**Curso de doctorado:**

# Servicios en la Web y Distribucion de Conetnidos

## Un sistema peer-to-peer no estructurado: BitTorrent

*Mikel Izal Azcárate*

# BitTorrent

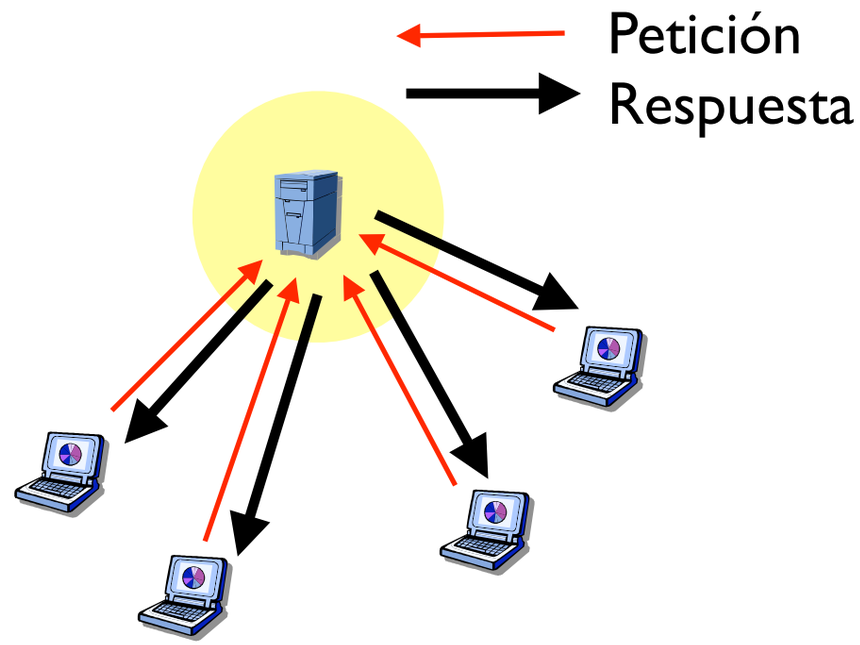
- Los sistemas peer-to-peer de intercambio de ficheros crecen en popularidad
- Uno de ellos ha cobrado notoriedad últimamente por su gran eficacia, aparte de por su enfoque en la distribución de contenidos legales. Es BitTorrent
- BitTorrent ha disparado el interés por los sistemas de distribución de contenidos peer-to-peer. Nuevas propuestas prometen mejores resultados [Slurpie] [Avalanche]
- Si el trafico peer-to-peer de compartición de archivos ya se está convirtiendo en la nueva killer application de Internet, estos nuevos sistemas de compartición altamente eficientes pueden jugar un papel muy importante en el tráfico futuro de Internet

# Contenido

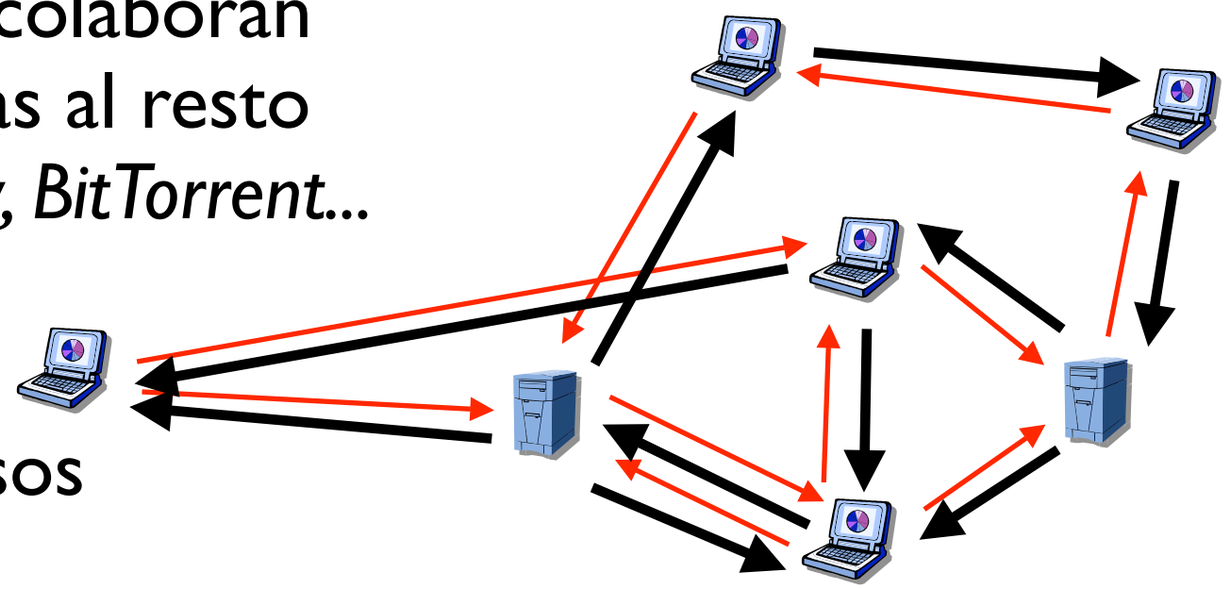
- **Introducción**: cómo funciona BitTorrent
- Estado del arte de la investigación sobre p2p de tipo BitTorrent
- Medidas realizadas
- Conclusiones y trabajos futuros

# *Introducción*: Paradigmas de servicios de Internet

- **Cliente-Servidor** (asimetría)
  cliente pide un servicio
  servidor responde
  Ejemplos: *web, ftp, mail...*
  un servidor - muchos clientes
  Mala escalabilidad




- **Peer-to-peer** (entre iguales)
  programas iguales (pares) colaboran
  cada uno puede pedir cosas al resto
  Ejemplos: *Gnutella, eDonkey, BitTorrent...*
  muchos peers distribuidos
  **Escalabilidad**
  mas clientes -> mas recursos



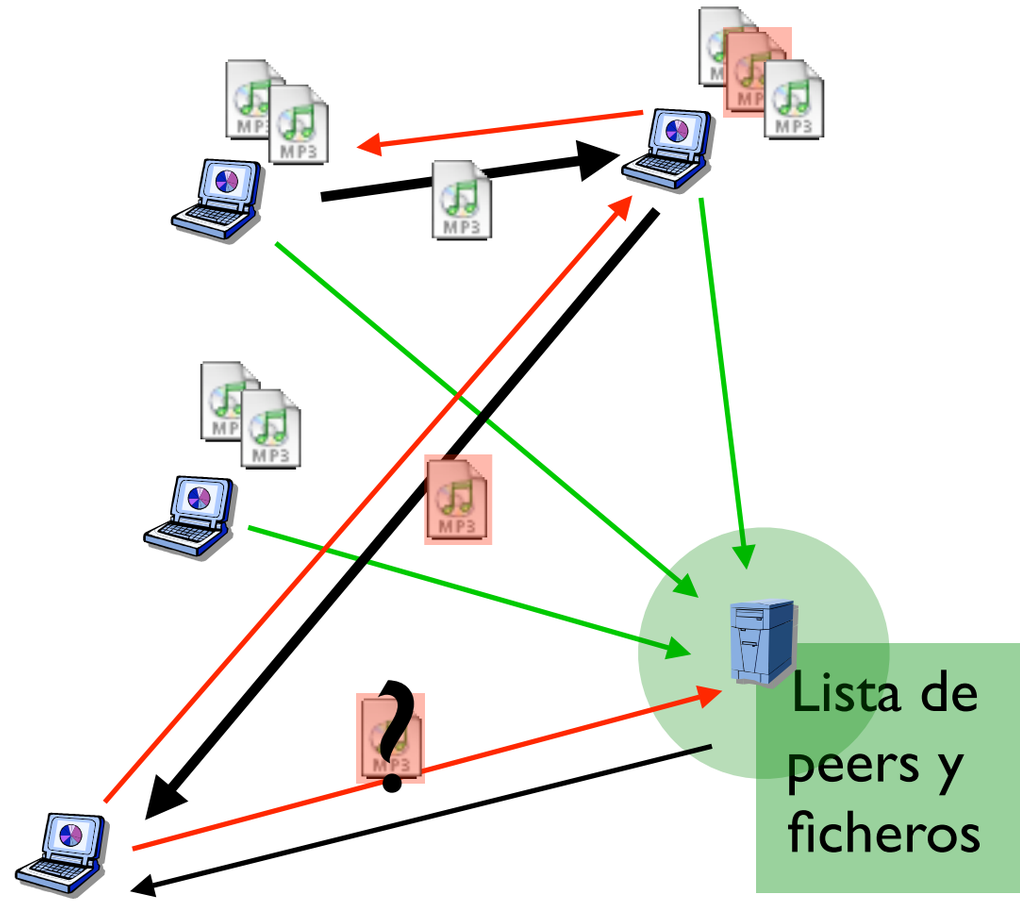
# *Introducción*: Evolución P2P

- Primeros sistemas. Sistema de **búsqueda centralizado** pero transferencia peer-to-peer: **Napster**
- Sistemas totalmente P2P: **Gnutella**. Búsqueda peer-to-peer por inundación. Transferencia peer-to-peer pero una sola fuente.
- Transferencia peer-to-peer de varias fuentes: **eDonkey/ eMule**. Sistema de reputación para premiar a los que colaboran. Descarga en general poco eficiente.
- Mejorando la eficiencia en la descarga. Sistemas de distribución peer-to-peer para distribución de contenidos populares y *flashcrowds*:

  **BitTorrent**. No integra sistema de busqueda de ficheros. Se centra en optimizar el transporte entre múltiples fuentes.

# *Introducción:* P2P y Napster

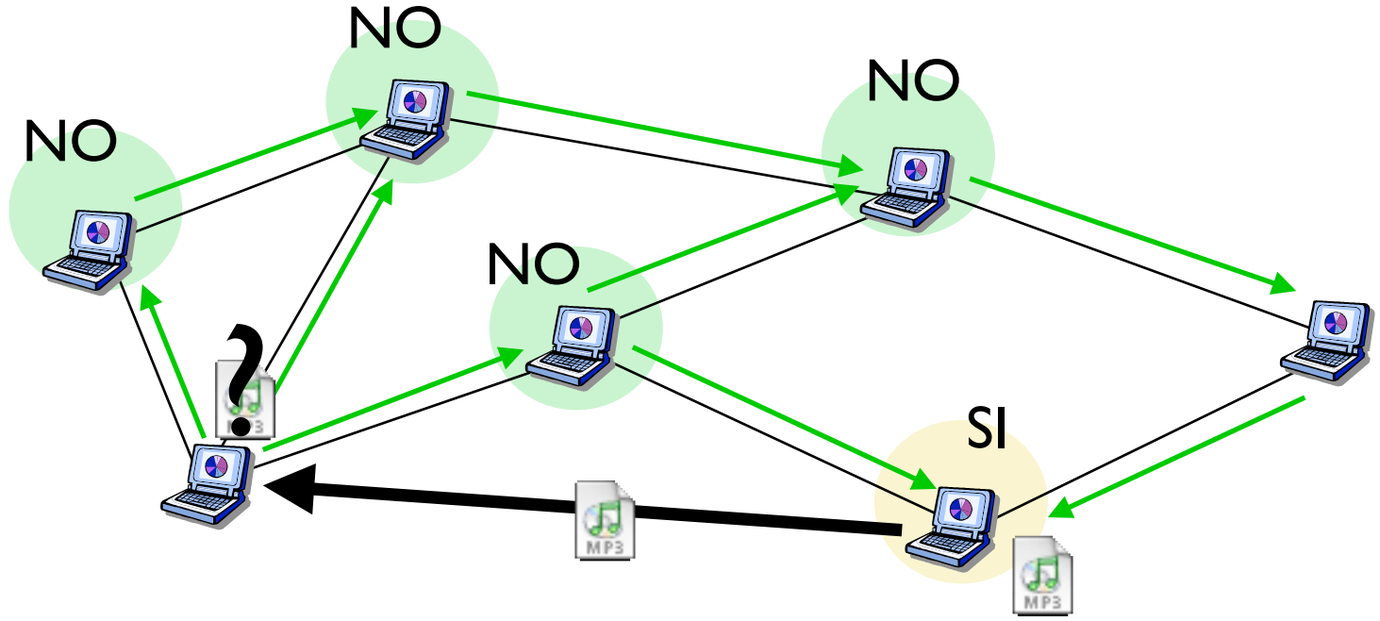
- Compartir ficheros (música, mp3)
- Hibrido P2P + Cliente/Servidor
  - Petición de ficheros P2P
    y a un solo peer
  - Busqueda cliente-servidor
    Servidor central de busqueda




- El servidor central acabó con Napster
  Punto de fallo (en este caso de demanda :-))

# *Introducción*: P2P y Gnutella

- Compartir todo tipo de ficheros
- Malla de peers sin tracker (*sólo conoces a tus vecinos*)
  Problema para entrar (*conoces a alguien dentro?*)
- Busqueda distribuida (peer-to-peer)
  Inundación (*alguien tiene este fichero?*)



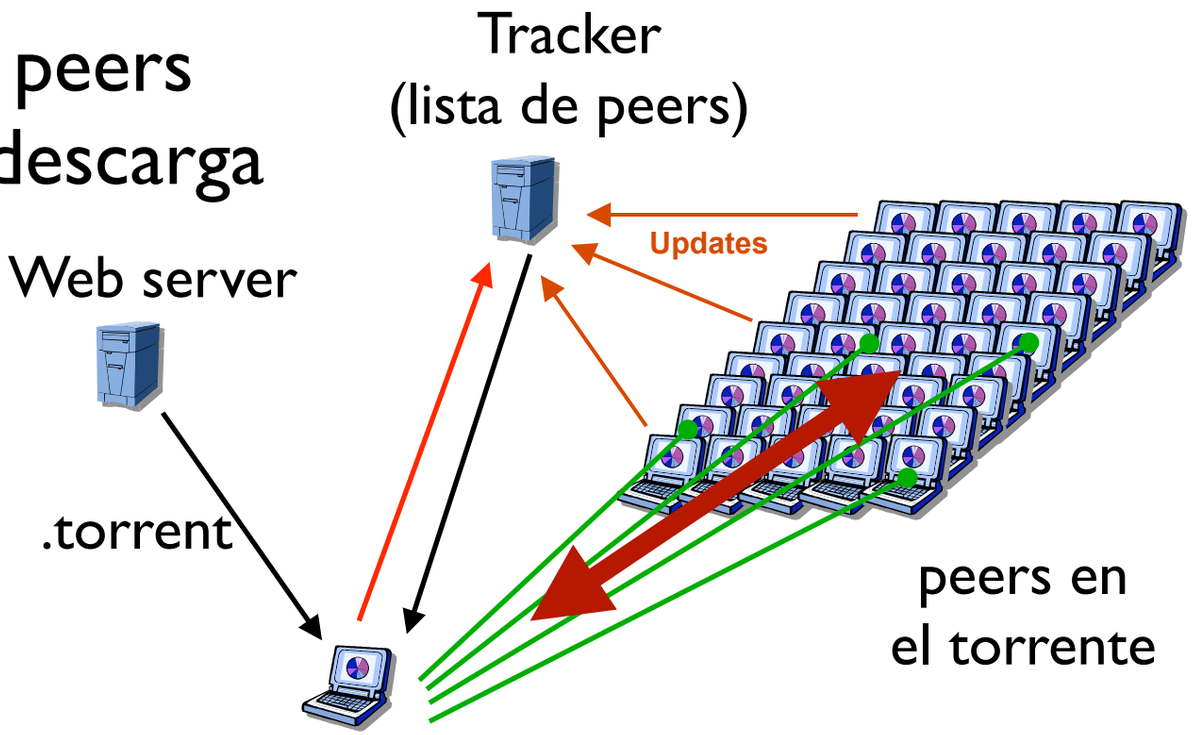

- Transporte P2P con un solo origen

# Contenido

- Introducción: **cómo funciona BitTorrent**
- Estado del arte de la investigación sobre p2p de tipo BitTorrent
- Medidas realizadas
- Conclusiones y trabajos futuros

# *Introducción:* BitTorrent

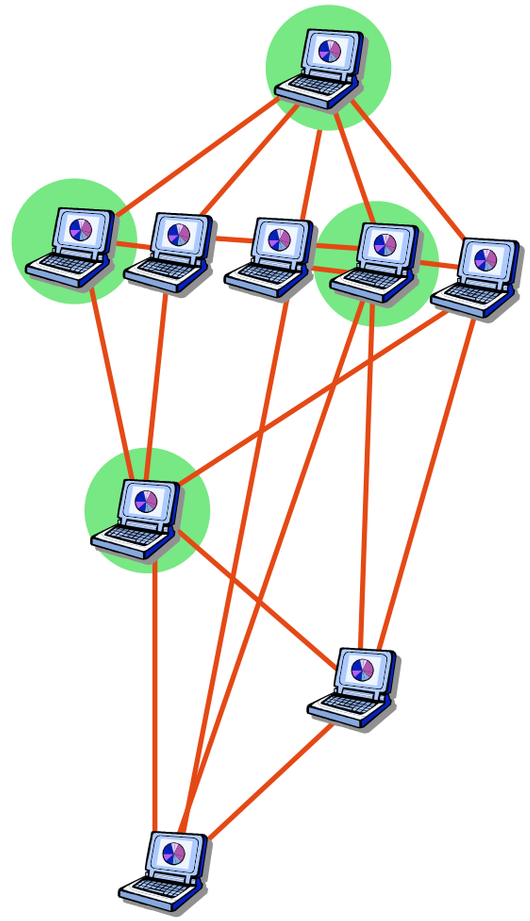
- **Replicación de un fichero**
  Malla de peers interesados en un fichero concreto: **Torrente**
- **Tracker** para encontrar otros peers
  Fichero *.torrent* describiendo la descarga
- Uniendose a un torrente
  - Bajar el .torrent
  - Lanzar peer de BT
  - Conseguir lista de peers
  - Conectarse a peers
  - Colaborar con los vecinos




- En un torrente
  **Seeds**: tienen el fichero **Leechers:** no tienen el fichero
  Un seed original. Leechers que acaban son seeds

# *Introducción:* BitTorrent: malla de peers

- Un peer se conecta (TCP) a N vecinos (*Por defecto 30*)
- Recibirá conexiones de peers
  *A menos que esté detrás de un Firewall o NAT*
- Los vecinos se convierten en seeds y/o se desconectan
- Si el numero de vecinos cae por debajo de un minimo se piden mas al tracker



# *Introducción:* BitTorrent: entre peers

- Los peers colaboran entre si (protocolo peer-to-peer) intercambiando partes del fichero

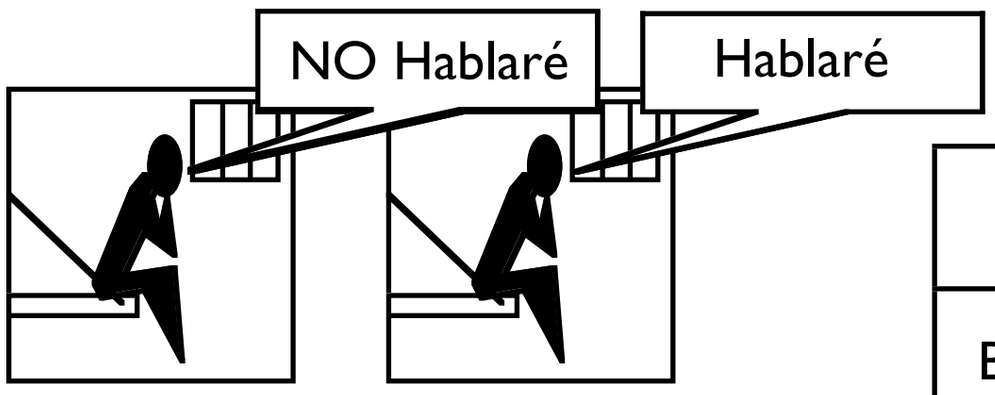
- El fichero se divide en bloques o piezas
  La lista de hashes de piezas están en el *.torrent*
  Un peer sabe si tiene una pieza verificando el hash

- Protocolo BitTorrent entre peers
  Por la conexión TCP se envían entre vecinos
  - Información del estado (bitmap de piezas y pieza nueva)
    **Siempre se lo que tienen mis vecinos**
  - Peticiones de piezas
  - Partes del fichero (en unidades mas pequeñas: subpiezas)
  - **Información de colaboración**

# *Introducción:* BitTorrent: colaboración(I)

- Los peers colaboran entre si
  Tarde o temprano alguien escribirá un peer que se aproveche
  **Se puede construir un protocolo que obligue a la colaboracion?**

- Teoría de juegos
  **El dilema del prisionero**



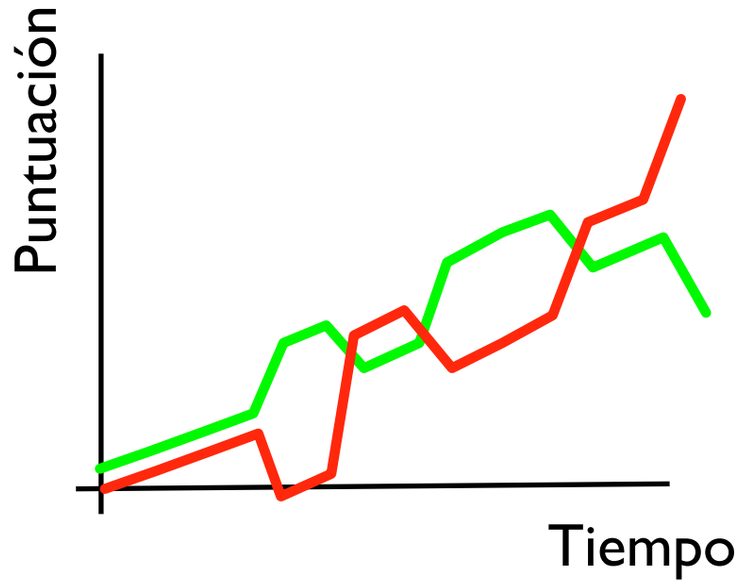

**Resultados**

| | A Colabora | A No colabora |
|---|---|---|
| B Colabora | A +1 puntos<br>B +1 puntos | A +3 puntos<br>B -2 puntos |
| B No colabora | A -2 puntos<br>B +3 puntos | A -1 puntos<br>B -1 puntos |

- Juego Iterativo: un dilema del prisionero cada turno
  Hay alguna estrategia para ganar???
  Que tiene que ver esto con la descarga de ficheros??

# *Introducción*: BitTorrent: colaboración(II)

- La mejor estrategia a largo plazo se llama **tit-for-tat**
  - Empiezo colaborando
  - Repito lo que ha hecho el otro en el turno anterior

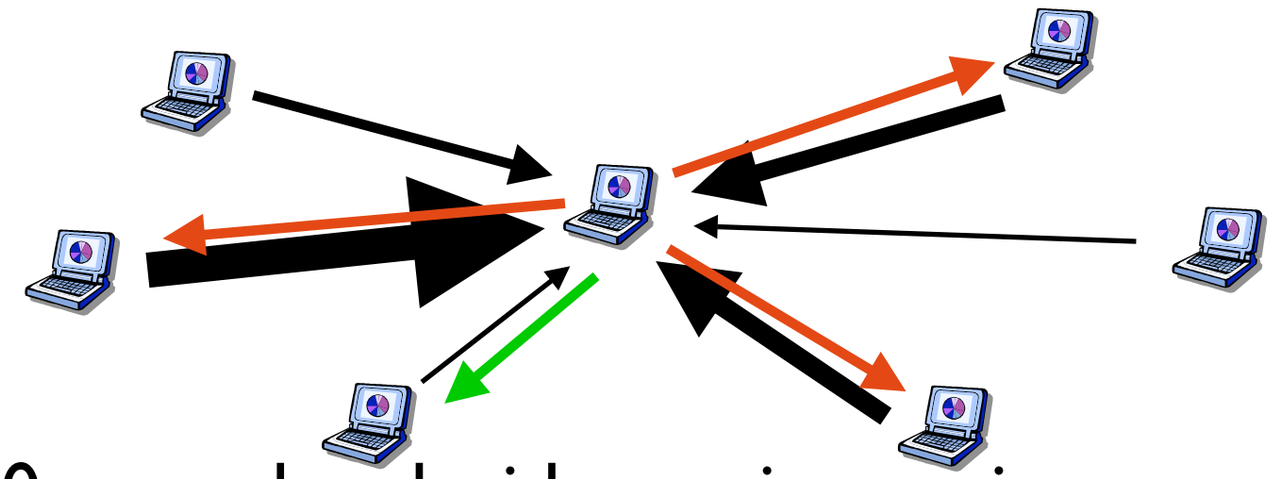



- 2 peers que se comunican siguen el mismo modelo
  Colaborar = enviar al otro peer lo que me pide
  Si no envío, todo el ancho de banda es para recibir
  Si los dos enviamos aun así recibo más que si los dos paramos

- BitTorrent se basa en este principio
  Por defecto envía lo que le piden
  Dejar de enviar: CHOKE
  Volver a enviar: UNCHOKE

# *Introducción*: BitTorrent: colaboración(III)

- Un peer de BitTorrent
  Envía solo a 4 vecinos (UNCHOKED) y no envia (CHOKE) al resto



- Cada 10 segundos decide a quien enviar
  Elige a los 4 que más han enviado (Throughput 30seg)
- Elige ademas a uno al azar para **Optimistic UNCHOKE**
- Algoritmo **tit-for-tat** (teoría de juegos)
- Los seeds eligen con el throughput de subida.
  Intentan maximizar su upload

# *Introducción:* BitTorrent: últimos detalles

- Para que el tit-for-tat funcione
  En general cada peer siempre mantiene peticiones pendientes a cada vecino

- Conozco todas las piezas que tiene mi vecino
  Qué pieza debo pedir?
  - Aproximación I: Pieza aleatoria
  - Aproximación II : Buscar la **pieza mas rara** para tener piezas que no tengan los demás
    **Pido la pieza que menos tengan mis vecinos**
    Esto no quiere decir que pida solo una pieza a la vez !!

- Básicamente esto es BitTorrent

# Contenido

- Introducción: cómo funciona BitTorrent
- **Estado del arte de la investigación sobre p2p de tipo BitTorrent**
- Medidas realizadas
- Conclusiones y trabajos futuros

# *Literatura:* Sobre BitTorrent


- [Cohen] B. Cohen. **Incentives to build robustness in bittorrent**, http://bitconjurer.org/bittorrent/bittorrentecon.pdf, Mayo 2003.
- [Izal] M. Izal, G. Urvoy-Keller, E. Biersack, P. Felber, A. A. Hamra, and L. Garcés-Erice. **Disecting bittorrent: Five months in a torrent's lifetime**. In Proceedings of the Passive and Active Network Measurement 5th International Workshop, Antibes Juan-les-Pins, France, April 2004.
- [Qiu] D. Qiu and R. Srikant. **Modeling and performance analysis of bittorrent-like peer-to-peer networks**. In *Proceedings of SIGCOMM'04*, Portland, Oregon, 2004.
- [Gkantsidis] C. Gkantsidis and P. Rodriguez. **Network coding for large scale content distribution**. In *IEEE/INFOCOM 2005*, Miami, Marzo 2005.

# *Literatura:* Sobre BitTorrent

[Cohen]

- Propuesta de BitTorrent, centrada sobre todo en sistema robusto y resistente a la no colaboración

[Izal]

- Medidas de un torrente real (RedHat9)

[Qiu]

- Modelado Markov de un torrente

[Gkantsidis]

- Propuesta de protocolo de tipo BitTorrent (Avalanche) que usa network coding para evitar la no utilización de enlaces debido a peers que no tienen nada que interese a sus vecinos

# Contenido

- Introducción: cómo funciona BitTorrent
- Estado del arte de la investigación sobre p2p de tipo BitTorrent
- **Medidas realizadas**
- Conclusiones y trabajos futuros

# Midiendo BT: observando un torrente

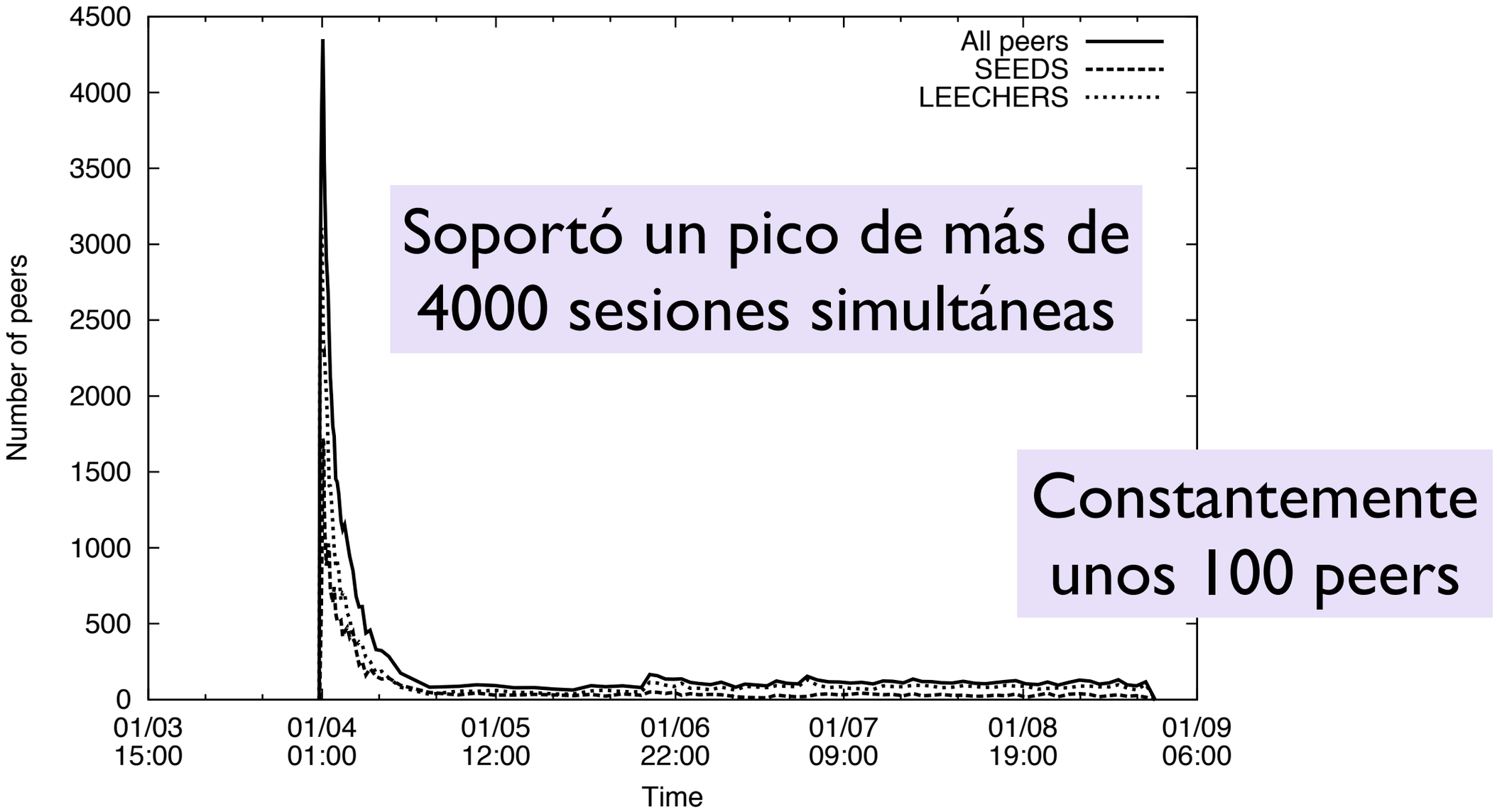
- Fuente: traza del tracker en f.scarywater.net del torrente de la distribución de Linux RedHat 9
- Traza conseguida en Ago2003 de 4 meses (aprox Abril-Julio 03)
- Es un apache-log-like file:

```
130.233.220.23 - - [31/Mar/2003:12:52:30] "GET /announce?info_hash=E%E9...%18%1D
        & peer_id=%00...%8E/ & port=6882 & ip=130.233.20.169
        & uploaded=0 & downloaded=0 & left=185509495l
        & event=started HTTP/1.0" 200 94
217.160.111.64 - - [31/Mar/2003:12:52:48] "GET /announce?info_hash=E%e9...%1d ...
80.198.193.222 gzip - [31/Mar/2003:12:53:17] "GET /announce?info_hash=E%E9..%1D ...
80.198.193.222 gzip - [31/Mar/2003:12:53:17] "GET /announce?info_hash=E%E9...%1D ...
```

- **Uploaded / downloaded / left** bytes

# Midiendo BitTorrent: observando un torrente

>> El número de peers en un momento dado
obtenido de la primera y ultima vez que se ve un **session_id**

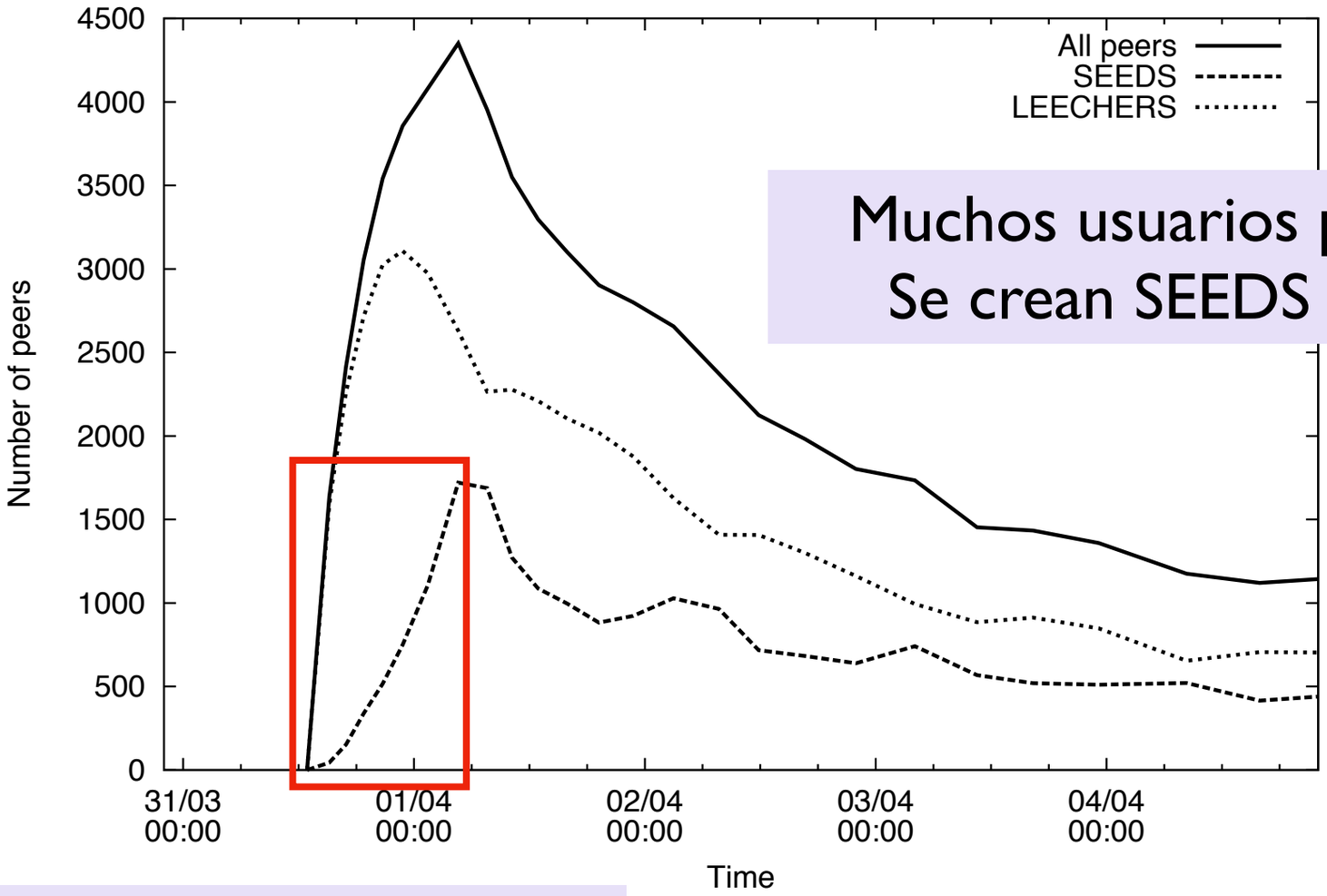



Soportó un pico de más de 4000 sesiones simultáneas

Constantemente unos 100 peers

# Midiendo BitTorrent: observando un torrente

>> Composicion de Seeds / Leechers en los primeros dias



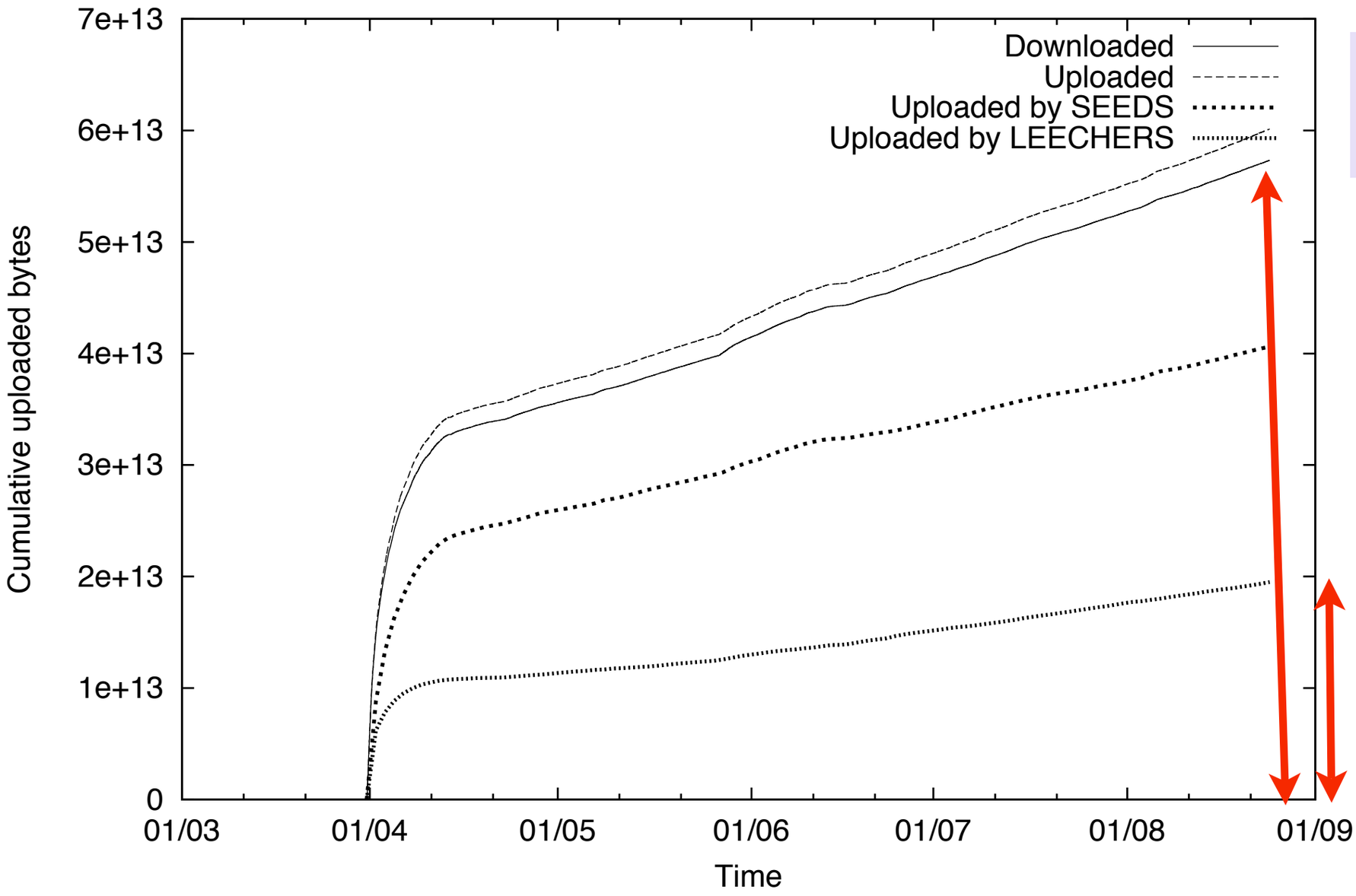

Muchos usuarios pidiendo el fichero.
Se crean SEEDS muy rápidamente

En medio día tenemos más de 1500 SEEDS !!!

Parece que BitTorrent aguanta muy bien los *flash-crowds* o *slashdot-effect*

# Midiendo BitTorrent: observando un torrente

>> Volumen total manejado por el torrente

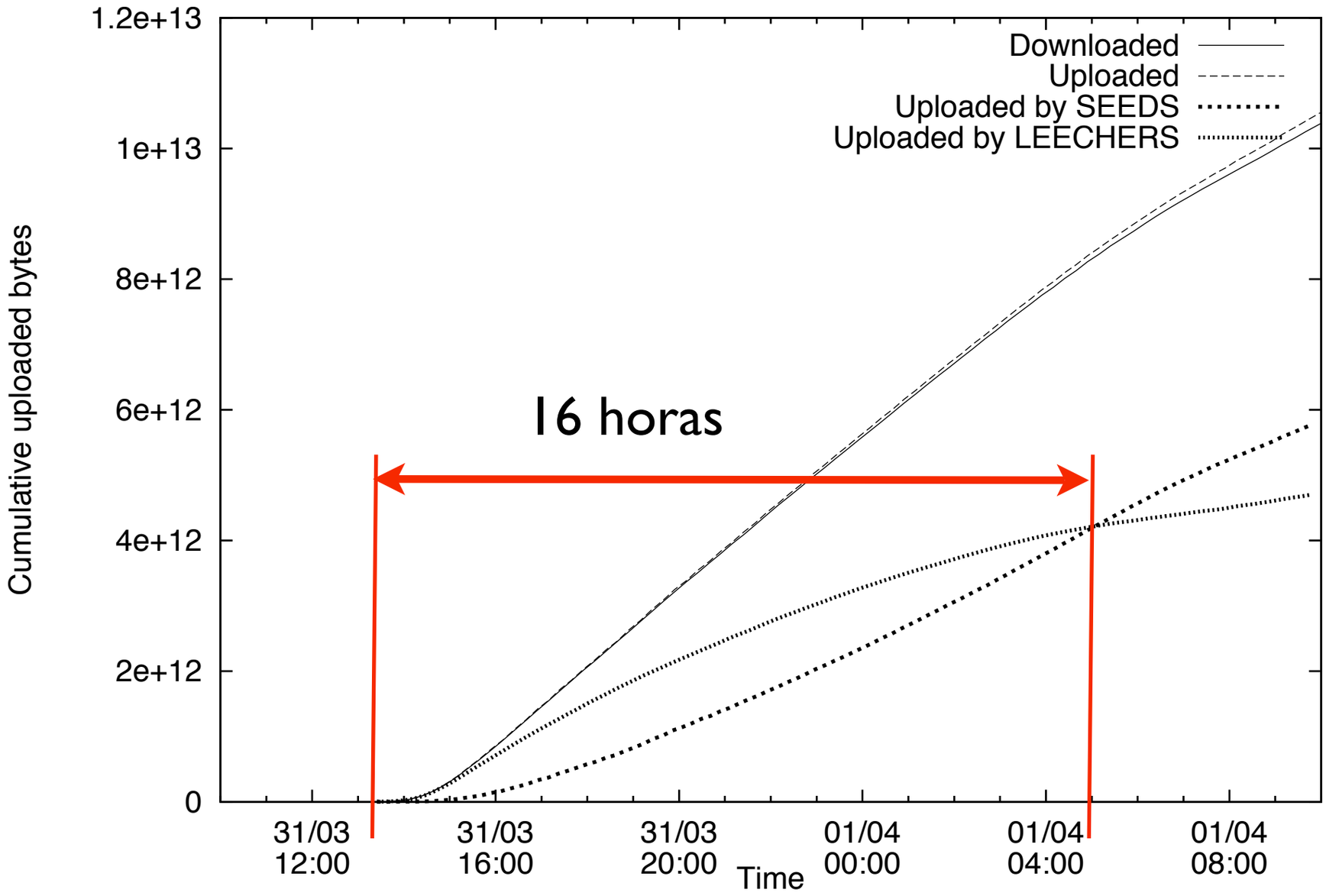



upload y download no cuadran

1/3 del upload de Leechers

# Midiendo BitTorrent: observando un torrente

>> En los primeros momentos del torrente los Leechers contribuyen más que los seeds



# Midiendo BitTorrent: observando un torrente

- **Problemas**
  - Resolución temporal de 15 minutos
  - Solo estado global del peer, no se puede reconstruir la malla de peers
  - Es dificil de repetir la medida
- **Ventajas / aspectos interesantes**
  - Se pueden ver proxies/Nat
  - Estadisticas de todo el torrente
  - Estadisticas de 4000 usuarios interesados en el mismo fichero

# Midiendo BitTorrent: siguiendo a un peer

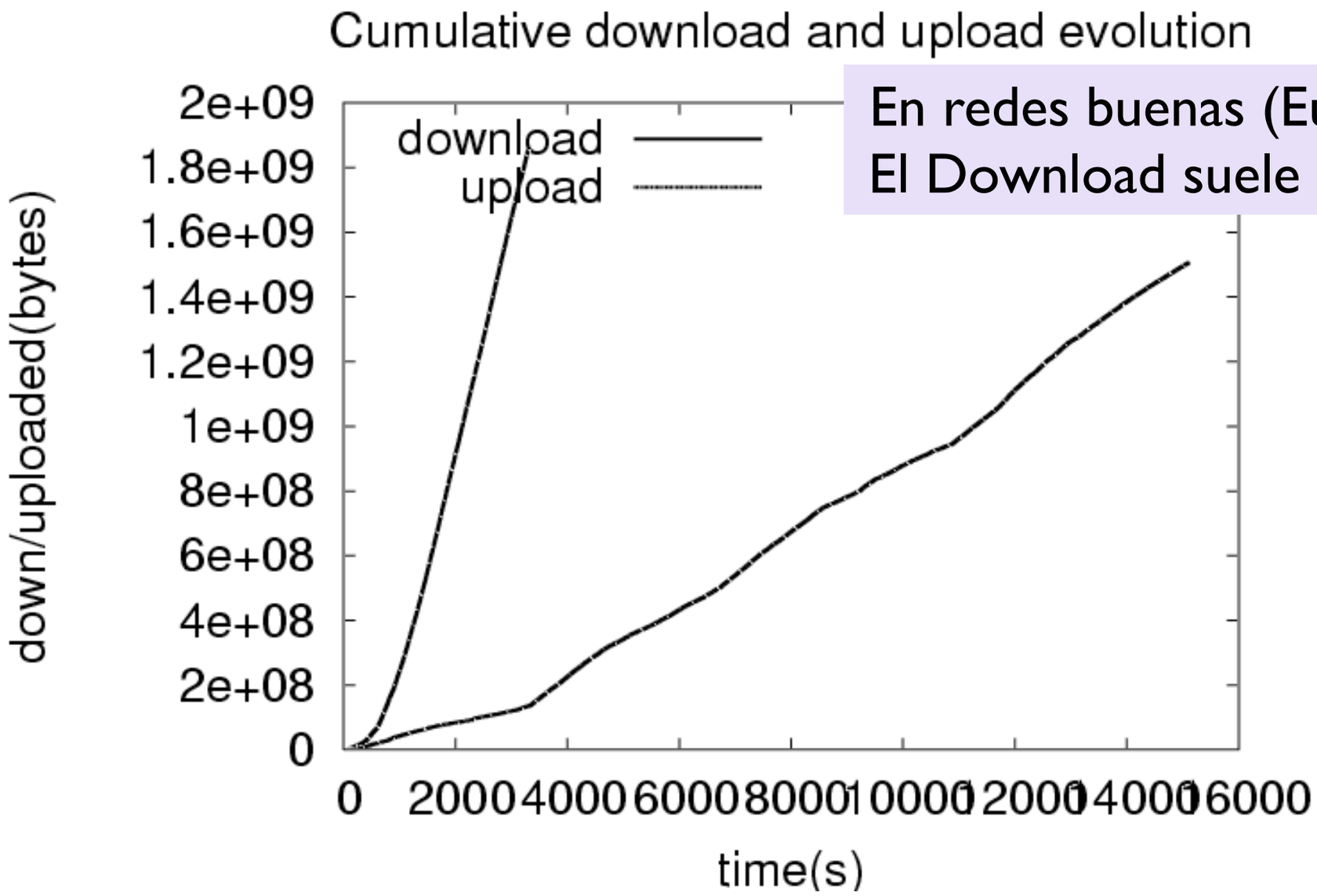
- Este estudio empezó analizando las conexiones TCP de BitTorrent a base de tcpdump y TCPRateAnalyzer

  En vista de que estabámos llegando a hacer ingeniería inversa del protocolo decidimos cambiar de estrategia:

- Análisis del código (python) de BitTorrent
- Estudio de la dinámica de un peer de BitTorrent
  - **BitTorrent modificado**
    - Log a fichero de variables del peer
    - Log a fichero (por cada vecino) de eventos
  - **Estudiar estas trazas de *nivel de aplicación***

# Midiendo BitTorrent: siguiendo a un peer

>> Bytes downloaded y uploaded (Traza del 14 de Julio) sólo por mi peer

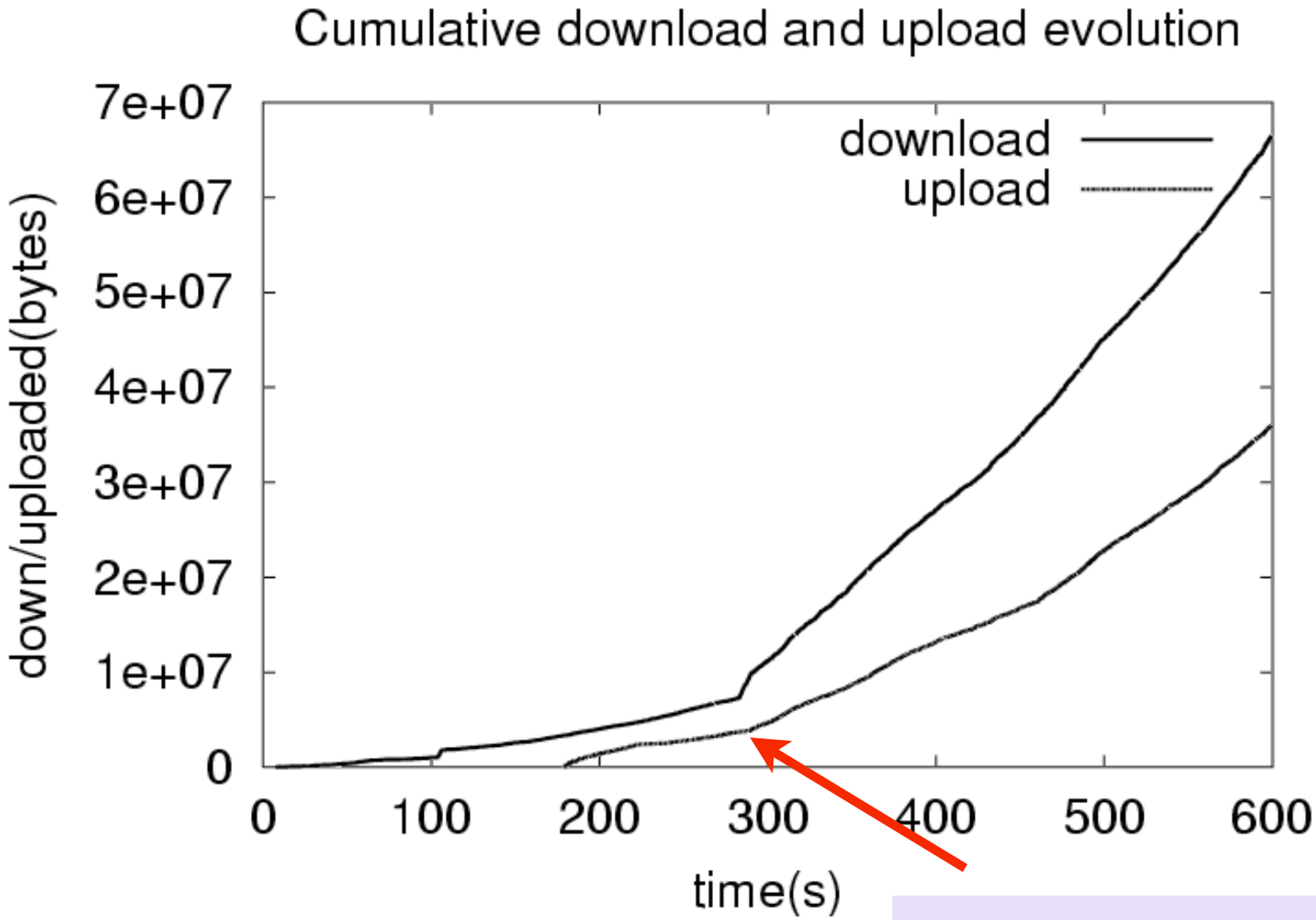



En redes buenas (Eurecom, UPNA)
El Download suele salir más rapido

# Midiendo BitTorrent: siguiendo a un peer

>> El comienzo de la descarga se podría mejorar

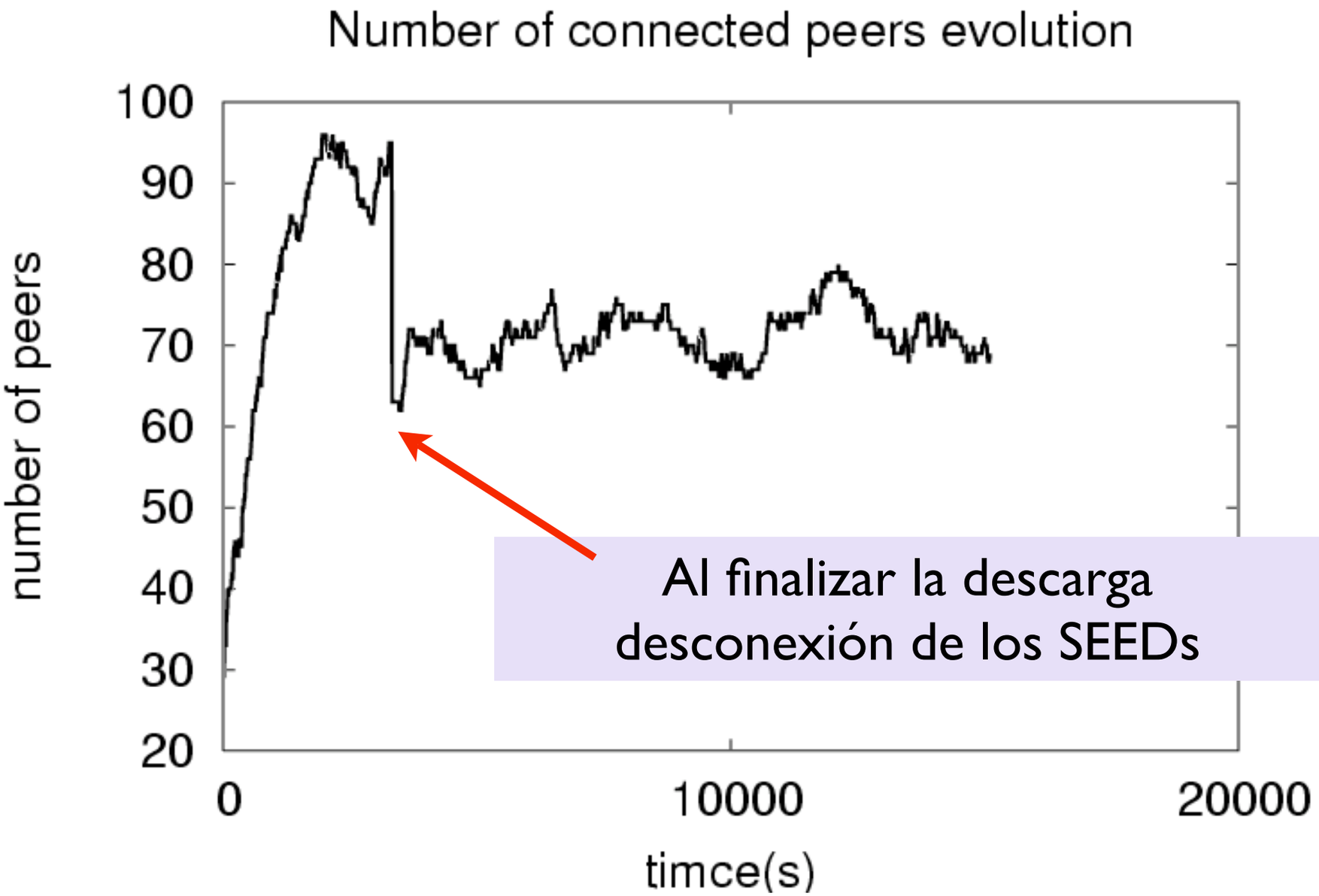
Puede tardar unos minutos en conseguir piezas para negociar




Hasta aqui recibimos por *optimistic unchoke ?*

# Midiendo BitTorrent: siguiendo a un peer

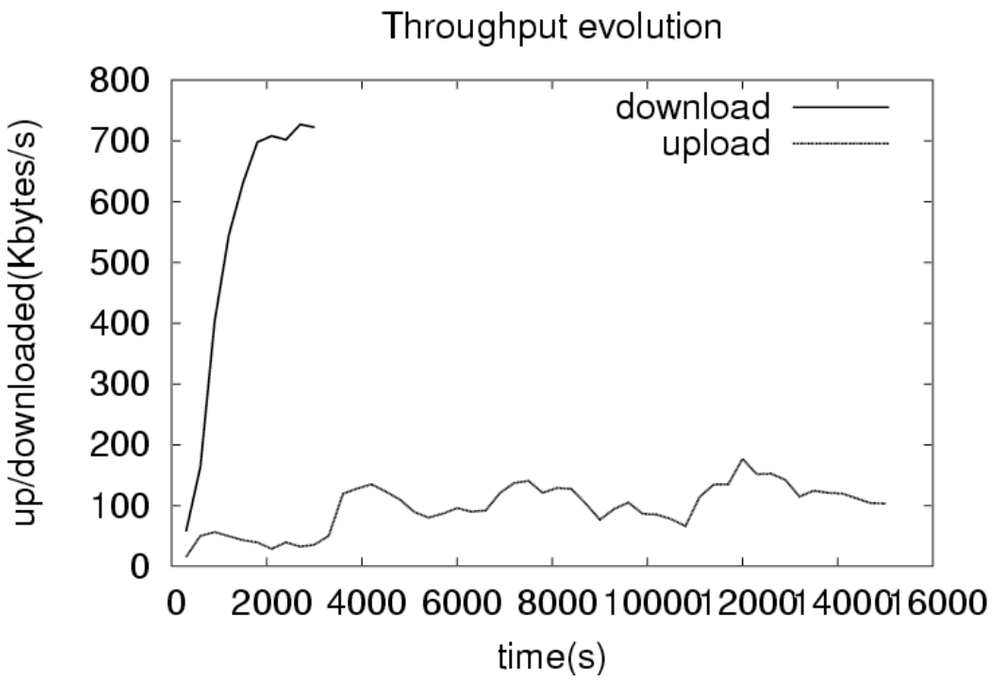
>> Vecinos conectados a mi peer



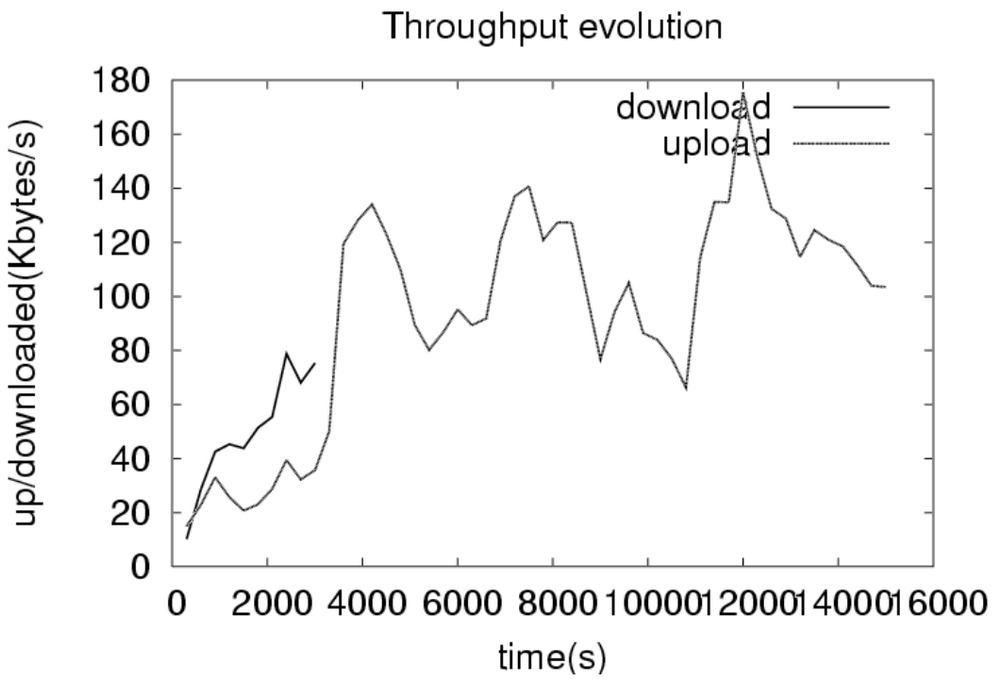
# Midiendo BitTorrent: siguiendo a un peer

- Parece que recibimos más de lo que enviamos
  Es normal porque los seeds solo envían
- Incluso sin contar lo que recibimos de los seeds recibimos más
  **Es normal o es que tenemos ventaja?**

Throughput Up y Down




Throughput Up y Down
Sin contar SEEDS



# Trazas en la UPNA

- Campaña de recogida de trazas de peers de BitTorrent en la UPNA
  - Trazas de peers
  - Trazas de SEEDs
    - En torrentes muy populares y en torrentes con pocos peers
    - En el enlace de la UPNA y en enlace ADSL

Algunas trazas
D < U

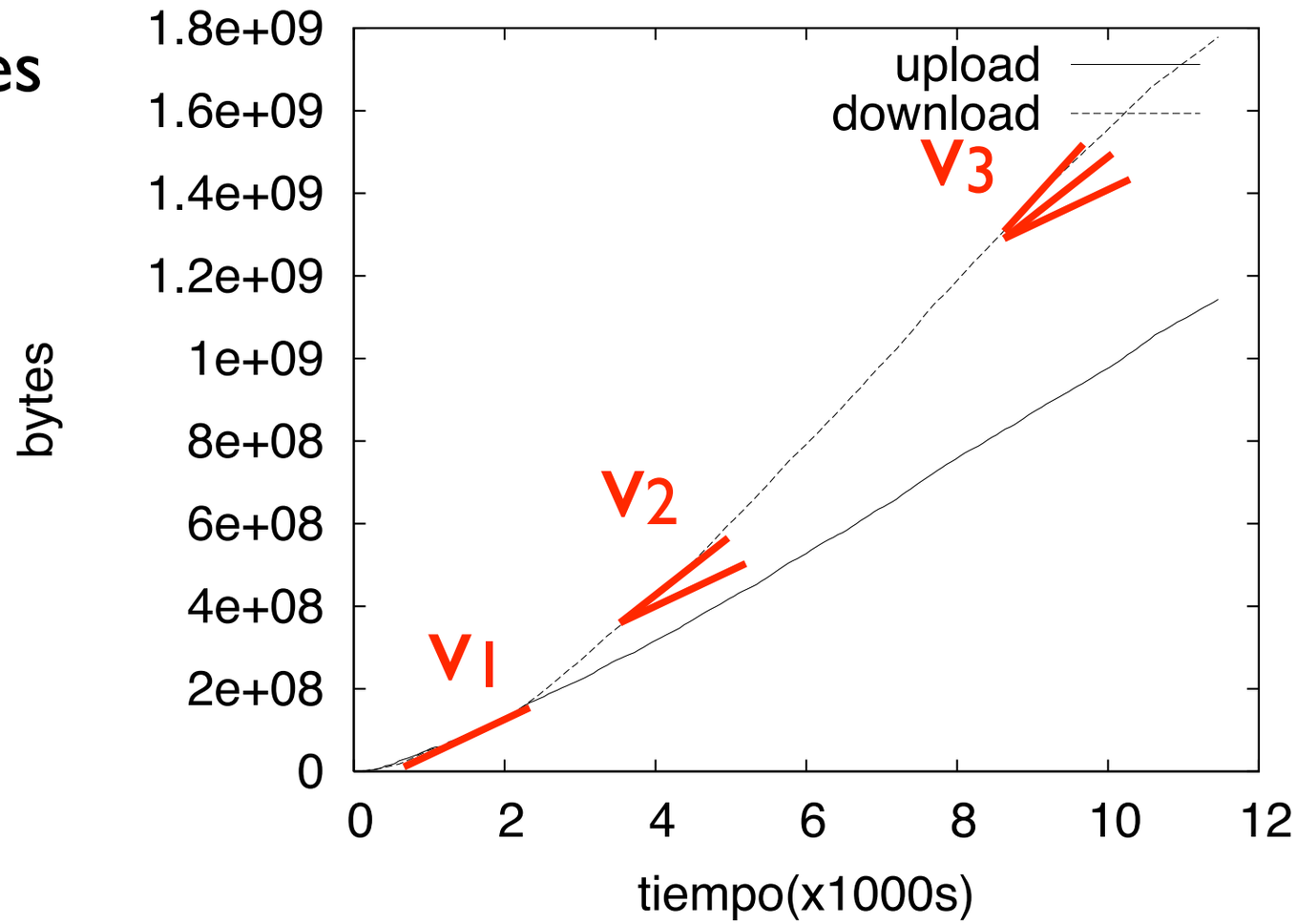
Mayor upload como seed

| Traza | Download ( kB/s ) | Upload como leech ( kB/s ) | Upload como seed ( kB/s ) |
|---|---|---|---|
| *Trz-1* | 155.2 | 99.8 | 0.0 |
| *Trz-2-fedora* | 292.3 | 29.5 | 32.5 |
| *Trz-3* | 71.5 | 53.7 | 62.5 |
| *Trz-4* | 46.9 | 105.7 | 144.4 |
| *Trz-seeding-fail1* | 0.0 | 0.0 | 4.5 |
| *Trz-seeding-fail2* | 0.0 | 0.0 | 1.0 |
| *Trz-seeding-fail3* | 0.0 | 0.0 | 1.2 |
| *Trz-seeding-fail4* | 0.0 | 0.0 | 173.8 |
| *Trz-seeding* | 0.0 | 0.0 | 524.8 |
| *Trz-5-casa* | 29.1 | 10.7 | 11.3 |
| *Trz-5-upna* | 126.1 | 379.4 | 722.0 |
| *Trz-6-knoppix* | 121.3 | 451.4 | 774.2 |
| *Trz-7a* | 1.9 | 19.7 | 18.8 |
| *Trz-7b* | 134.0 | 426.3 | 606.9 |

Mismo torrente en la UPNA
y en un enlace ADSL

# Comportamiento de un peer

- Forma de la descarga se repite
- En general la velocidad es mayor a más tiempo de estancia en el torrente



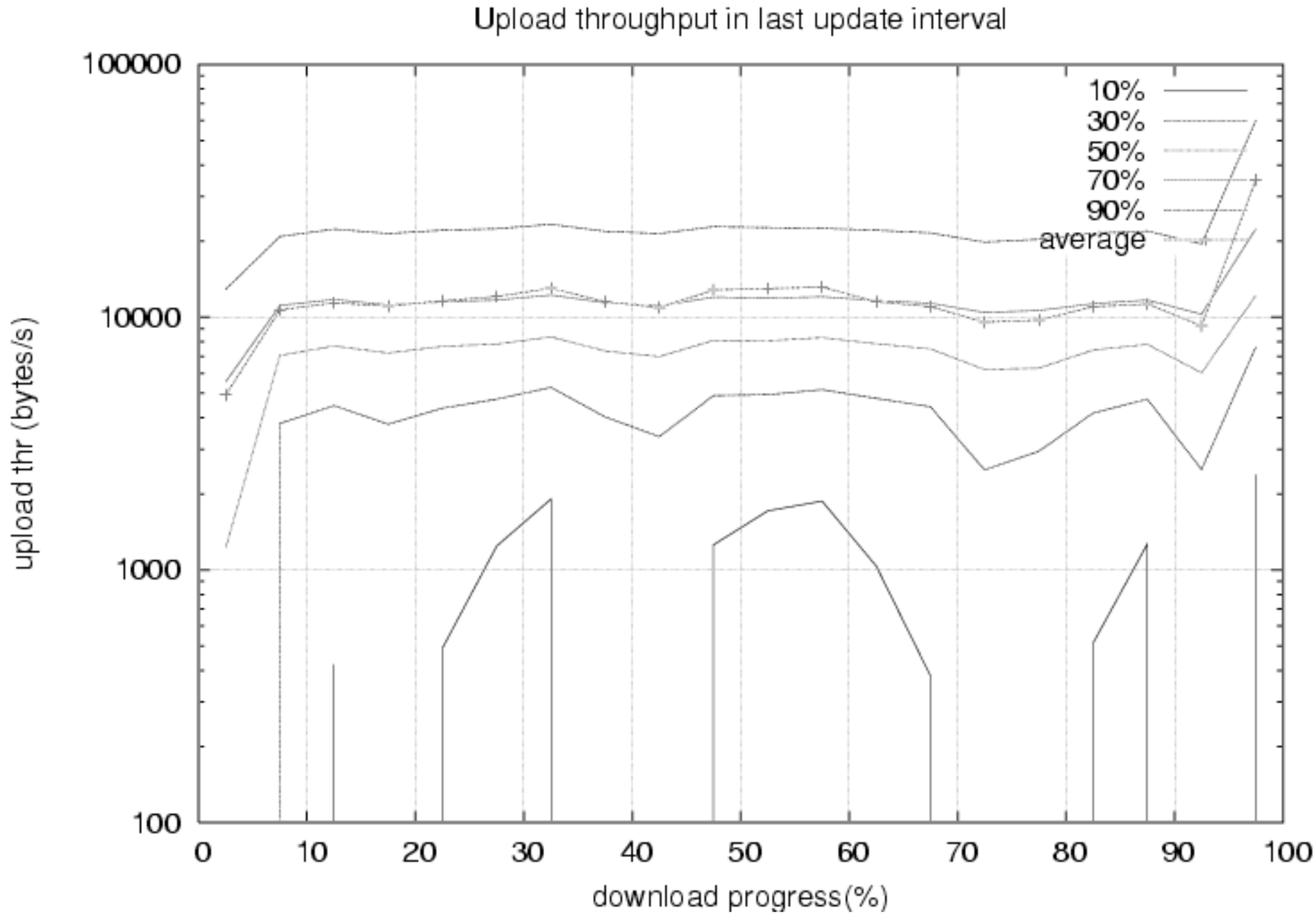

- ¿Por qué los peers más veteranos reciben más rápido?
- ¿Tiene que ver con que tengan más porcentaje del fichero?

# Midiendo BitTorrent: de donde viene la ventaja?

- Midiendo la velocidad de subida/bajada según el estado de descarga
  - Throughput en los últimos 5 minutos
  - Clasificado según el porcentaje del fichero que tenemos

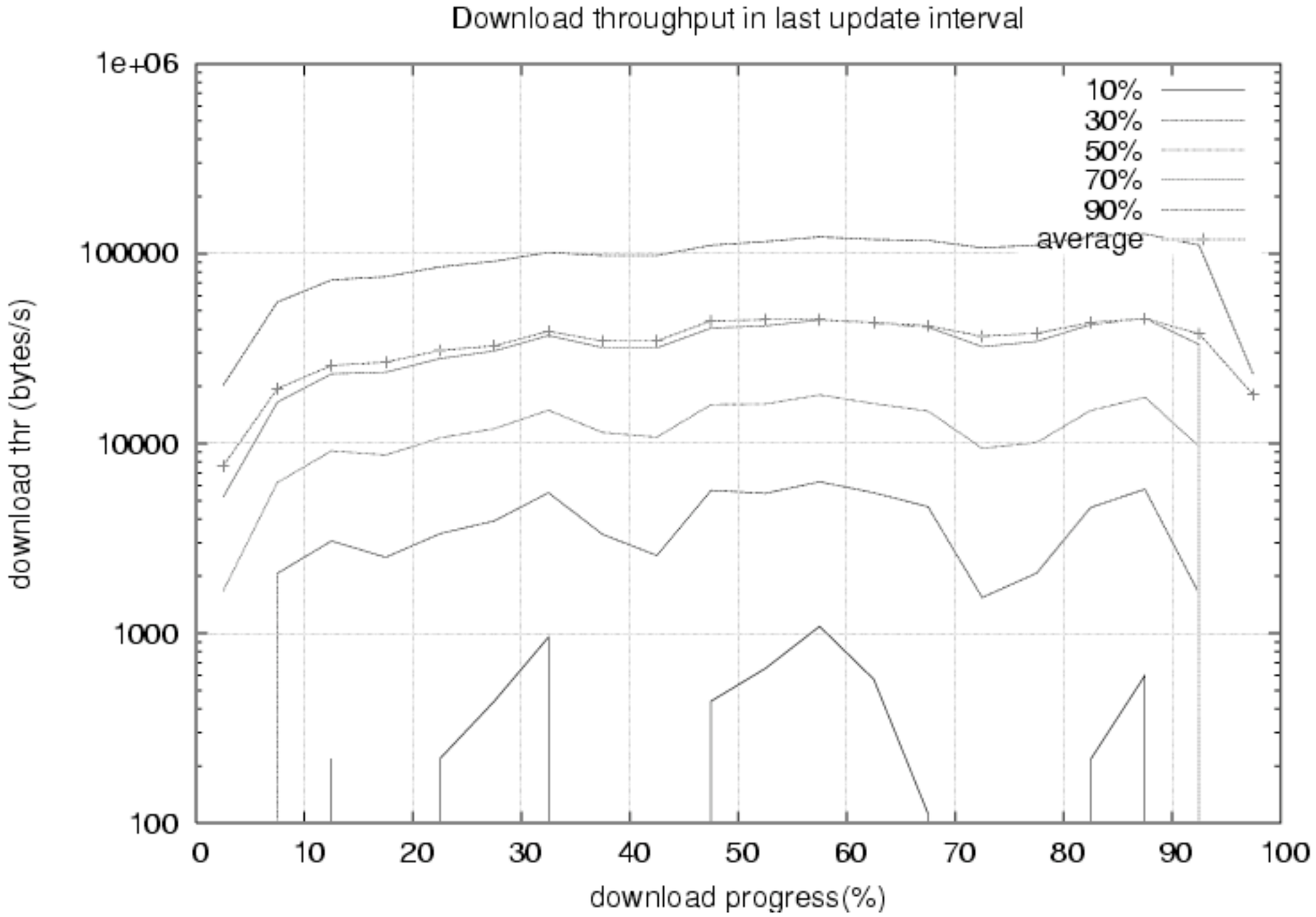
# Midiendo BitTorrent: de donde viene la ventaja?

>> Throughput enviado desde un peer



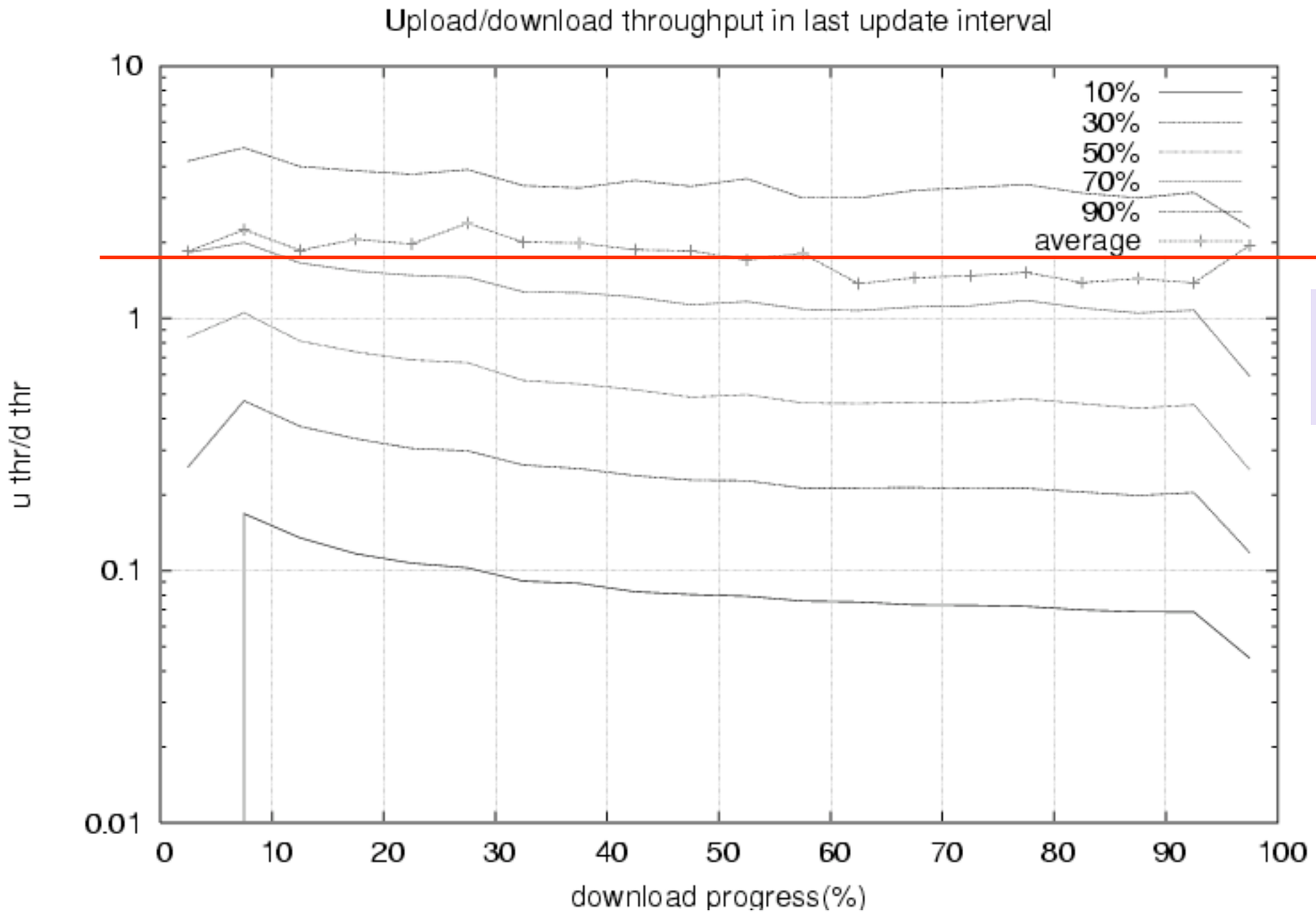
# Midiendo BitTorrent: de donde viene la ventaja?

>> Throughput recibido por un peer



# Midiendo BitTorrent: de donde viene la ventaja?

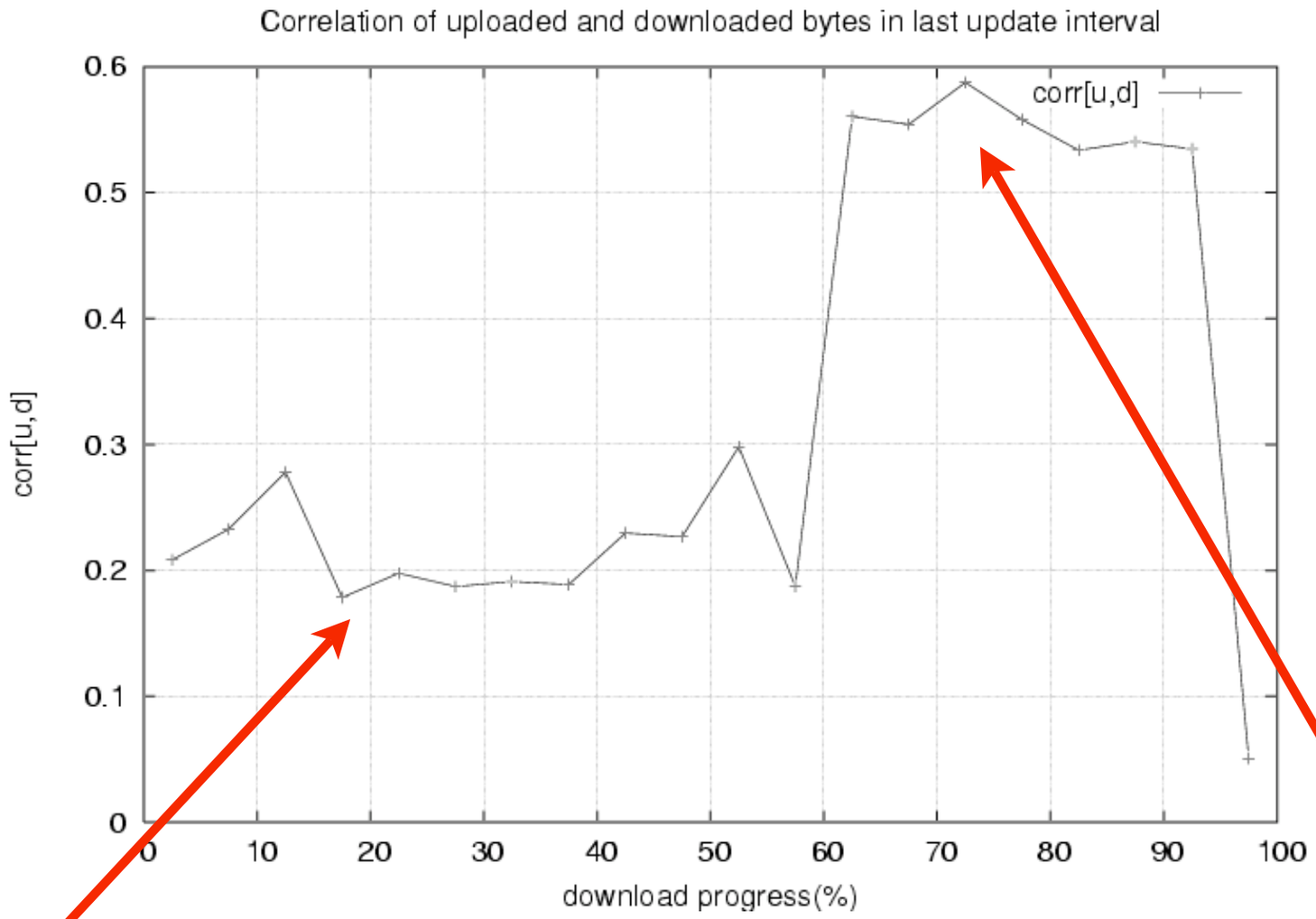
>> Throughput enviado/recibido por un peer




Los que más tienen reciben más

# Midiendo BitTorrent: de donde viene la ventaja?

>> Correlación (enviado, recibido) según el estado de descarga

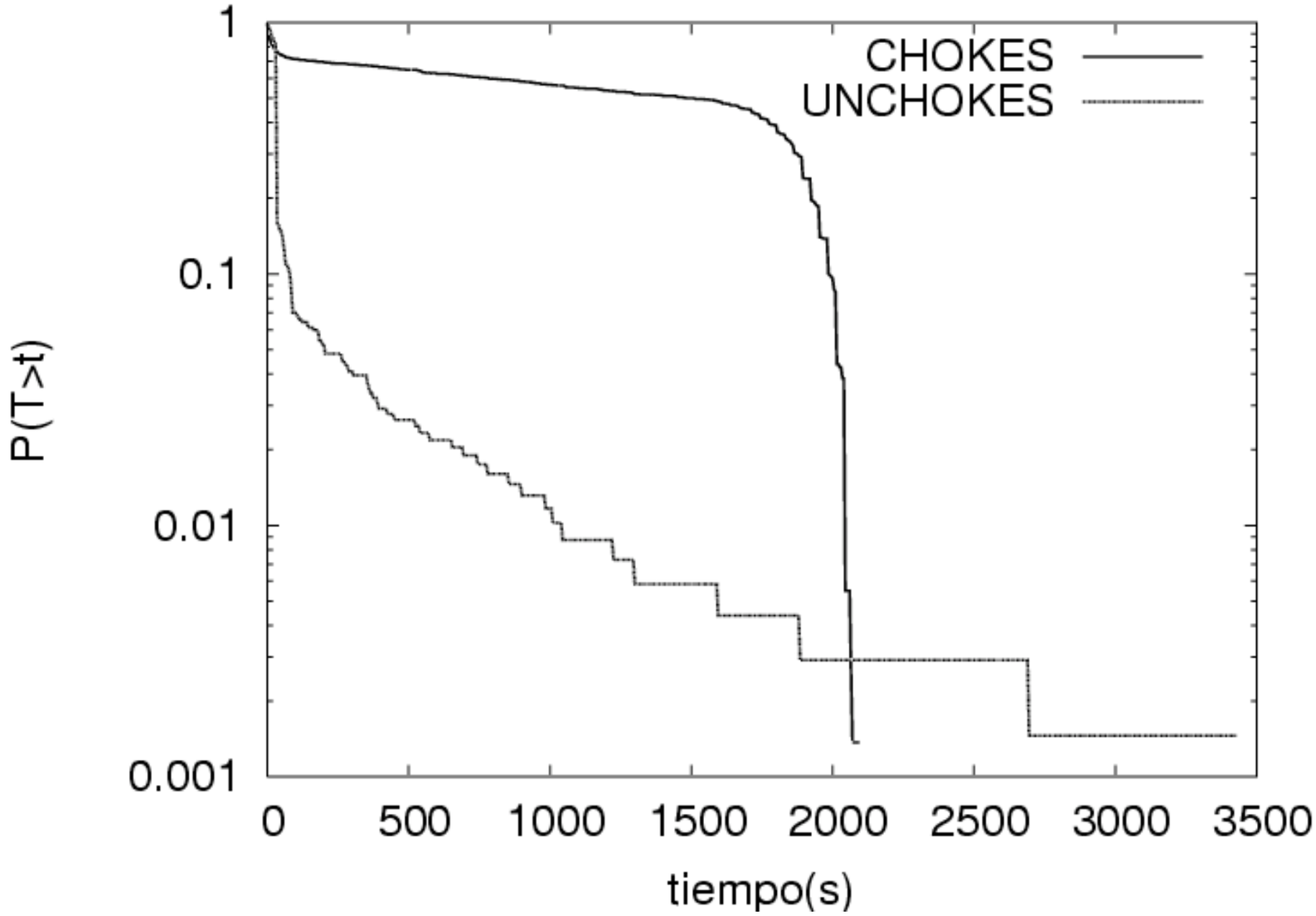



PEERS recientes
No hay (mucha) correlacion

PEERS veteranos
SI hay correlacion

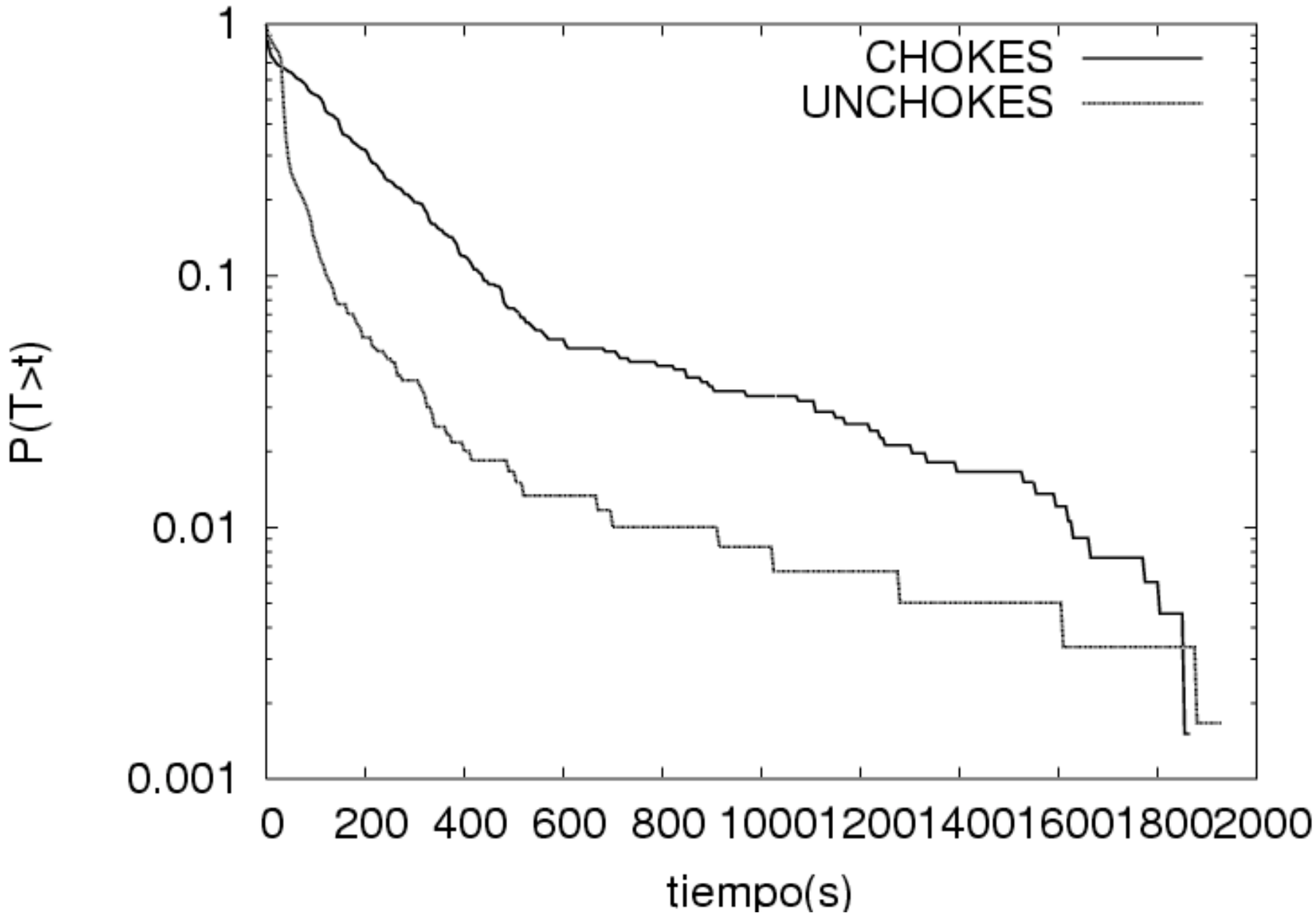
# Midiendo BitTorrent: de donde viene la ventaja?

>> Tiempos de CHOKE/UNCHOKE en envío
Cuanto tiempo estoy enviando o sin enviar a otro peer?



# Midiendo BitTorrent: de donde viene la ventaja?

>> Tiempos de CHOKE/UNCHOKE en recepción
Cuanto tiempo estoy recibiendo o sin recibir de otro peer?



# Medidas: conclusiones

- **Arranque lento hasta obtener piezas para negociar**

  (minutos en descargas de horas)
- **Los seeds contribuyen en gran medida**

  sobre todo para peers con buen BW de acceso

  BitTorrent es muy eficiente creando rápido SEEDS

# Medidas: conclusiones

- **Peers veteranos tienen ventaja sobre los recien llegados**
  - Tienen mayor número de piezas raras
  - Son capaces de mantener el tit-for-tat por más tiempo
  - Capturan más fácilmente el flujo de otros peers porque les envían más datos
- En general la velocidad de descarga aumenta con el tiempo
- En general los peers que llegaron antes acaban antes

**El torrente es aproximadamente FIFO**

# Contenido

- Introducción: cómo funciona BitTorrent
- Estado del arte de la investigación sobre p2p de tipo BitTorrent
- Medidas realizadas
- **Conclusiones y trabajos futuros**

# Conclusiones

- Mostrado el funcionamiento de BitTorrent y los principios en que se basa su alta eficiencia
- Analizado trazas de tracker y de peers de BitTorrent
  - Ventajas para peers antiguos en el torrente

# Lineas futuras

- Caracterización de trazas de BitTorrent y comparación con otros sistemas populares (eDonkey...)
  - Estudio del tráfico de sistemas peer-to-peer generales
- Estudio de la velocidad de download y los factores de que depende (S, L, limitación de upload, otros...)
  - ¿Se puede calcular que velocidad voy a obtener conociendo los parámetros del torrente?
- Estudio y modelado de la evolución de S y L
  - Estudio del sistema de ecuaciones (puntos de equilibrio...)
  - Comparación con torrentes reales