,
and so the less benefit from helpers

# Optimizing Peer Relationships in a Super-Peer Network

**Paweł Garbacki, Dick Epema, and**

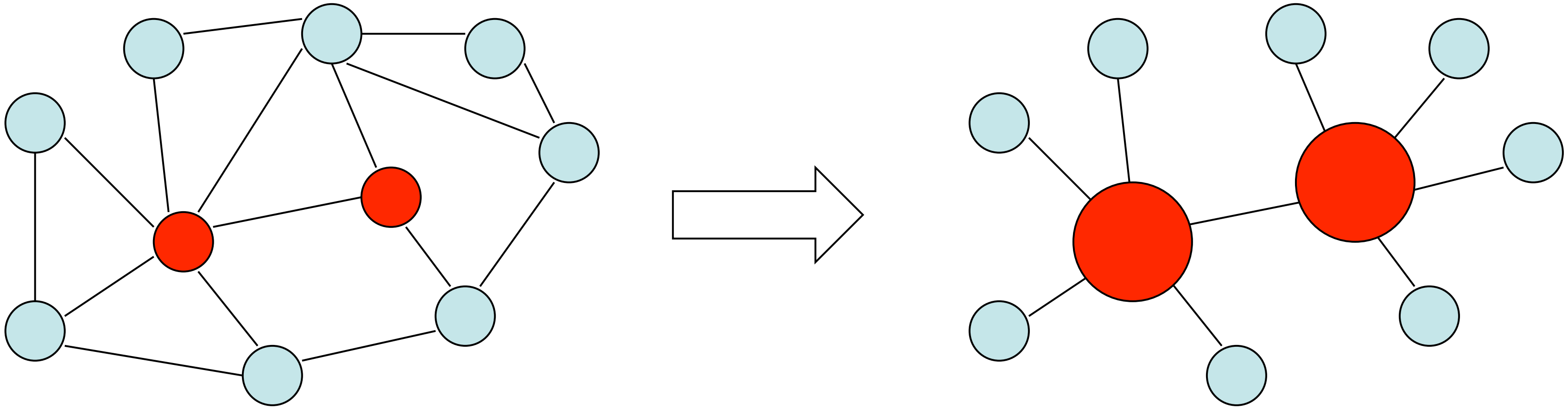
**Maarten van Steen (VU)**

See

1. P. Garbacki, D.H.J. Epema, and M. van Steen, "Optimizing Peer Relationships in a Super-Peer Network," *Int'l Conference on Distributed Computing Systems* (ICDCS), June 2007.
2. P. Garbacki, D.H.J. Epema, and M. van Steen, "The Design and Evaluation of a Self-Organizing Super-Peer Network, *IEEE Trans. on Computers*, to appear.

**august 25, 2009**

# Super-peer network

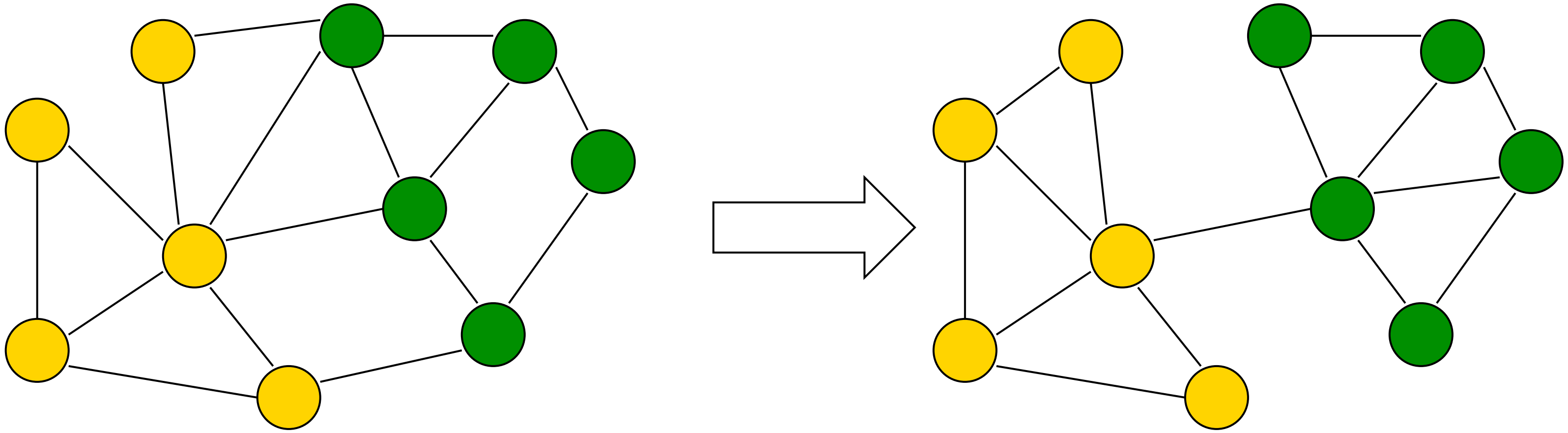


- Observation: peers vary in availability, bandwidth, processing power, etc.
- Create network backbone from highly available and powerful super-peers
- Super-peer acts as centralized servers to a subset of weak peers

# Limitations of existing super-peer networks

1. Each weak peer is assigned to a small number (usually one) of super-peers
   - super-peers become bottlenecks in terms of fault tolerance
2. Weak peers are assigned to super-peers statically and randomly
   - no adaptation to changes in network structure and peer interests
3. All-or-nothing peer-to-super-peer assignment
   - load balancing is difficult

# Semantic clustering



- Users in P2P network share interests and have files in common
- Can we cluster them according to their interests and improve the performance?
  - semantic-based search

- Natural match:

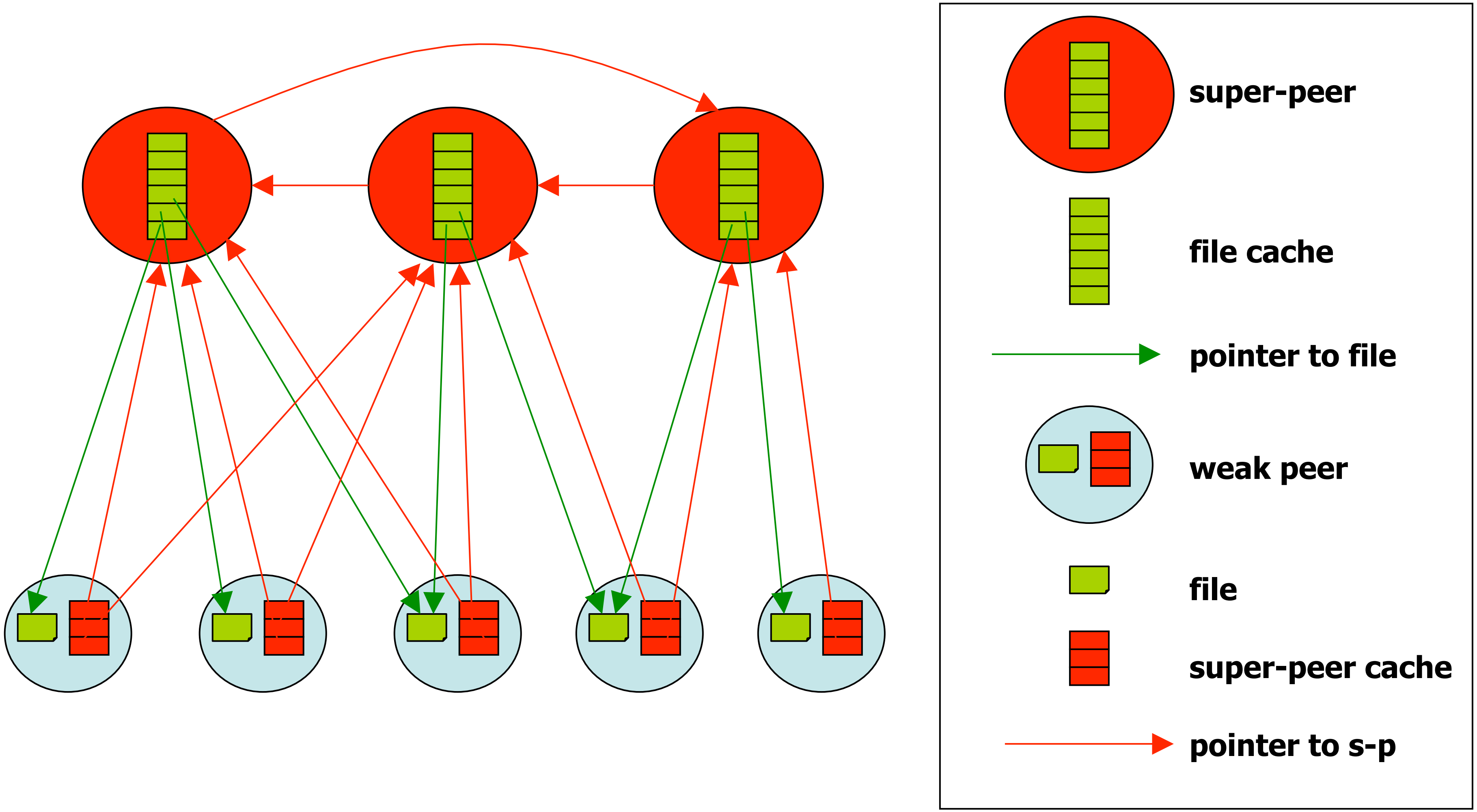
  semantic cluster = set of peers assigned to one super-peer

# Self-Organizing Super-Peer Network (SOSPNet)

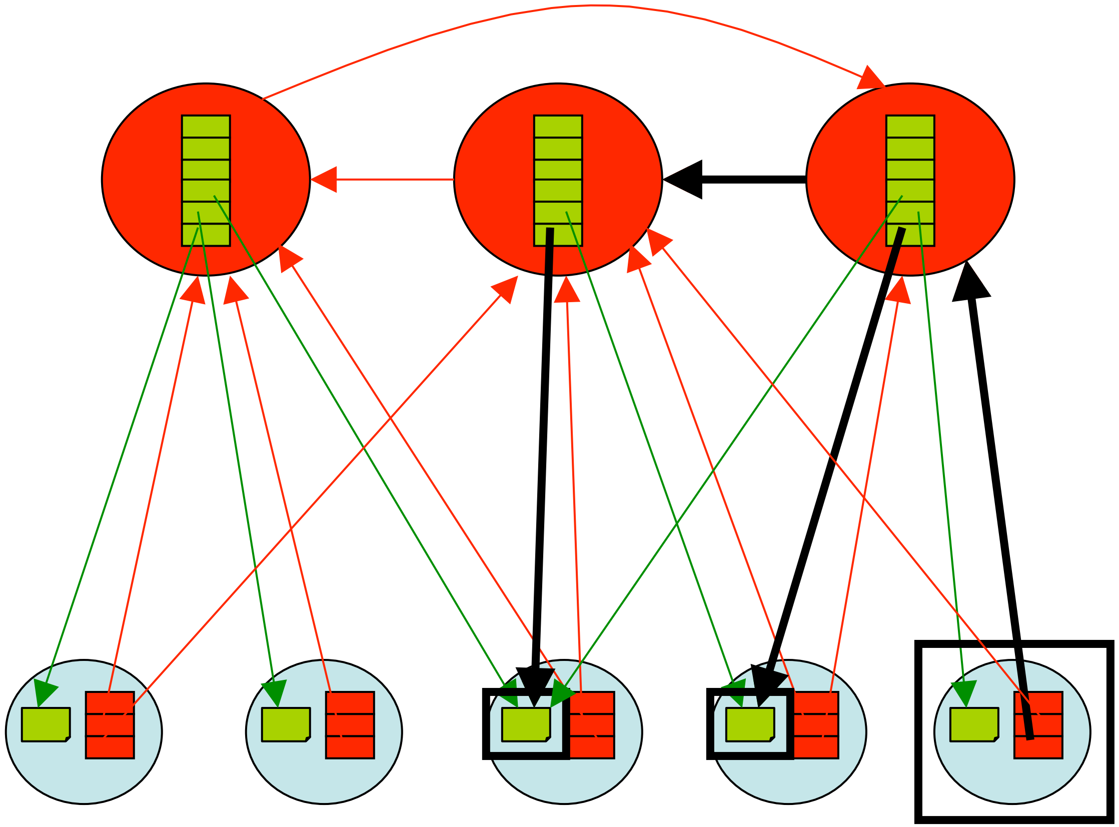
- Key design decisions
  - weak peer assigned to more than one super-peer
  - uses two types of caches to model semantic dependencies between peers and between content
  - super-peers group files, not peers

- Properties
  - super-peers group semantically correlated files
  - semantically correlated peers contact the same super-peers

# SOSPNet architecture

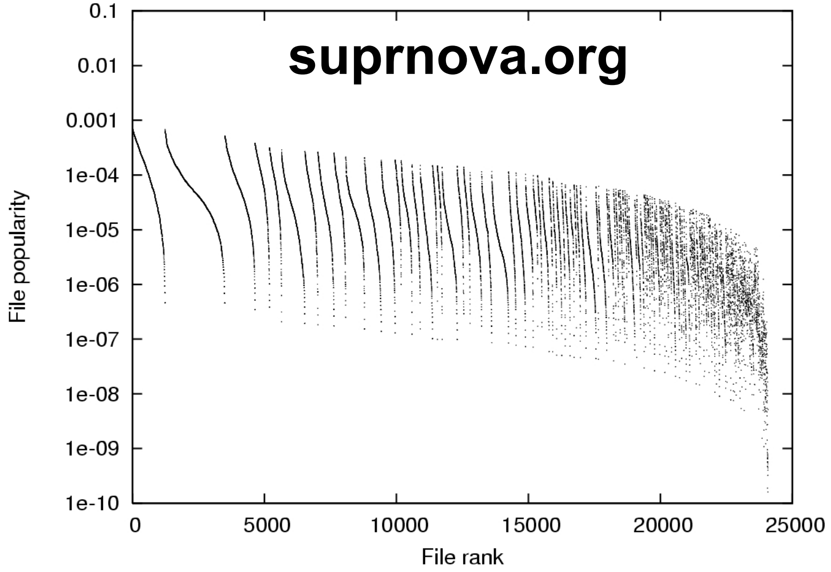


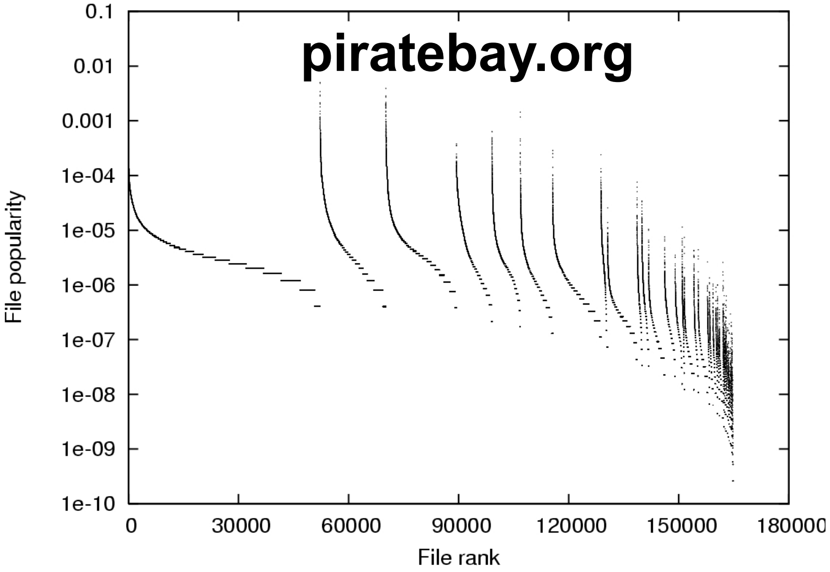
# Search protocol



# System model based on real traces

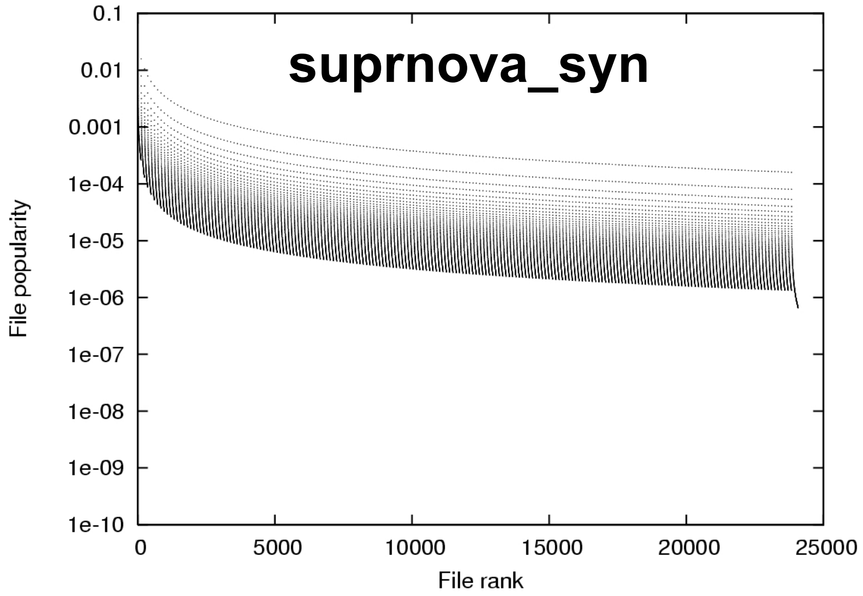
- 8-month trace data collected for two popular file sharing communities: suprnova.org and piratebay.org
- 24,081 suprnova.org and 164,821 piratebay.org files divided into 198 (suprnova.org) and 40 (piratebay.org) semantic types by moderators

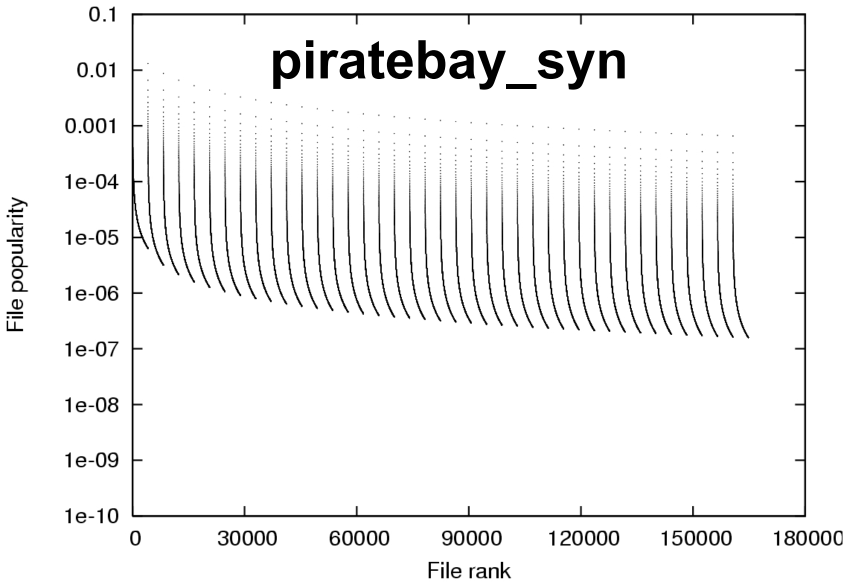




# Synthetic system model

- Number of files and semantic types the same as in the trace-based model (for comparison)
- Number of files of each type is the same
- File popularities follow Zipf's distribution



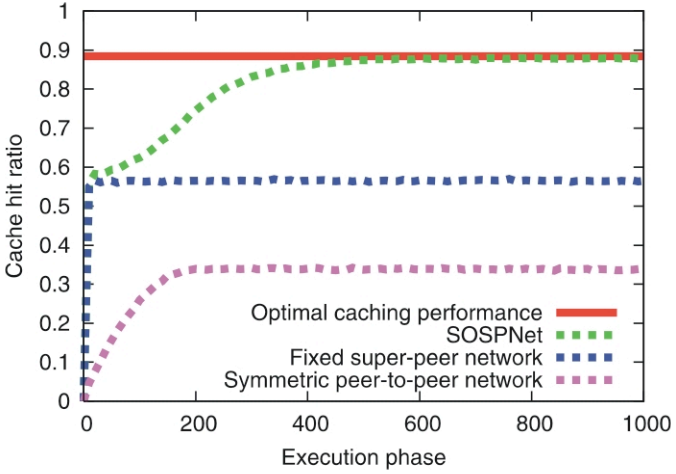


# Experimental evaluation

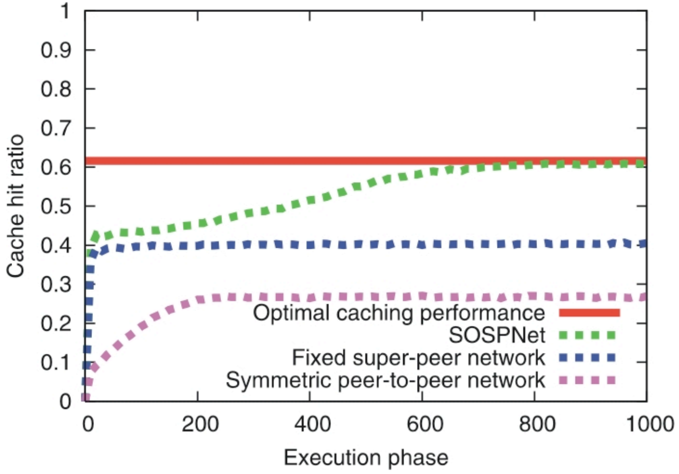
- 100,000 weak peers and 1,000 super-peers
- File caches of size 1,000 and super-peer caches of size 10
- Peers divided into **semantic types** request files with distribution biased towards their semantic type
- Simulation performed in **phases**
    - in each phase every weak peer generates a search request
    - target file of the request is selected based on file popularity
- For comparison:
    - **symmetric network** of peers with one-level caches of size 40
    - **traditional fixed super-peer network** where weak peers do not dynamically change super-peers

# Caching performance

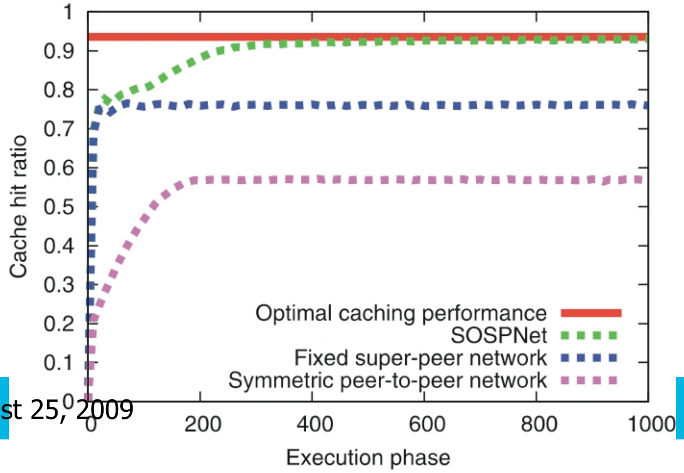
suprnova.org



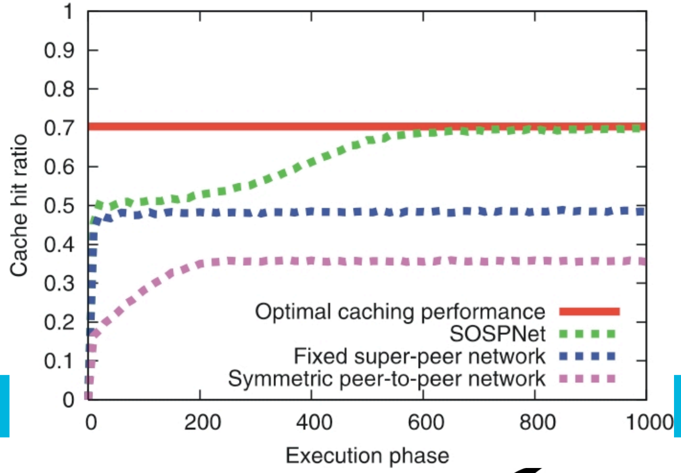
piratebay.org



suprnova_syn



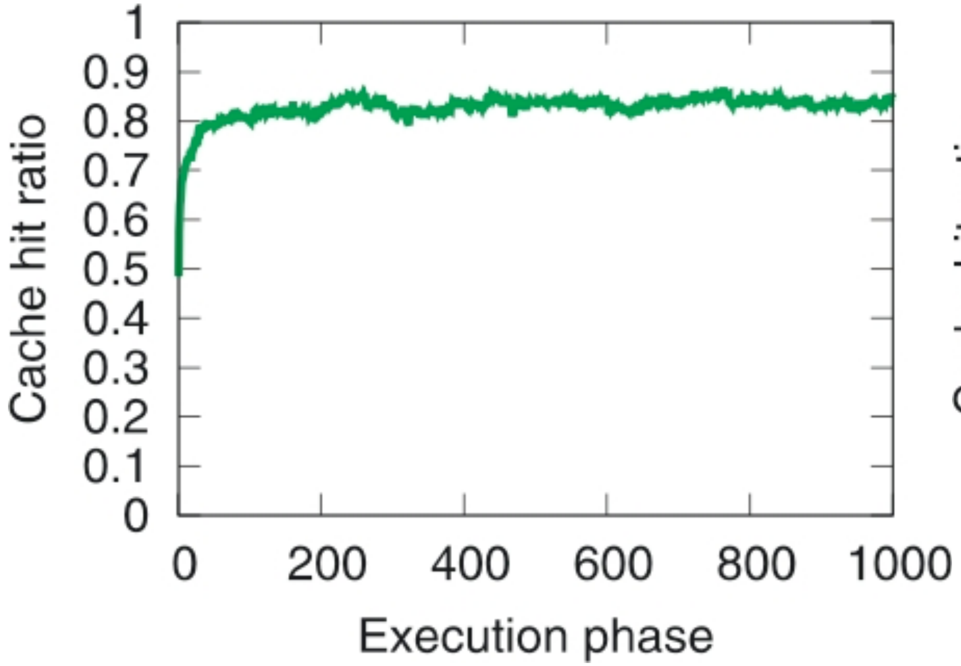
piratebay_syn



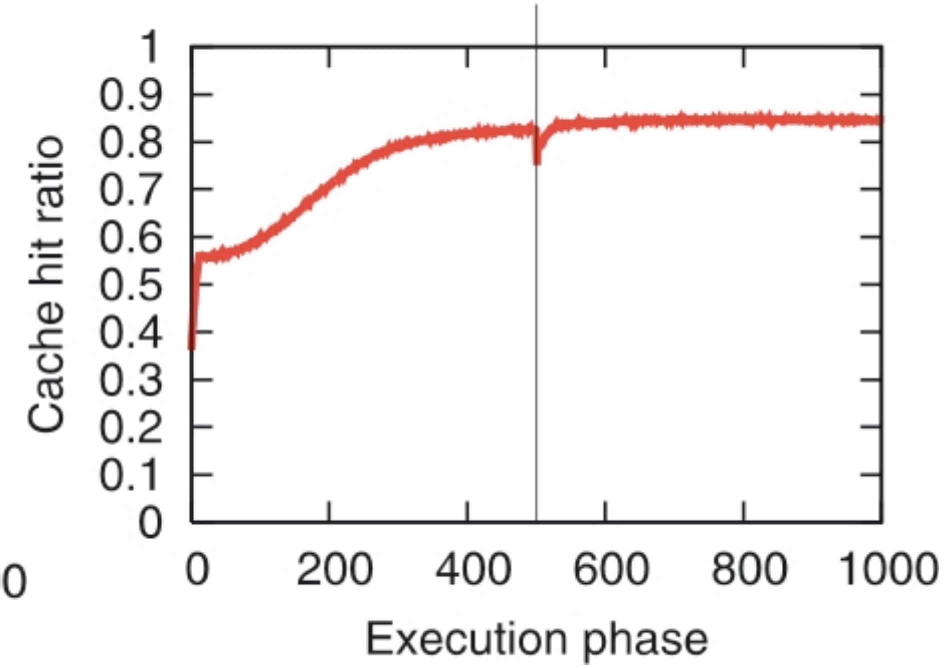
# Peer joins and leaves

**suprnova.org**

New peer joining



50% of super-peers and
50% of weak peers fail in phase 500



# Clustering of files and peers

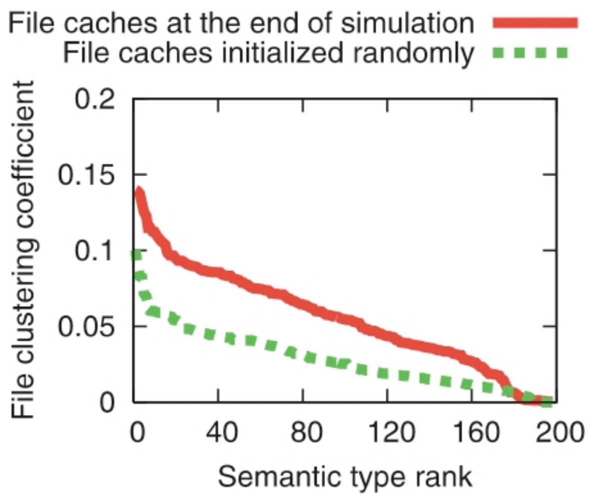
- File clustering coefficient – average of the Jaccard's coefficients of pairs of files of the same semantic type

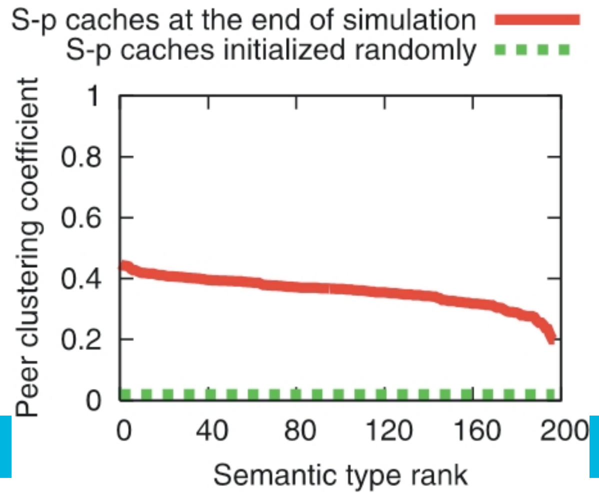
Jaccard's coefficient :

$$J(f_1, f_2) = \frac{|Q(f_1) \cap Q(f_2)|}{|Q(f_1)| + |Q(f_2)|}$$

$Q(f_i)$ is the set of super-peers that have a pointer to $f_i$ in their file cache

- Peer clustering coefficient – average number of identical items in the s-p caches of peers of one semantic type





# P2P Research in Delft

- Research topics:
  - Social-based features (friends, taste buddies)
  - Epidemic protocols for peer and content discovery
  - Mechanisms for all forms of video distribution (recorded, live, VoD)
  - Near-zero cost video distribution
- Research vehicle: the BitTorrent-based client **Tribler**
- Group of about 15 people
- EU FP7 IP P2P-Next

# Information

- **Publications**
  - see PDS publication database at www.pds.ewi.tudelft.nl
- **Web sites**:
  - Projects: www.pds.ewi.tudelft.nl/~epema
  - KOALA: www.st.ewi.tudelft.nl/koala
  - DAS3: www.cs.vu.nl/das3
  - VL-e: www.vl-e.nl
  - Tribler: www.tribler.org