Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

Institute of Computer Science
Chair of Communication Networks
Prof. Dr.-Ing. P. Tran-Gia

The Impact of Caching on BitTorrent-Like Peer-to-Peer Systems

Best Paper Award at IEEE P2P 2010 in Delft, Netherlands

Frank Lehrieder[1], György Dán[2], Tobias Hoßfeld[1], Simon Oechsner[1], Vlad Singeorzan[1]

[1]University of Würzburg, Germany
[2]KTH Royal Institute of Technology, Stockholm, Sweden

euro nf
NETWORK OF EXCELLENCE

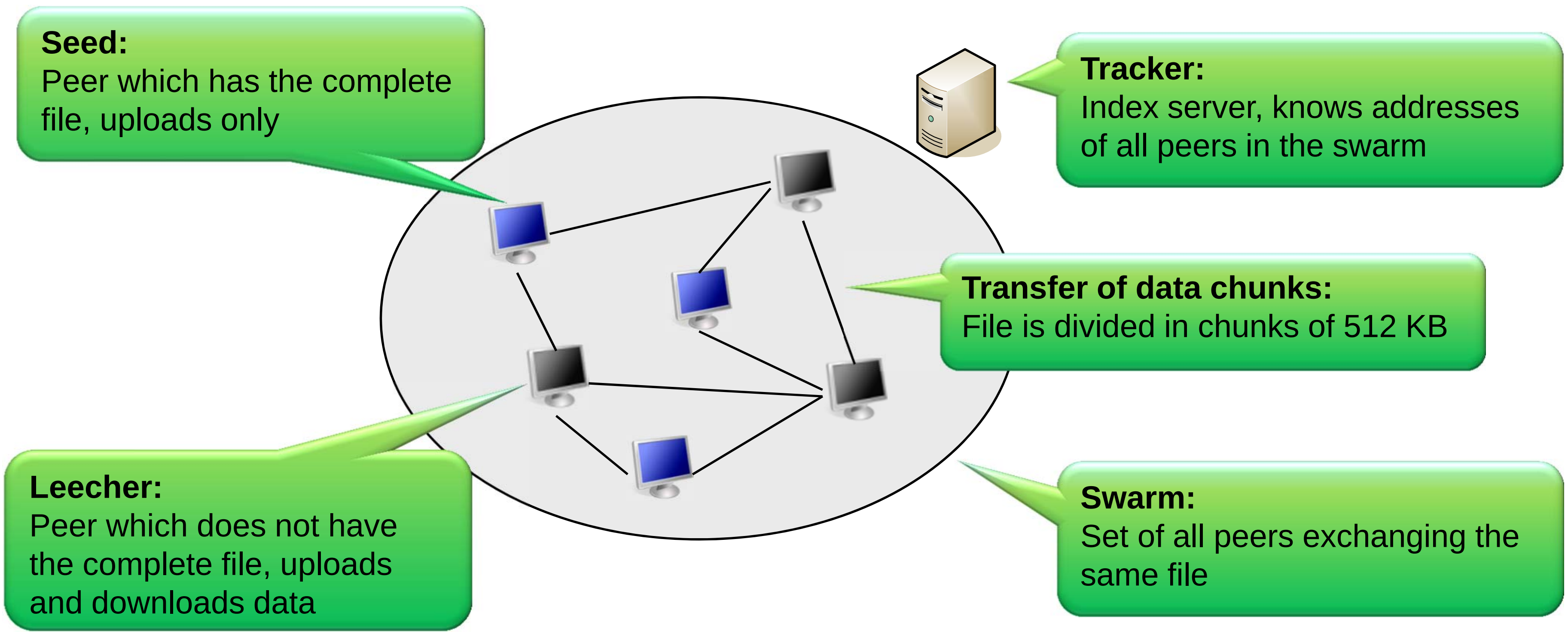
G Lab
www.german-lab.de

Agenda

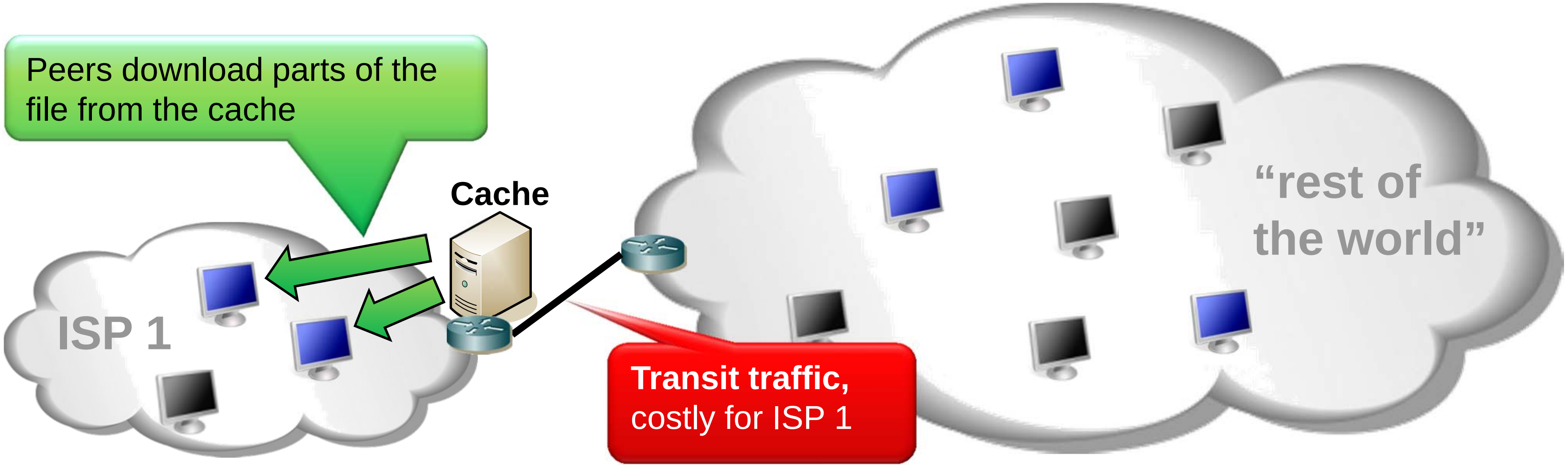
- Introduction
  - BitTorrent-like P2P networks
  - Caching in BitTorrent-like P2P networks
- Fluid model of caching
  - Number of peers
  - Transit traffic estimates
- Experimental and simulative validation
- Analytical results and insights
- Conclusion

BitTorrent-Like P2P Networks

- In wide use for user-assisted content distribution, mostly file-sharing
- Responsible for a large fraction (60%) of today's traffic in the Internet
- Example network:



Caching in BitTorrent-Like Networks



- Focus of the study: impact of caches on
  - Number of leechers and seeds
  - Transit traffic between different ISPs
  - Single swarm scenario: no storage replacement strategies
- Caches (e.g. OverSi's OverCache P2P)
  - Run BitTorrent protocol, appear as high capacity peers
  - Upload only to local leechers

A Fluid Model of Caching – Overview

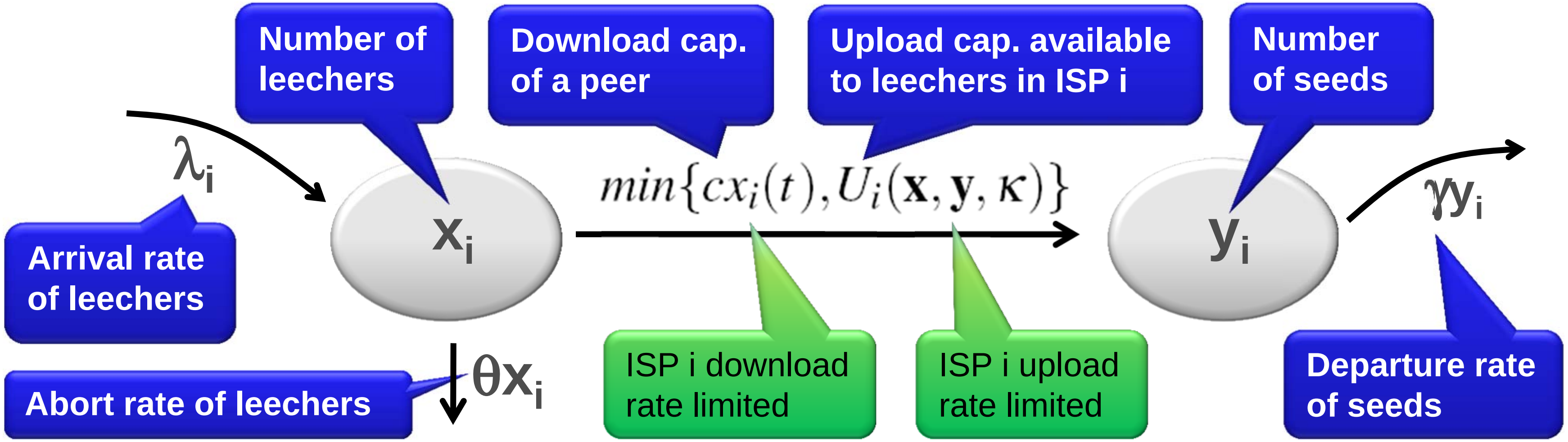
- Basis: fluid model of Qiu and Srikant (SigComm 2004)
  - Number of leechers and seeds in a BitTorrent swarm
  - Depending on arrival- and departure rates, up- and download capacities of the peers
  - Dynamics and steady state equations

- **Our extensions**
  - Multiple ISPs $i \in \{1,\dots,I\}$
  - Caches with upload capacities κ_i
  - Incoming and outgoing transit traffic of ISPs

- Road map
  - Model impact of caches on number leechers and seeds
  - Derive transit traffic estimates based on ISP affiliations of peers

System Dynamics

- Flow diagram for ISP i



- Fluid model

$$\frac{dx_i(t)}{dt} = \lambda_i - \theta x_i(t) - min\{cx_i(t), U_i(\mathbf{x}, \mathbf{y}, \kappa)\}$$

$$\frac{dy_i(t)}{dt} = min\{cx_i(t), U_i(\mathbf{x}, \mathbf{y}, \kappa)\} - \gamma y_i(t)$$

Steady State Solutions and Insights

- Steady state of the system: $\frac{dx_i(t)}{dt} = \frac{dy_i(t)}{dt} = 0 \quad i = 1, \ldots, I.$
- Analytical solutions for avg. number of **leechers x_i** and **seeds y_i** in ISP i

- Insights (derived from the equations)
  - Case 1: all ISPs upload rate limited
    - Cache in ISP i **decreases** avg. number of leechers x_i
    - Cache in ISP i **increases** the avg. number of seeds in ISP i if peers are impatient (θ>0)
  - Case 2: all ISPs download rate limited:
    - no impact on number of peers

- Supposed impact on transit traffic
  - Incoming transit traffic decreased
  - Outgoing transit traffic increased or decreased?

A Simple Model for Transit Traffic

- Model of **incoming and outgoing transit traffic** of the ISPs
  - Based on the number of leechers x_i and seeds y_i in the ISPs
  - Abstracts from inter-ISP delays, BitTorrent neighbor selection, and the choke algorithm

- **Incoming transit traffic estimate**

Incoming transit traffic of ISP i = **Total transfer rate in swarm (caches not included)** • **Fraction of leechers which are in ISP i** • **Fraction of upload capacity of peers outside ISP i**

- **Outgoing transit traffic estimate**

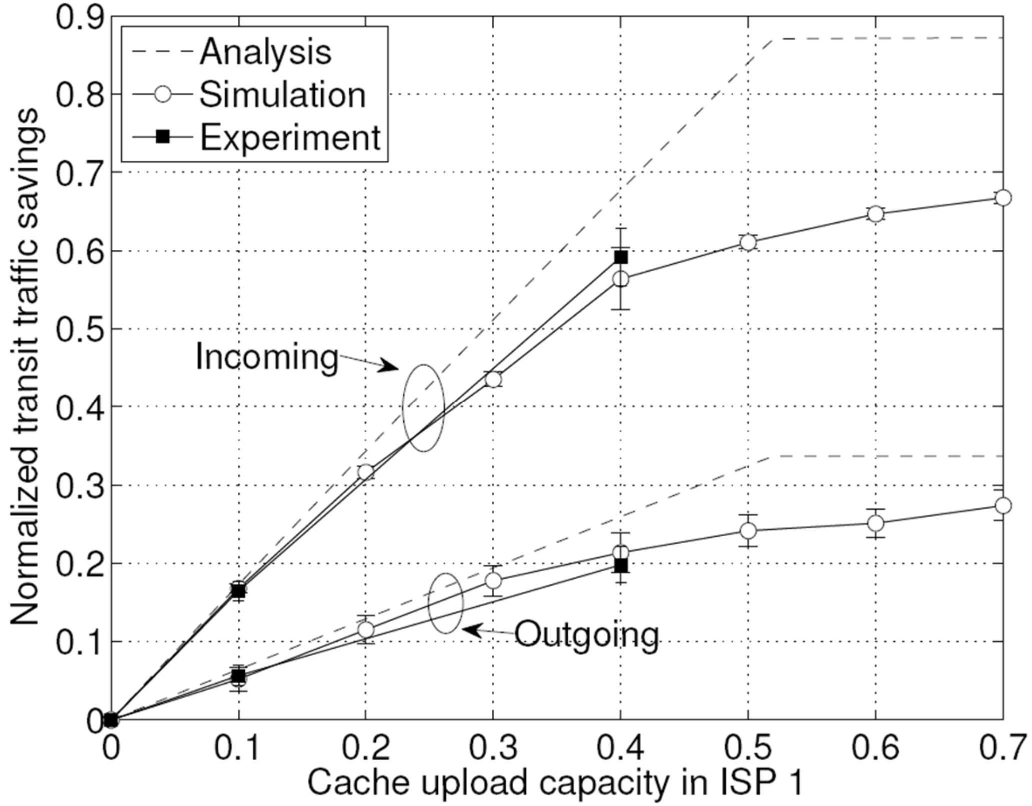
Traffic from ISP i to ISP j = **Incoming transit traffic of ISP j** • **Ratio of upload capacity of peers in ISP i to upload capacity of peers outside ISP j**

Validation of the Model: Methodology

- Simulator
  - Simulation framework ProtoPeer
  - BitTorrent library of ProtoPeer
  - 25 simulation runs per configuration
- Experimental facility: German-Lab
  - Around 160 nodes, distributed across 5 universities in Germany
  - BitTorrent mainline client (version 4.4.0)
  - 5 experiment runs per configuration
- Scenario
  - Two ISPs, ISP 2 is 10 times larger than ISP 1
  - Cache in ISP 1 with varying upload capacities
  - Around 120 peers concurrently online

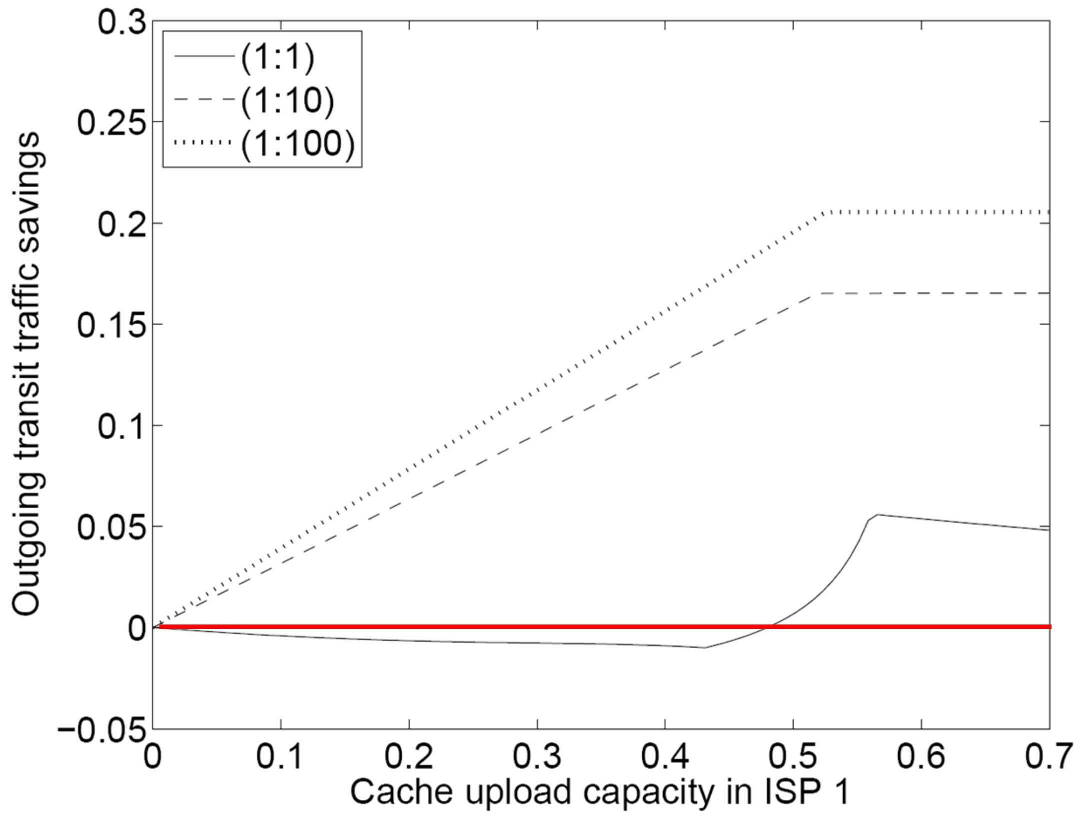
Validation of the Model: Transit Traffic

- Normalized transit traffic savings:
  fraction of traffic that can be saved by installing a cache
- Good match for outgoing traffic and incoming traffic with small cache capacities
- Incoming traffic savings overestimated (due to fluctuation of number of leechers)
- Even better match for larger swarms (see figures in the paper)



Analytical Results: Outgoing Transit Traffic

- Outgoing transit traffic savings wrt. to cache upload capacity
- Ratio of peer arrivals (ISP 1:ISP 2): (1:1), (1:10), (1:100)
- Caches more efficient when large fraction of the swarm outside ISP with cache
- Outgoing transit traffic may increase due to the cache
- Management of cache upload rates to different swarms required to maximize efficiency



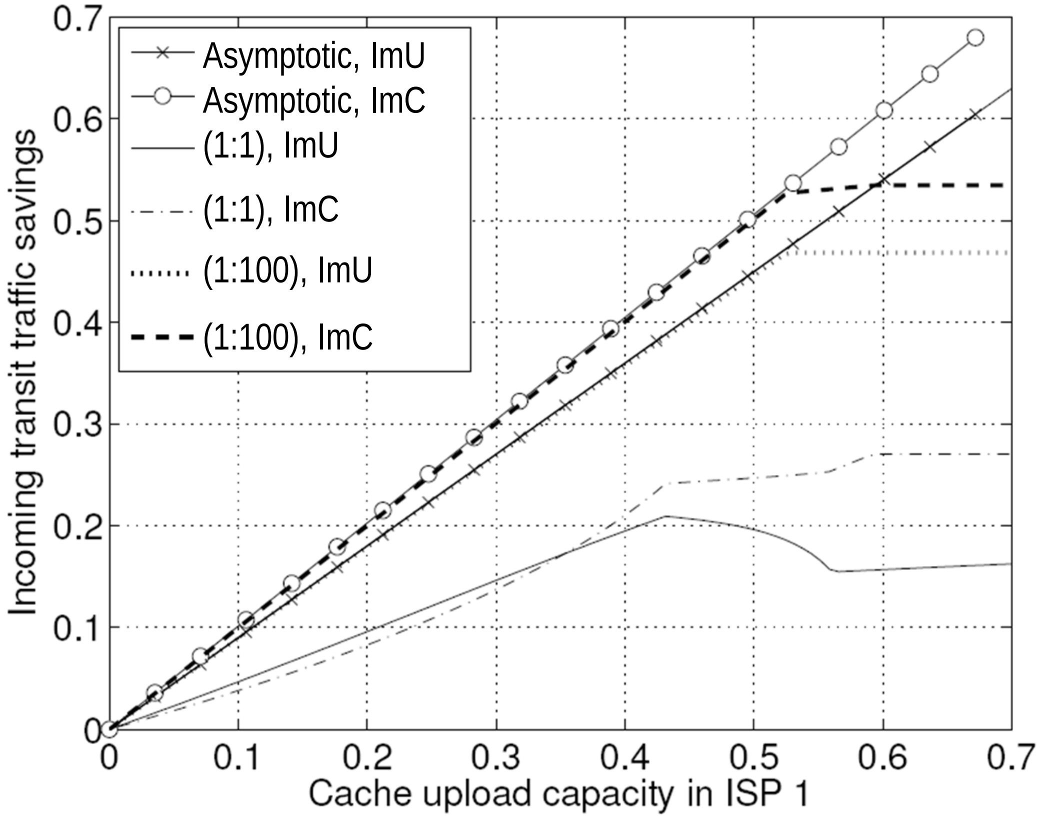
Conclusion

- Proposed a **fluid model for caches** in BitTorrent-like P2P networks to estimate **impact on transit traffic**
- Validation via simulations and experiments with real BitTorrent clients
- Insights
  - Caches effective when a large fraction of peers outside the ISP
  - Caches can lead to increased outgoing transit traffic
- Future work
  - Impact of proximity-aware peer selection
  - Management of cache upload rates in multi-swarm scenarios

BACKUP

Analytical Results: Incoming Transit Traffic

- Incoming transit traffic savings for ImU and ImC
- Ratio of peer arrivals (ISP 1:ISP 2): (1:1), (1:100), (1:∞) “asymptotic”
- Incoming transit traffic savings of ISP 1 larger for the (1:100)-scenario
- Cache ineffective when a large fraction of the peers is inside the ISP with the cache



Steady State Solutions and Insights

- All ISPs upload rate limited

Average number of leechers

$$\overline{x}_i = \frac{\lambda_i}{\nu\left(1+\frac{\theta}{\nu}\right)} - \frac{\kappa_i}{\mu\eta\left(1+\frac{\theta}{\nu}\right)} - \Delta_i(\mathbf{x},\mathbf{y},\kappa)$$

Average number of seeds

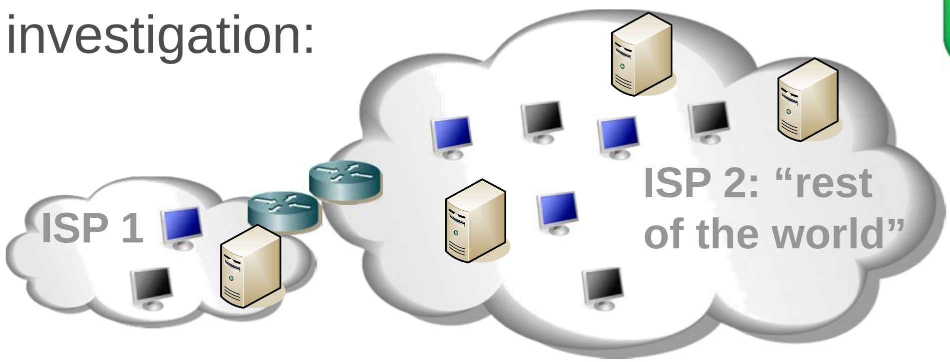
$$\overline{y}_i = \frac{\lambda_i}{\gamma\left(1+\frac{\theta}{\nu}\right)} + \frac{\kappa_i\theta}{\mu\eta\gamma\left(1+\frac{\theta}{\nu}\right)} + \frac{\theta}{\gamma}\Delta_i(\mathbf{x},\mathbf{y},\kappa),$$

A cache in ISP i decreases the average number of leechers in ISP i.

A cache increases the number of seeds in ISP i if θ>0, i.e., when peers abort the download

depends on
(1) aggregate cache capacities and
(2) aggregate arrival rates in other ISPs,
but not on their individual values!

- **Two ISP scenario sufficient** for investigation:



- All ISPs download rate limited: no impact on number of peers

Steady State Solutions for a Single System

- No distinction of ISPs, illustrates general impact of caches
- Upload rate limited case

Average number of leechers

$$\bar{x} = \frac{\lambda}{\nu\left(1+\frac{\theta}{\nu}\right)} - \frac{\kappa}{\mu\eta\left(1+\frac{\theta}{\nu}\right)}$$

Caches decrease the average number of leechers

Average number of seeds

$$\bar{y} = \frac{\lambda}{\gamma\left(1+\frac{\theta}{\nu}\right)} + \frac{\kappa\theta}{\mu\eta\gamma\left(1+\frac{\theta}{\nu}\right)}$$

Caches increase the number of seeds if $\theta>0$, i.e., when peers abort the download

- Download rate limited case:
  no impact of a cache on average number of peers

A Simple Model for Transit Traffic

- Model of **incoming and outgoing transit traffic** of the ISPs
  - Based on the number of leechers x_i and seeds y_i in the ISPs
  - Abstracts from inter-ISP delays, BitTorrent neighbor selection, and the choke algorithm
- Notation
  - Publicly available upload rate in ISP i:
    $u_i^P = \mu(\eta x_i + y_i)$
  - Demand rate in ISP i:
    $D_i^d = max(0, cx_i - \kappa_i)$ (for ImC, similar for ImU)
  - Received rate of peers in ISP i:
    $$D_i^r = D_i^d min\left(1, \frac{\sum_j u_j^P}{\sum_j D_j^d}\right)$$

Upload rate of peers in ISP i that can be used by leechers outside ISP i

Rate that the peers in ISP i demand from the total public upload rate

Rate at which peers in ISP i can receive data from the swarm

Transit Traffic Estimates

- Incoming transit traffic

$$\rho_i^I = D_i^r \left(1 - \frac{u_i^P}{\sum_j u_j^P} \right)$$

Assumption:
Incoming transit traffic the ISP proportional to the publicly available upload rate outside the ISP

- Outgoing transit traffic

$$\rho_i^O = \sum_{j \neq i} \rho_j^I \frac{u_i^P}{\sum_{k \neq j} u_k^P}$$

Assumption:
Transit traffic from ISP i to ISP j is proportional to the ratio of the publicly available upload rate in ISP i and the aggregate publicly available upload rate outside ISP j

A Simple Model for Transit Traffic

- Model of **incoming and outgoing transit traffic** of the ISPs
  - Based on the number of leechers x_i and seeds y_i in the ISPs
  - Abstracts from inter-ISP delays, BitTorrent neighbor selection, and the choke algorithm
- **Incoming transit traffic estimate**

Incoming transit traffic of ISP i = **Total transfer rate in swarm (caches not included)** • **Fraction of leechers which are in ISP i** • **Fraction of upload capacity of peers outside ISP i**

Assumption: incoming traffic of the ISP is proportional to the upload rate of the peers outside the ISP

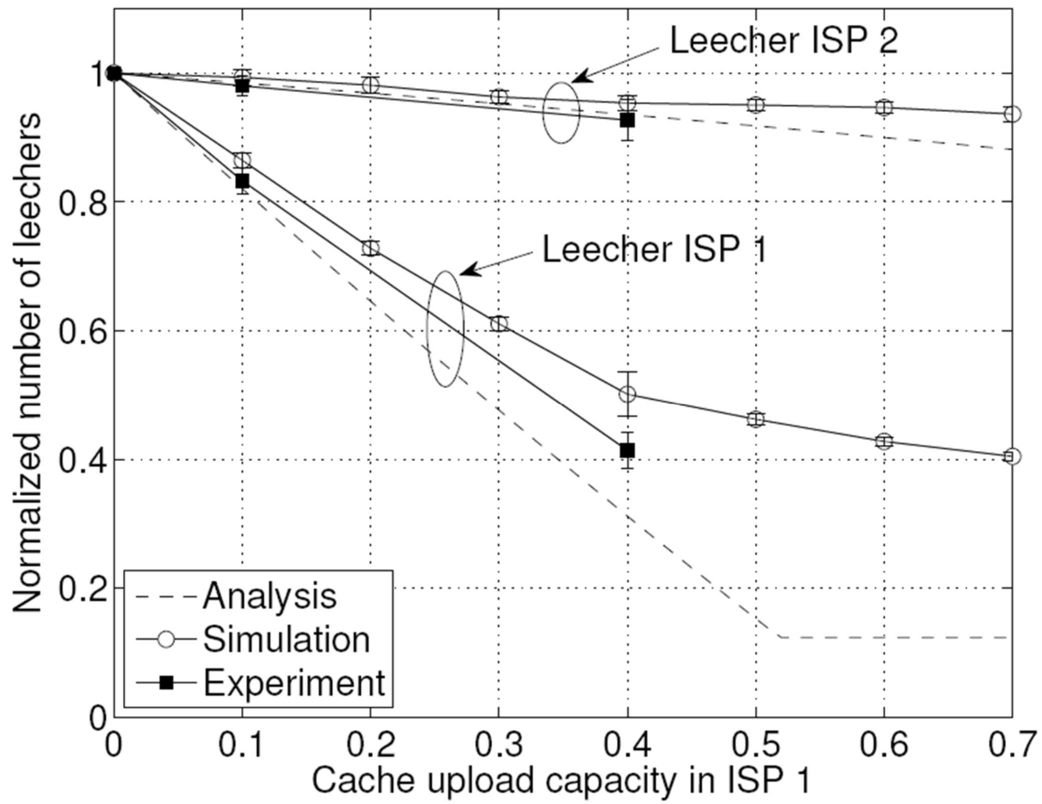
- **Outgoing transit traffic estimate**

Traffic from ISP i to ISP j = **Incoming transit traffic of ISP j** • **Ratio of upload capacity of peers in ISP i to upload capacity of peers outside ISP j**

Assumption: transit traffic from ISP i to ISP j is proportional to the ratio of the upload rate in ISP i and the aggregate upload rate outside ISP j

Validation of the Model: Number of Peers

- Normalized number: ratio of leechers with and without a cache
- Good match for ISP 2 and ISP 1 with small cache capacities
- Number of leecher in ISP 1 under-estimated for large cache capacities
- Reasons
  - Oscillations between an up- and download rate limited systems
  - Cache capacity not fully utilized



Taxonomy of Caches

- ISP-managed ultra-peers (ImU)
  - Run the BitTorrent protocol
  - **Appear as high capacity peers in the swarm**
  - Upload only to local leechers
  - Example: OverSi's OverCache P2P

- ISP-managed caches (ImC)
  - Similar to ImUs
  - Peers explicitly **prefer downloading from the cache**
  - Cache discovery protocols required (IETF ALTO & DECADE)

- Transparent caches
  - Intercept requests to external peers (DPI) and serve them
  - Example: PeerApp's UltraBand
  - Used for comparison, not part of our model