Estudio dinámico y propuesta alternativa para el sistema BitTorrent

Fabián Kozynski
Universidad ORT Uruguay
kozynski@ort.edu.uy

Andrés Ferragut
Universidad ORT Uruguay
ferragut@ort.edu.uy

Fernando Paganini
Universidad ORT Uruguay
paganini@ort.edu.uy

ABSTRACT

Dado el crecimiento de los sistemas *peer-to-peer* y en particular del protocolo BitTorrent para la distribución de contenido en redes, han surgido modelos que buscan comprender su funcionamiento y predecir su desempeño. En este trabajo realizamos estudios de simulación en NS-2 para verificar la validez de uno de estos modelos, que predice la operación alrededor de un equilibrio estable. Encontramos situaciones en las que el modelo predice los equilibrios adecuadamente, pero otras en los que se observan oscilaciones importantes, que pueden conducir a fallas de funcionamiento. A partir de estos resultados proponemos una variante del protocolo BitTorrent, tendiente a lograr mayor estabilidad. A través de simulaciones y del modelado matemático, evaluamos las mejoras de desempeño que ofrece esta variante.

Categories and Subject Descriptors

I.6.6 [**Simulation and Modeling**]: Simulation Output Analysis; H.1 [**Information Systems**]: Models and Principles

General Terms

Theory, Performance

Keywords

Peer-to-peer networks , BitTorrent, Simulations, Modelization, Choking algorithm

1. INTRODUCCIÓN

BitTorrent [1, 2, 8] es un sistema *peer-to-peer* de distribución de contenido en el cual un conjunto de usuarios busca descargar un conjunto determinado de archivos. Utilizar un sistema *peer-to-peer* permite que la capacidad de servicio del sistema escale junto con la demanda del mismo. Para lograr esto, los usuarios intercambian partes del contenido entre sí hasta lograr completar la descarga. Aquellos *peers* que se encuentran en la primer instancia, es decir, todavía están descargando contenido del sistema, son llamados *leechers*. Los *peers* que ya completaron la descarga son llamados *seeders*. Estos últimos tienen la posibilidad de permanecer en el sistema tanto tiempo como deseen, facilitando así el proceso de otros usuarios.

El protocolo BitTorrent consiste en una serie de mecanismos y heurísticas. Todo el sistema de archivos se divide en piezas (*chunks*), las cuales son indexadas para poder identificarlas y cuentan con un control de integridad. Cada usuario busca conseguir todas las piezas solicitándolas a aquellos *peers* con los que se encuentra conectado. Cuando un *leecher* descarga una de estas piezas en forma completa, avisa a todos sus contactos que la posee y que puede ser solicitada. Esto, junto con el envío de un bitfield marcando las piezas que tienen al momento del contacto inicial, permite a un *peer* mantener la información de estado, la lista de piezas que posee cada uno de sus contactos.

A partir de las solicitudes recibidas, cada *peer* decide a cuales otros *peers* autoriza a descargar durante el siguiente intervalo de tiempo (generalmente $10\,s$), proceso llamado *unchoking*. En esta selección se incorpora un mecanismo conocido como *tit-for-tat* (TFT), para incentivar la compartición y evitar los *free-riders* que no contribuyen al sistema. Dicho mecanismo consiste en autorizar a descargar (*unchoke*) a aquellos *leechers* que durante una ventana de tiempo han contribuido más contenido al *peer* en cuestión. Se agregan además autorizaciones extra (*optimistic unchoking*) de un *peer* al azar, para detectar potenciales nuevos colaboradores.

Una pregunta natural es si estos mecanismos de base heurística logran un desempeño adecuado, especialmente en una situación *dinámica* en la que *peers* entran y salen del sistema. Para ello se ha recurrido al modelado matemático, comenzando por [3] con modelos Markovianos estudiados en forma numérica. En [7] se presenta un modelo de fluido inspirado en el de [3], y se obtienen resultados analíticos en cuanto al equilibrio al que llega el sistema BitTorrent, y su estabilidad. Este modelo se resume en la Sección 2.

Un primer objetivo del presente trabajo es la evaluación por simulaciones de BitTorrent, buscando validar su desempeño y las conclusiones de este modelo. En la Sección 3 se describen estas simulaciones, basadas en la biblioteca [4] para NS-2. Observamos que según los parámetros empleados, el sistema puede comportarse como es debido (operando alrededor del equilibrio previsto por [7]), o por el contrario

generar graves oscilaciones que llevan a la falla, en particular la llamada *muerte* del torrent. Concluímos por tanto que el modelo tiene limitaciones en su poder predictivo, y lo que es más grave: el BitTorrent práctico también tiene problemas.

Un estudio detallado de la situación de falla nos lleva a proponer una variante al protocolo BitTorrent, descrita en la Sección 4. Esta variante es estudiada por simulación, y también en la Sección 5 a través del modelado matemático. Observamos que efectivamente se logra una mejora de desempeño en las situaciones problemáticas. Las conclusiones y líneas de trabajo a futuro se describen en la Sección 6.

2. MODELO FLUIDO DE QIU Y SRIKANT

Para modelar un sistema de BitTorrent, Qiu y Srikant proponen en [7] un modelo fluido. A partir de dicho modelo, obtienen una serie de resultados, los cuales se utilizarán como base para las siguientes secciones. A continuación se incluye una breve descripción del modelo, así como los resultados principales respecto de los equilibrios del mismo.

El modelo se encuentra basado en dos colas en cascada. Una de las colas, representada por x, corresponde a la cantidad de *leechers* en el sistema. La otra, y, corresponde a la cantidad de *seeders*. Ambas pueden tomar valores en el intervalo $[0, +\infty)$. La dinámica de estas dos variables se modela como se describe a continuación:

$$\dot{x} = \lambda - \min\{cx, \mu(y + \eta x)\} - \theta x \tag{1a}$$

$$\dot{y} = \min\{cx, \mu(y + \eta x)\} - \gamma y. \tag{1b}$$

- λ es la tasa de arribo de *leechers*, que llegan sin ningún contenido.
- μ es la capacidad de subida de un *peer* en archivos por segundo, por tanto $\mu(y+\eta x)$ es la capacidad de subida total del sistema, donde se incluye un parámetro $\eta \in [0, 1]$ que representa la eficiencia de la compartición entre *leechers*.
- $c > \mu$ es la capacidad máxima de bajada de un *peer*. La velocidad de bajada total está entonces determinada por $\min\{cx, \mu(y + \eta x)\}$; ésta determina la tasa de pasaje de *leechers* a *seeders*, como se indica en (1).
- También se incluye una tasa θ de abandono de los *leechers* , y una tasa γ de abandono de los *seeders*.

Se analizarán los equilibrios en el caso en que los *leechers* no abandonan el sistema antes de completar la descarga ($\theta = 0$). El punto de equilibrio de este sistema tiene dos casos, dependiendo de los valores de los parámetros.:

(i) Si $\frac{1}{\mu} < \frac{1}{\gamma} + \frac{\eta}{c}$, el equilibrio es

$$x^* = \frac{\lambda}{c}, \qquad y^* = \frac{\lambda}{\gamma}$$

y está limitado por la capacidad de bajada c . Es decir, el mínimo de (1) ocurre en cx^*.

(ii) Si $\frac{1}{\mu} > \frac{1}{\gamma} + \frac{\eta}{c}$, el equilibrio es

$$x^* = \frac{\lambda}{\eta}\left(\frac{1}{\mu} - \frac{1}{\gamma}\right) \qquad y^* = \frac{\lambda}{\gamma}, \tag{2}$$

y está limitado por la capacidad de subida.

Notar que para $c \gg \mu$, común en la práctica, los casos anteriores son aproximadamente (i) $\gamma < \mu$ y (ii) $\gamma > \mu$. El caso (i) se da cuando el tiempo medio $1/\gamma$ de permanencia de un *seeder* supera al tiempo $1/\mu$ necesario para reponer un archivo al sistema, generando otro *seeder*. Por tanto, aquí los *seeders* solos podrían sostener la carga, y se acumulan en el sistema hasta saturar la capacidad de bajada. El caso (ii) es el contrario: aquí los *seeders* solos no son suficientes para atender la demanda, y la compartición entre *leechers* se vuelve esencial. La cantidad de *leechers* en el sistema se ajusta en el equilibrio para proveer la tasa de subida faltante.

En [7] se prueba que en ambos casos el equilibrio es localmente estable. Más aún, en [6] se prueba la estabilidad *global* del sistema no-lineal conmutado (1).

También puede extraerse del modelo el tiempo medio de descarga del archivo. En el caso (i) es simplemente $\bar{T} = 1/c$, en el caso (ii) se obtiene

$$\bar{T} = \frac{1}{\eta}\left(\frac{1}{\mu} - \frac{1}{\gamma}\right). \tag{3}$$

3. ANÁLISIS POR SIMULACIONES

Para validar el modelo mencionado en la sección anterior se realizaron simulaciones en un sistema que reproduce los algoritmos del protocolo BitTorrent. Para esto se utilizó una biblioteca para BitTorrent implementada en NS-2 [4].

Las simulaciones se realizaron tomando valores para los parámetros que podrían ser los que se encuentren en un sistema real compuesto por usuarios domésticos. A partir de dichas simulaciones se relevó la evolución de las cantidades de *seeders* y *leechers* a lo largo de la simulación, y la evolución de la subida y la descarga para cada uno de los *peers*.

Todas las simulaciones se llevaron a cabo en un sistema compuesto por *peers* ubicados en una topología de estrella, todos conectados a un nodo central. Dichos *peers* intentan descargar un archivo de $100\,MB$ dividido en 400 piezas. Todos los *leechers* ingresan al sistema sin contenido y solo se van una vez que completaron la descarga, luego de permanecer un tiempo como *seeders*.

Para simular una situación real se utilizan arribos y partidas aleatorios. Para los arribos de *leechers* se utilizó un proceso de Poisson de parámetro λ, y para las partidas de *seeders* se tomó un tiempo exponencial de parámetro γ. El modelo determinístico (1) permite estudiar el comportamiento medio de este sistema aleatorio. Las oscilaciones alrededor de este valor medio son estudiadas en [7] considerando un modelo de ruido Browniano cosa que no haremos aquí por simplicidad. Por lo tanto nuestra validación se basará en los valores medios de las variables simuladas.

3.1 Caso limitado por capacidad de bajada

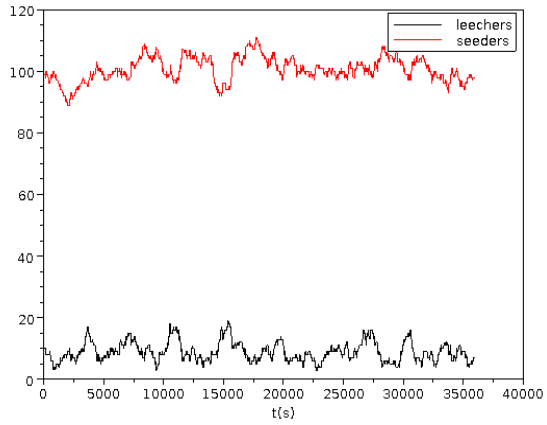

Figura 1: Evolución de un sistema en torno a un equilibrio limitado por capacidad de bajada.

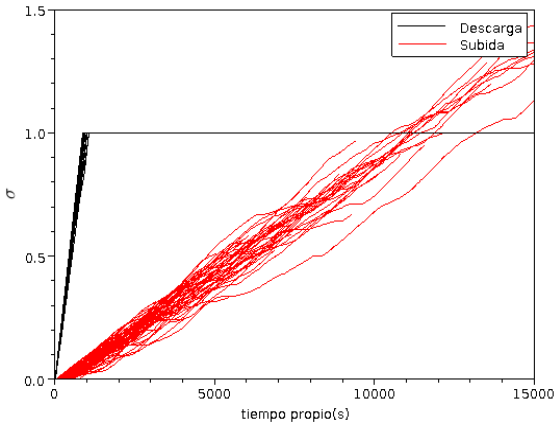

Figura 2: Transferencia acumulada de distintos *peers*, caso limitado por capacidad de bajada.

Primero consideramos el caso (i) de la Sección 2 (aproximadamente, $\gamma < \mu$), donde el modelo predice un equilibrio saturado por la capacidad c de bajada. Esta predicción se verificó efectivamente en numerosas simulaciones. A continuación mostramos un caso, con los siguientes parámetros: $\lambda = 0.01$ $^{\text{arribos}}/_{s}$, $\gamma = 10^{-4}\, s^{-1}$; $\mu \approx 1.5 \cdot 10^{-4}\, s^{-1}$, correspondiente a una velocidad de subida de $128\, Kbps$; $c = 8\mu \approx 1.2 \cdot 10^{-3}$, correspondiente a una velocidad de bajada de $1\, Mbps$. La simulación se comenzó con una cantidad de 100 *seeders* y 10 *leechers* con contenido aleatorio.

En la Figura 1 se observa la evolución del sistema. Comparando los valores medios tanto de *seeders* como de *leechers* en la simulación, se observa que éstos son similares a los del modelo. La cantidad de *leechers* en el equilibrio es de 8.9, cuando $x^* = \frac{\lambda}{c} = 8.19$; y la cantidad de *seeders* en el equilibrio es de 100.58, siendo $y^* = \frac{\lambda}{\gamma} = 100$.

También se midió empíricamente la velocidad de subida y de bajada de los distintos *peers*, a través de la gráfica de la Figura 2. Allí se muestra el volumen de descarga en función del tiempo para cada *peer*, a partir del momento de ingreso al sistema (tiempo propio). Por lo tanto las pendientes de las curvas corresponden a las velocidades de subida y bajada. Se verifica que la velocidad de bajada es uniforme entre los *peers*, y próxima al límite de $1\, Mbps$. Por otra parte, la velocidad de subida presenta mayor dispersión, y está por debajo del límite de $128\, Kbps$; esto es consistente con el modelo, estamos en el caso en que hay sobrante en la capacidad de subida.

3.2 Caso limitado por capacidad de subida

Se estudia ahora el caso (ii) de la Sección 2; se tomó $\mu \approx 3 \cdot 10^{-4}$ $^{\text{arch.}}/_{s}$ (correspondiente a una velocidad de subida de $256\, Kbps$) y $\gamma = 5 \cdot 10^{-4}\, s^{-1}$, lo cual corresponde a un tiempo medio de estadía de los *seeders* de aproximadamente media hora. Se cumple entonces $\mu < \gamma$, y también la restricción más estricta (ii) para la capacidad $c = 8\mu$ utilizada.

En esta simulación $\lambda = 0.01$ $^{\text{arribos}}/_{s}$; se comienza con 40 *seeders* y ningún *leecher*. En la Figura 3 se observa la evolución de los *seeders* y *leechers* a lo largo de la simulación. Ambos valores evolucionan presentando oscilaciones en torno a un equilibrio. Si bien el ruido aquí es mayor, el valor promedio de los *seeders* observados es 21, aproximadamente consistente con la cantidad $\frac{\lambda}{\gamma} = 20$ indicada por el modelo.

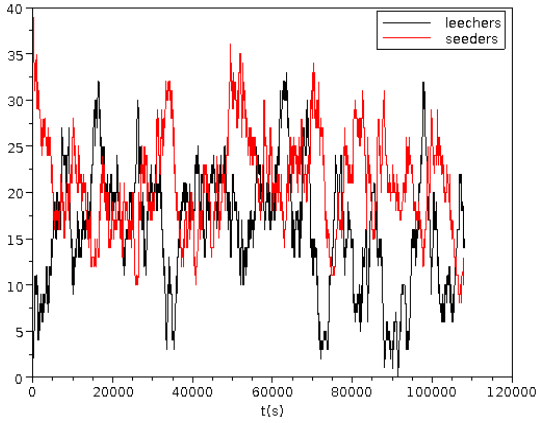

Figura 3: Evolución de un sistema en torno a un equilibrio limitado por capacidad de subida.

Para comparar la cantidad media de *leechers* con el modelo, es necesario conocer el parámetro η de eficiencia de la compartición. Estudiamos para ello las tasas de transferencia de los distintos *peers* a lo largo de la simulación. La Figura 4 muestra otra vez transferencias acumuladas.

Se observa ahora que las tasas de *subida* son uniformes entre *peers*, consistente con la saturación por subida. El valor obtenido empíricamente es $\mu' \approx 0.9\mu$; podría pensarse en atribuir esto al parametro η de eficiencia; sin embargo, una observación cuidadosa muestra que la velocidad de subida permanence constante, incluso después de que el *peer* se

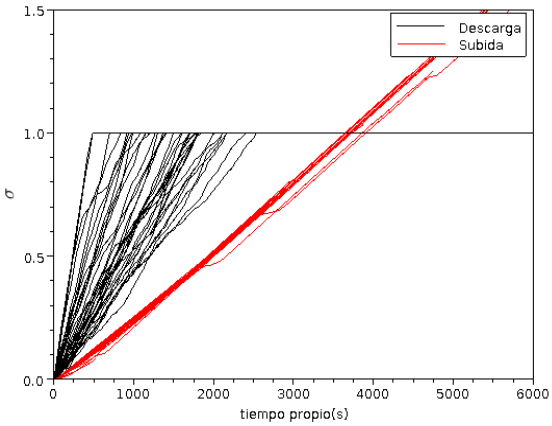

Figura 4: Transferencia acumulada de distintos *peers*, caso limitado por capacidad de subida.

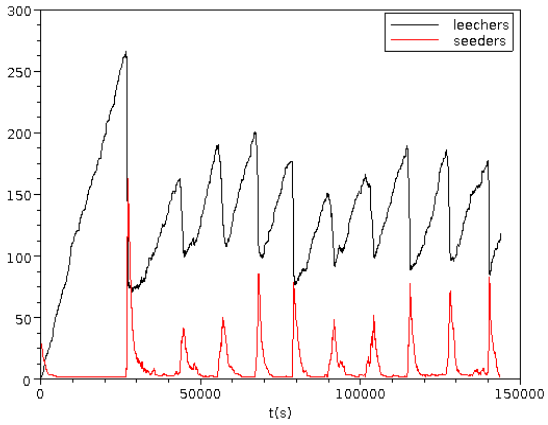

Figura 5: Evolución de un sistema con servidor que presenta oscilaciones.

transformó en *seeder*. Por lo tanto, el factor de eficiencia debe tomarse como $\eta \approx 1$ (consistente con un análisis de [7]), y la pérdida del 10 % se atribuye en cambio a un overhead general del sistema.

En ese sentido, notar que las gráficas de las Figura 4 se realizaron a partir de trazas de transferencias de carga útil (es decir, mensajes PIECE de BitTorrent), y no se encuentran considerados tanto los mensajes de control del protocolo como los distintos encabezados de las capas inferiores. Si contemplamos todo esto reemplazando μ por $\mu' \approx 0.9\mu$ en (2), y tomamos $\eta = 1$, el modelo predice una cantidad de *leechers* en el equilibrio de $\lambda\left(\frac{1}{\mu'} - \frac{1}{\gamma}\right) = 16.4$, mientras que la cantidad media de *leechers* en la simulación es aproximadamente 16. Por lo tanto concluimos que una vez más el modelo está validado en este caso.

3.3 Límitaciones del modelo

Consideramos ahora otra simulación, con parámetros $\lambda = 0.01\ {}^{\text{arribos}}/_{s}$, $\gamma = 10^{-3}\, s^{-1}$ (reduciendo el tiempo medio de permanencia de los *seeders* a $1000\, s$) y reduciendo μ a $7{\cdot}10^{-5}$ ($64\, Kbps$). Estamos por tanto en un caso de $\gamma \gg \mu$, en el que el modelo predice un equilibrio saturado por subida, y sostenido principalmente por la compartición de los *leechers*. En cambio, en la simulación se observa algo muy distinto: el torrent "muere" al poco tiempo. Esto significa que en cierto momento el sistema se queda sin *seeders*, y los *leechers* restantes no contienen entre todos una copia completa del archivo. Esta patología está prevista en el modelo, ya que éste no hace referencia a las piezas.

Para poder proseguir el estudio, se agregó un *seeder* permanente (servidor), el cual mantiene todas las piezas disponibles y evita la muerte del torrent. Se espera que el agregado de este servidor no tenga mucho impacto en el equilibrio, donde en este caso se contaría con $\lambda/\gamma = 10$ *seeders*.

Sin embargo, al realizar una nueva simulación contando con el servidor no se logra este equilibrio. Como muestra la Figura 5 tenemos en su lugar una oscilación cuasi-periódica, con

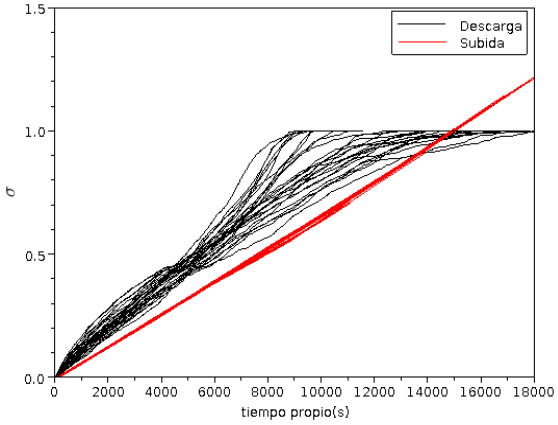

Figura 6: Transferencia acumulada de distintos *peers*, en un sistema con ciclo límite.

picos de creación y partida de *seeders*, y largos intervalos en los que el sistema queda vacío de *seeders*, dependiendo exclusivamente del servidor. Estas oscilaciones son demasiado sistemáticas para atribuirlas exclusivamente al ruido, indican una limitación del modelo.

Los picos de generación de *seeders* corresponden a un conjunto de *leechers* que culminan la descarga en un corto intervalo de tiempo. Dado que las llegadas de los *leechers* están distribuidas en el tiempo, esto implica que hay *leechers* que llegan más tarde y logran equipararse a los que llegaron antes, a través de una velocidad de descarga mayor. Esto en consistente con las gráficas de descarga de la Figura 6, correspondientes a un ciclo determinado, donde se ven velocidades de bajada no uniformes entre *peers* y con variaciones a lo largo del tiempo.

Notar que la velocidad de *subida* de todos los *peers* es prácticamente la misma y constante durante toda la evolución;

se mantiene la eficiencia observada en la simulación anterior.

La discrepancia observada entre modelo y simulación se atribuye entonces a dos aspectos no considerados en el modelo:

- La distribución desigual del ancho de banda entre *peers* con diferente grado de descarga;
- La entidad específica de las piezas intercambiadas, como se discute más abajo.

3.4 Estudio detallado de la falla observada en BitTorrent

Más allá de las limitaciones del modelo en su capacidad de predecir el comportamiento real, este comportamiento en sí mismo resulta problemático, ya que se observa la muerte del torrent. Esto ocurre en una situación en la que debería poder sostenerse el sistema a través de la compartición de *leechers*. Y si se evita la muerte a través de los servidores, estos juegan un papel demasiado central, sosteniendo el sistema durante un largo período, llevando a un comportamiento oscilatorio no deseable. Tenemos por tanto una limitación inherente al protocolo BitTorrent.

Observamos detenidamente la simulación con servidores, con especial atención a la descarga de aquellos *leechers* que se encuentran más tiempo en el sistema, a través de la evolución de los respectivos *bitfield*. Se detecta que existen usuarios con una gran cantidad de piezas (faltándoles no más de 5), los cuales durante mucho tiempo no logran completar ninguna pieza. Esto se debe a que dichas piezas no se encuentran en el sistema y sólo las poseen los servidores. Por lo tanto, necesitan que éstos les realicen un *unchoke* para obtenerlas y que así ingresar al sistema.

Se hace notar que BitTorrent establece la preferencia a solicitar piezas más escasas, justamente para atender este tipo de problema. Sin embargo, para conseguir las piezas faltantes, se pueden presentar dos situaciones. Por un lado, puede que los *leechers* interesados no consigan *unchoke* de los servidores, que son los únicos que poseen las piezas que necesitan. Por otro lado, los *leechers* a los que sí realizan *unchoke* los servidores, también ven otras piezas como escasas entonces no necesariamente solicitan las piezas faltantes. Estas dos situaciones llevan a que las piezas faltantes demoren más tiempo en ingresar al sistema.

Detectamos entonces una debilidad del sistema BitTorrent cuando opera con una proporción baja de *seeders*, es decir cuando los usuarios son egoístas y se retiran poco después de terminar su descarga. Si bien en teoría este sistema podría funcionar sosteniéndose principalmente en la compartición entre *leechers*, en la práctica se observa una tendencia a sincronizar las partidas de los *seeders* que conduce a la muerte del torrent.

4. BITTORRENT ELITISTA

El problema descrito en la sección anterior surge a partir de *leechers* que se estancan en la descarga de las últimas piezas debido a que no las consiguen. A partir de esto se decidió modificar el algoritmo de *unchoke* presente en el protocolo de forma de solucionar este problema.

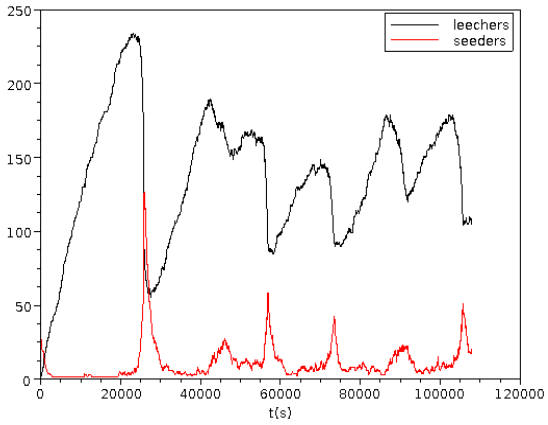

Figura 7: Evolución de un sistema patológico utilizando un cliente elitista.

La propuesta, que llamamos *BitTorrent elitista*, es que los *peers* dediquen sus puestos de subida (con la excepción del *Optimistic unchoke*) a aquellos *peers* que observan con la *mayor* cantidad de piezas. De esta manera, el sistema siempre buscaría crear *seeders* a partir de aquellos *leechers* que se encuentran cercanos a terminar la descarga. Con esta modificación se espera que se generen *seeders* a una tasa más estable, evitándose las avalanchas observadas anteriormente.

La modificación fue implementada en NS-2 modificando la aplicación (cliente) de [4], y se realizaron simulaciones cubriendo los distintos casos de funcionamiento.

4.1 Resultados de simulación

En términos generales, puede decirse que en los casos en que BitTorrent llegaba a un equilibrio, la variante elitista también lo hace. Además, las cantidades de *leechers* y *seeders* en equilibrio no varían significativamente. En la Sección 5 justificaremos este hecho mediante un modelo. Se observa que el sistema ahora presenta menores oscilaciones.

Una vez verificado que el cliente modificado no empeora las cosas, pasamos a ver si efectivamente logra una mejora en los casos problemáticos. En la Figura 7 se muestra la evaluación con los parámetros de la Sección 3.3. Si bien sigue habiendo oscilaciones, hay una gran diferencia: en este caso, existen *leechers* culminando la descarga en todo momento, lo que lleva a un comportamiento más suave del sistema con un número de *seeders* que no llega casi nunca a cero (exceptuando el transitorio inicial); esto implica que la supervivencia del Torrent no depende crucialmente de los servidores.

Se pudo observar mediante simulaciones con distintos valores de parámetros, que aquellos casos en los cuales se observaba un comportamiento oscilatorio se vieron reducidos significativamente. Además, en todos los casos se mantiene una compartición eficiente, donde la velocidad de subida de todos los *peers* se encuentra saturada.

4.2 Mejora en la justicia del sistema

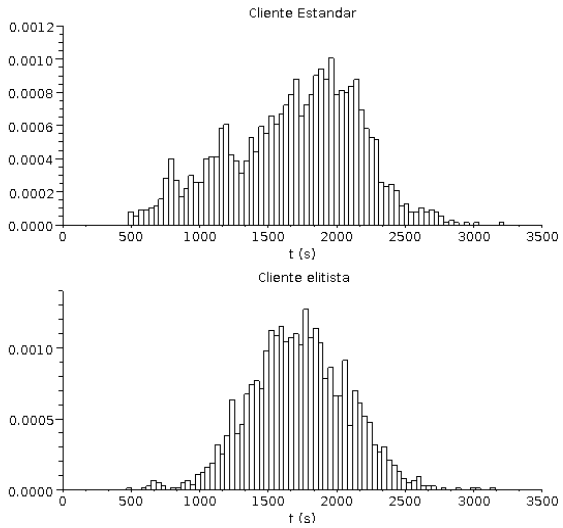

Figura 8: Duración de las descargas utilizando clientes estándar y clientes elitistas.

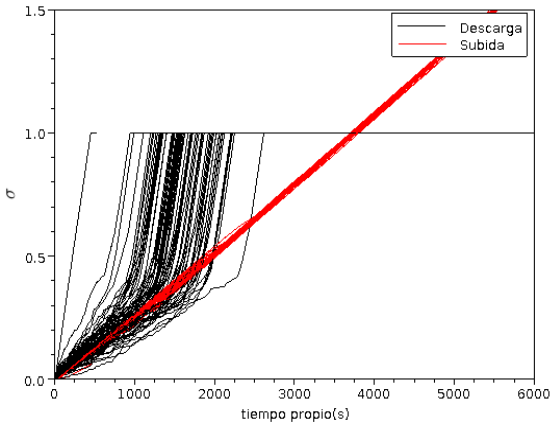

Figura 9: Evolución de la descarga de distintos *peers* en un sistema elitista.

Superficialmente, podría pensarse que un sistema elitista que favorece los *unchoke* a *leechers* más "ricos" puede generar injusticias, por ejemplo en los tiempos de descarga. Sin embargo, ocurre lo contrario.

En la Figura 8 se observa la distribución de las duraciones de las descargas en dos simulaciones con los mismos parámetros, utilizando en cada una un cliente distinto. En ambas simulaciones el sistema trabaja en torno a un equilibrio, con parámetros similares a los de la simulación de la Figura 3.

Una primera observación es que los valores *medios* de los tiempos de descarga no cambian, son consistentes con la fórmula (3) (reemplazando con $\mu' \approx 0.9\mu$). Esto es también consistente con el número medio de *peers* en equilibrio, que como se dijo tampoco cambia.

La diferencia principal observada en los histogramas es en la varianza de los tiempos de descarga, que disminuye claramente en el cliente elitista; se pasa de una desviación estándar de $475\,s$ para clientes comunes a un valor de $355\,s$ para los elitistas. Esta disminución de la varianza mejora la justicia del sistema: los tiempos de descarga son más parejos entre los distintos *leechers*.

La explicación de esta menor varianza se atribuye a la disminución de las oscilaciones. En un sistema saturado por subida, la velocidad de descarga que experimentan los *leechers* en un cierto período de tiempo es proporcional a la cantidad de *peers* en el sistema durante dicho período. Si esta cantidad no varía significativamente en el tiempo, *leechers* que ingresan en distintos momentos de la simulación experimentan una red de capacidad más estable. Si bien esta capacidad ahora *no* se distribuye uniformemente entre los *leechers*, sino que hay un sesgo hacia los "ricos", esto no afecta el tiempo total de descarga, ya que todo *peer* aprovecha esta ventaja sobre el final de su descarga.

En otras palabras, ahora los *leechers* tienen una fase en que no son privilegiados y avanzan lentamente, y a partir de cierto momento son privilegiados y terminan rápidamente. Estas dos fases pueden observarse en la gráfica de descarga de la Figura 9. En esa figura, en la etapa privilegiada los *leechers* saturan por capacidad de bajada, logrando un desempeño confiable; las variaciones entre *leechers* se atribuyen principalmente a la primer fase, por lo que no sorprende que la varianza total sea de menor entidad.

5. MODELADO DEL SISTEMA ELITISTA

Para entender mejor el cliente elitista, se realizará un modelo que cuente con algo de información de la distribución de los *leechers* en el sistema según la cantidad de contenido descargado. Nos concentramos en el caso (ii) de la Sección 2, restringido por la capacidad de subida, con $\mu < \gamma$, y con capacidad de bajada c suficientemente grande como para ser no ser incluida en el modelo.

Para esto, se dividirá el estado x de los *leechers* del modelo introducido en la Sección 2 en dos estados n_1 y n_2. Dichos estados corresponderán a *leechers* que tienen menos de la mitad del contenido descargado (n_1) y más de la mitad del contenido descargado (n_2).

Suponiendo que los *leechers* comparten contenido con máxima eficiencia, la velocidad de descarga total será $\mu(y + n_1 + n_2)$ archivos por segundo o equivalentemente, $2\mu(y+n_1+n_2)$ mitades de archivo por segundo. Esta última expresión es la relevante, ya que los avances de n_1 a n_2, y de n_2 a y, corresponden a descargar medio archivo.

Para el cliente elitista, dicha capacidad total no se distribuirá en forma equitativa entre ambos estados. Se introduce la variable ρ, la cual representa el cociente entre el porcentaje de capacidad correspondiente a n_2 y el correspondiente a n_1. A partir de esto, el sistema de ecuaciones diferenciales que

determina la dinámica del sistema queda así:

$$\begin{cases} \dot{n_1} = \lambda - 2\mu(y + n_1 + n_2)\dfrac{n_1}{n_1 + \rho n_2}, \\ \dot{n_2} = 2\mu(y + n_1 + n_2)\left[\dfrac{n_1}{n_1 + \rho n_2} - \dfrac{\rho n_2}{n_1 + \rho n_2}\right], \\ \dot{y} = 2\mu(y + n_1 + n_2)\dfrac{\rho n_2}{n_1 + \rho n_2} - \gamma y. \end{cases}$$

A partir de dicho sistema, es posible obtener el estado de equilibrio:

$$n_1^* = \frac{\lambda\rho}{\rho + 1}\left(\frac{1}{\mu} - \frac{1}{\gamma}\right),$$
$$n_2^* = \frac{\lambda}{\rho + 1}\left(\frac{1}{\mu} - \frac{1}{\gamma}\right),$$
$$y^* = \frac{\lambda}{\gamma}.$$

Se puede observar que la cantidad total de *leechers*, $n_1^* + n_2^*$ es independiente de ρ y es igual al valor de x^* dado en (2) para un sistema de eficiencia $\eta = 1$. Por lo tanto, la forma en que se destine el ancho de banda disponible, no modifica la cantidad de *leechers* en el equilibrio, Además, la cantidad de *seeders* en el equilibrio sigue siendo $\frac{\lambda}{\gamma}$. Estos resultados pueden generalizarse para modelos con más estados intermedios y comparticiones más generales (ver[5]).

Para estudiar la dinámica de este sistema, se realiza la linealización en torno al punto de equilibrio. Normalizando el sistema con $\mu = 1$, se obtiene la siguiente matriz jacobiana

$$J = \begin{bmatrix} \frac{(2-3\gamma)\rho-\gamma}{2(\gamma-1)\rho} & \frac{\gamma(\rho-1)+2}{2(\gamma-1)} & -1 \\ \frac{\gamma(\rho+1)}{(\gamma-1)\rho} & -\frac{\gamma(\rho+1)}{\gamma-1} & 0 \\ \frac{(\gamma-2)\rho-\gamma}{2(\gamma-)\rho} & \frac{\gamma(\rho+3)-2}{2(\gamma-)} & 1-\gamma \end{bmatrix}.$$

Dicha matriz presenta siempre un valor propio real negativo, y dos valores dominantes complejos. Mediante el criterio de Routh - Hurwitz es posible determinar que todos sus valores propios poseen parte real negativa y por lo tanto el sistema lineal es siempre estable. Para analizar el efecto de la compartición en los valores propios de la matriz, se obtuvieron las curvas de dichos valores propios para distintos valores de $\gamma > 1$, variando el valor de ρ desde $\rho = 1$ que corresponde al cliente "igualitario", a $\rho > 1$ que modela al cliente elitista.

En la Figura 10 se observa como varían los valores propios complejos de J. Cada curva representa un valor de γ distinto y muestra la variación de los valores propios al variar ρ, aumentando desde $\rho = 1$ en la curva negra. Se puede observar que en todos los casos, para un valor de γ determinado, al aumentar ρ el sistema se amortigua. En la Figura 11 se muestra el cambio del amortiguamiento para distintos valores de γ. Esto explica cualitativamente la disminución de las oscilaciones.

6. CONCLUSIONES

En este trabajo se estudió la dinámica del sistema BitTorrent, buscándose validar el modelo de Qiu y Srikant [7]. Para esto se realizaron simulaciones NS-2 de la dinámica en distintas situaciones de los parámetros. Se observó que los

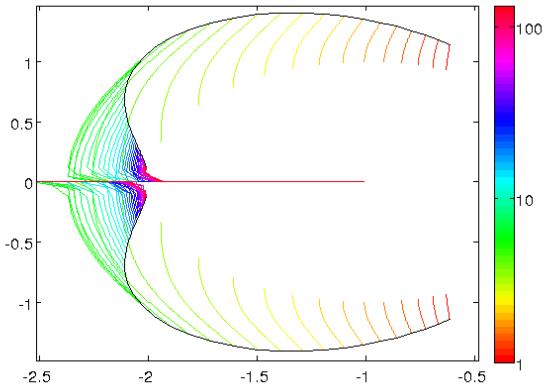

Figura 10: Lugar de los valores propios complejos de J.

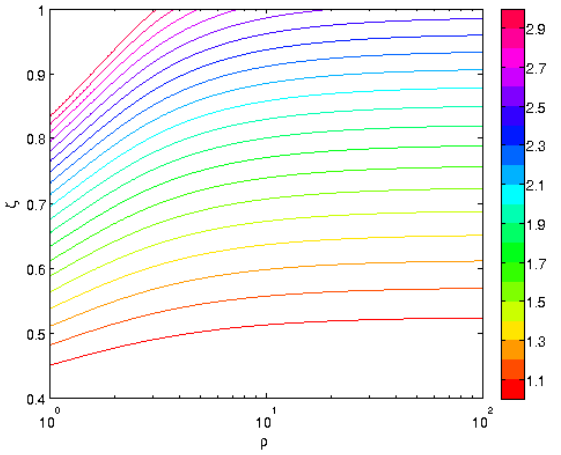

Figura 11: Amortiguamiento en función de ρ y parametrizado en γ.

valores de equilibrio son los esperados. Además de esto, a partir de las simulaciones se pudo concluir que la eficiencia de la compartición de los *peers* es prácticamente 1, es decir no hay oportunidades de intercambio desaprovechadas.

Sin embargo, existen situaciones en las cuales el sistema no alcanza un equilibrio, llegándose a la situación de *muerte* del torrent. Si se fuerza la supervivencia mediante un *seeder* permanente, se presenta una oscilación indeseada, que va más allá de la natural aleatoriedad del sistema. Se observan aquí dificultades de los *leechers* para conseguir sus últimas piezas. A pesar de estos problemas, la eficiencia del sistema continúa siendo 1.

De forma de solucionar este problema, se planteó una modificación al algoritmo de *unchoke* original. El *BitTorrent elitista* consiste en obligar a todos los *peers* a destinar sus *unchokes* a aquellos contactos que poseen la mayor cantidad de *chunks* y por lo tanto se encuentran más cercanos a completar la descarga. Esto se realizó con la idea de disminuir el tiempo de espera de aquellos *leechers* que no consiguen

su pieza final y de esta manera, generar *seeders* en forma no tan concentrada en el tiempo.

El cambio propuesto logra solucionar en parte el problema presentado, consiguiendo que el sistema presente menores oscilaciones y disminuyendo la dependencia de los *seeders* permanentes. Como beneficio extra, se observó que el cambio realizado disminuye la variación entre los tiempos de descarga de los distintos *peers*, aumentando así la justicia del sistema.

Extendiendo el modelo de [7] de forma de incluir la modificación realizada, es posible ofrecer una explicación a la disminución de las oscilaciones. Al estudiar la dinámica del modelo, se observa que al destinar mayor cantidad de recursos a aquellos *leechers* más avanzados en la descarga, el sistema presenta modos de mayor amortiguamiento.

En un futuro trabajo, debería investigarse más a fondo al nuevo cliente elitista para determinar si su efecto siempre es beneficioso. Esto incluiría estudios por simulación o experimentos en situaciones más diversas, no sólo la considerada con arribos estacionarios. También, estudiar la posibilidad de coexistencia con clientes de BitTorrent que implementen el algoritmo original.

Otro punto importante a considerar es el tema de *incentivos* para que los *leechers* compartan su contenido. El sistema TFT habitual ofrece incentivos de este tipo; la variante propuesta no considera ese aspecto. Sería necesario incluir algún mecanismo que requiera que los *leechers* colaboren con el sistema para ser considerados para los *unchokes*.

7. AGRADECIMIENTOS

Este trabajo fue financiado parcialmente por ANII-Uruguay, beca BE_INI_2010_2106 y proyecto FCE2009_1_2158.

8. REFERENCIAS

[1] B. Cohen. Incentives build robustness in BitTorrent, 2003. In *1st Workshop on the Economics of Peer-2-Peer Systems, Berkley*, 2003.

[2] B. Cohen. Bep 3: The bittorrent protocol specification, Feb. 2008.

[3] G. De Veciana and X. Yang. Fairness, incentives and performance in peer-to-peer networks. *Seeds*, 250:300–350, 2003.

[4] K. Eger, T. Hoßfeld, A. Binzenhöfer, and G. Kunzmann. Efficient simulation of large-scale p2p networks: packet-level vs. flow-level simulations. In *Proceedings of the second workshop on Use of P2P, GRID and agents for the development of content networks*, UPGRADE '07, pages 9–16, New York, NY, USA, 2007. ACM.

[5] F. Kozynski. Optimización de redes *peer-to-peer*. Technical report, Universidad ORT Uruguay, 2011.

[6] D. Qiu and W. Sang. Global stability of peer-to-peer file sharing systems. *Computer Communications*, 31(2):212–219, 2008.

[7] D. Qiu and R. Srikant. Modeling and performance analysis of BitTorrent-like peer-to-peer networks. *ACM SIGCOMM Computer Communication Review*, 34(4):367–378, 2004.

[8] Theory.org. Bittorrent protocol specification v1.02, Aug. 2010.