Algoritmos de gestión de tráfico: Leaky Bucket, Token Bucket y Virtual Scheduling

Traffic management algorithms: Leaky Bucket, Token Bucket and Virtual Scheduling

Gina Katherín Sierra Páez
Ingeniera electrónica, estudiante de la Maestría en Ciencias de la Información y las Comunicaciones de la Universidad Distrital Francisco José de Caldas. Bogotá, Colombia. *ksierrap@udistrital.edu.co*

Judy Carolina Guevara Amaya
Ingeniera en control, estudiante de la Maestría en Ciencias de la Información y las Comunicaciones de la Universidad Distrital Francisco José de Caldas. Bogotá, Colombia. *jcguevaraa@correo.udistrital.edu.co*

Clasificación del artículo: Revisión (Recreaciones)

Fecha de recepción: 14 de mayo de 2011 **Fecha de aceptación: 29 de agosto de 2011**

Palabras clave: Control de congestion, Leaky Bucket, Token Bucket, Virtual Scheduling.

Key words: Congestion control, Leaky Bucket, Token Bucket, Virtual Scheduling.

RESUMEN

Este artículo presenta los tres algoritmos principales empleados para el control de congestión en redes de comunicaciones: Leaky Bucket, Token Bucket y Virtual Scheduling. Su objetivo es evitar que el tráfico llegue a niveles inaceptables de congestión. También se presentan algunas de las modificaciones hechas a estos algoritmos y sus aplicaciones más destacadas.

ABSTRACT

This paper presents the three main algorithms used for congestion control in communication networks: Leaky Bucket, Token Bucket and Virtual Scheduling. Its aim is to avoid traffic reaches unacceptable levels of congestion. It also offers some of the modifications made to these algorithms and their main applications.

1. INTRODUCCIÓN

Una de las principales ventajas de las redes de conmutación de paquetes es su habilidad para asignar dinámicamente el ancho de banda que los usuarios requieren en instantes particulares. Debido a que las redes son sujetas a rápidas variaciones en la demanda, debe asegurarse un desempeño aceptable bajo condiciones de picos de tráfico. El problema de controlar los efectos de estos picos es particularmente serio en redes orientadas a la no conexión, las cuales tienen una capacidad limitada para restringir la demanda total. En una red orientada a la conexión, el ancho de banda puede ser asignado a nuevos usuarios cuando las conexiones son establecidas y pueden ser rechazadas nuevas conexiones si no hay suficiente ancho de banda disponible, lo que asegura un desempeño predecible una vez se establece una conexión. Las redes orientadas a la no conexión, por otra parte, responden a sobrecargas degradando el desempeño visto por todos los usuarios.

El control de congestión se refiere a la colección de métodos usados para asegurar a cada usuario un desempeño aceptable sobre condiciones de tráfico variable [1]. Las principales razones por las cuales este desempeño puede degradarse son:

- La tasa de llegada de paquetes excede la capacidad del enlace de salida.
- No hay memoria suficiente para almacenar los paquetes que llegan.
- Existen ráfagas de tráfico.
- Hay un lento procesamiento.

Como un método para evitar la congestión, se da "forma" al tráfico antes de que ingrese a la red, controlando la velocidad a la que se envían los paquetes. Este método comúnmente se utiliza en redes ATM [2], [4] y en redes de servicios integrados [5], [8]. Este artículo presenta tres de los algoritmos más comunes en la formación del tráfico: Leaky Bucket, Token Bucket y Virtual Scheduling.

El algoritmo Leaky Bucket se utiliza para controlar la tasa de transmisión de una red y se implementa como una cola de un único servidor con tiempo de servicio constante [9]. Si el buffer se desborda, entonces los paquetes se descartan. También se encuentra en la literatura diferentes esquemas de este algoritmo que buscan mejores resultados en comparación con su contraparte tradicional: DRLB (Dynamic Rate Leaky Bucket) [10], Dual-Leaky-Bucket [11], Token-Bank Leaky Bucket [12].

En contraste con el Leaky Bucket, el algoritmo Token Bucket, permite variaciones en la tasa de salida, dependiendo del tamaño de la ráfaga. En este algoritmo, los tokens son generados por un reloj, a una tasa de un token cada Δt segundos y son almacenados en un buffer. Para transmitir un paquete, el host debe capturar y destruir un token. Cuando se presenta inactividad, el host puede capturar y guardar tokens (hasta el máximo tamaño del buffer) para enviar grandes ráfagas posteriormente.

Diversos autores han abordado este algoritmo desde diferentes enfoques. Su modelo analítico por ejemplo, es tratado en [13] y algunos aspectos matemáticos en [14]. En [15], [16] se presentan algunas mejoras y variaciones al algoritmo. Inclusive, se ha llegado a usar técnicas de inteligencia computacional sobre el algoritmo convencional mostrando mejores resultados [17]. Sus aplicaciones abarcan las redes WLANs [18], WiMAX [19], Ad hoc [20], redes en malla [21] y redes ópticas [22], transmisión de video [23] y voz [24] entre otros campos. Su implementación por otra parte, se ha realizado sobre Linux [25] y FPGAs [26].

Finalmente, el algoritmo Virtual Scheduling (VS) monitorea directamente la tasa de llegadas de las

celdas. Cuando la diferencia en tiempos de llegada de dos celdas consecutivas es menor al tiempo teórico de llegada (en inglés, TAT) sumado a un valor de tolerancia, las celdas son consideradas no conformes y por tanto son descartadas [27]. Este algoritmo es comúnmente utilizado en redes ATM [3] y redes industriales inalámbricas [28], [29].

En el siguiente numeral se presenta el algoritmo Leaky Bucket y sus variaciones, la sección III describe el algoritmo Token Bucket junto con las modificaciones que se han planteado al respecto. La sección IV trata el algoritmo Virtual Scheduling. En la sección V se comparan los tres algoritmos. La sección VI menciona algunas de las aplicaciones de estos algoritmos. El artículo termina con las conclusiones que se presentan en la sección VII.

2. ALGORITMO LEAKY BUCKET

El algoritmo Leaky Bucket tiene por objeto regular la tasa media de flujo de tráfico, incluso en presencia de ráfagas ocasionales de datos. Se basa en la tasa de datos para controlar la velocidad máxima del tráfico procedente de una fuente. Si la tasa de entrada de datos es menor que la tasa máxima especificada por el algoritmo, los datos son aceptados. Si la tasa de entrada de datos excede la tasa máxima, el algoritmo transmite los datos a la tasa máxima de datos y el exceso de datos es almacenado en un buffer. Si el buffer está lleno, entonces se descarta el exceso de datos [30].

De acuerdo con esto, se define λ_a como la tasa de entrada promedio de datos, estimada como el número de paquetes (N) enviados en el intervalo de tiempo t , es decir, $\lambda_a = N/t$. Esta tasa se compara con la tasa máxima λ_a especificada por el algoritmo Leaky Bucket. Dependiendo de la tasa de llegada de datos y el estado de ocupación del buffer, se pueden presentar los siguientes escenarios:

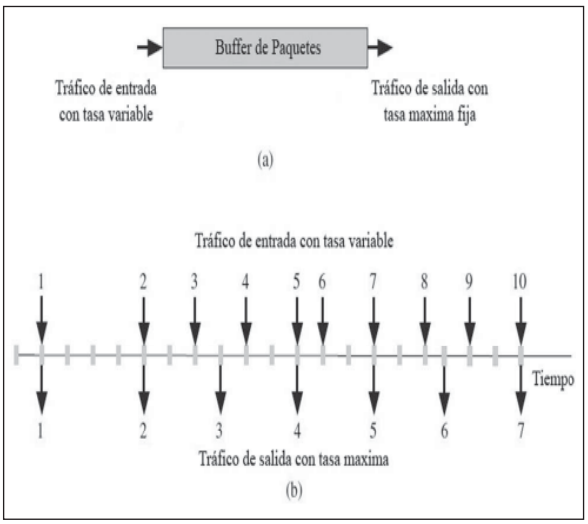

Fig. 1. El algoritmo Leaky Bucket suaviza el tráfico entrante con tasas variables por medio de su almacenamiento en el buffer y regula el tráfico máximo de salida del buffer: (a) Buffer de paquetes; (b) Llegada y salida de paquetes (tomado de [30]).

a.) $\lambda_a < \lambda_b$: la tasa de llegada de datos (λ_a) es más baja que la tasa máxima (λ_b) especificada por el algoritmo. En este caso, los datos son aceptados tal y como se observa en la llegada de los paquetes 1 y 2 de la Fig. 1. En la figura se asume que la tasa máxima de paquetes es equivalente a tres pasos o intervalos de tiempo. Tan pronto como los paquetes llegan, son transmitidos.

b.) $\lambda_a > \lambda_b$: los datos llegan a una tasa superior a λ_b y el buffer de datos no está lleno. En este caso, los datos son almacenados de forma que a la salida se transmiten a una tasa máxima λ_b. Esto se observa en la llegada de los paquetes 3, 4 y 5 de la Fig. 1 donde el espaciamiento entre paquetes es equivalente a la tasa de transmisión de datos λ_a.

c.) $\lambda_a > \lambda_b$: los datos llegan a una tasa superior a λ_b, y el buffer de datos está lleno. En este caso los datos son descartados. Como se muestra en la llegada de los paquetes 6 y 7 de la Fig. 1.

2.1 Variaciones del algoritmo Leaky Bucket

En la sección anterior se describió el algoritmo Leaky Bucket convencional. Sin embargo, en la literatura se encuentran algunas variaciones al algoritmo.

Para abordar el problema del control de congestión debido a las ráfagas naturales de las diferentes fuentes de tráfico en redes ATM, los parámetros UPC/NPC (user parameter control/network parameter control) han sido ampliamente estudiados. El algoritmo DRLB (Dynamic Rate Leaky Bucket) en el cual la tasa de generación de señal cambia dinámicamente de acuerdo con los estados de las fuentes de datos y a la ocupación del buffer, es un buen ejemplo de los parámetros UPC/NPC. Sin embargo, el algoritmo DRLB presenta varias desventajas como la baja eficiencia y difícil aplicación en tiempo real para fuentes de tráfico con ráfagas, debido a que la determinación de la tasa de generación de señal en el algoritmo se basa en el estado actual de la red. Por tanto, en [31] se propone un algoritmo de control de congestión más eficaz mediante la combinación del algoritmo DRLB y la predicción de las redes neuronales para remediar los inconvenientes del algoritmo DRLB.

En [10] se presenta una variación al algoritmo donde la tasa de generación de tráfico varía de acuerdo con el estado de una fuente de datos on-off. El algoritmo propuesto requiere buffers más pequeños que el algoritmo de tipo estático para satisfacer la misma calidad de servicio (QoS).

En [32] se proponen dos algoritmos Leaky Bucket inteligentes para el control de tráfico en redes ATM. Uno de ellos es el algoritmo Leaky Bucket Difuso, en el cual un controlador de incremento difuso (FIC) es incorporado al algoritmo Leaky Bucket convencional; el otro es el algoritmo Leaky Bucket Neuro Difuso, donde un controlador incremental neuro difuso (NFIC)

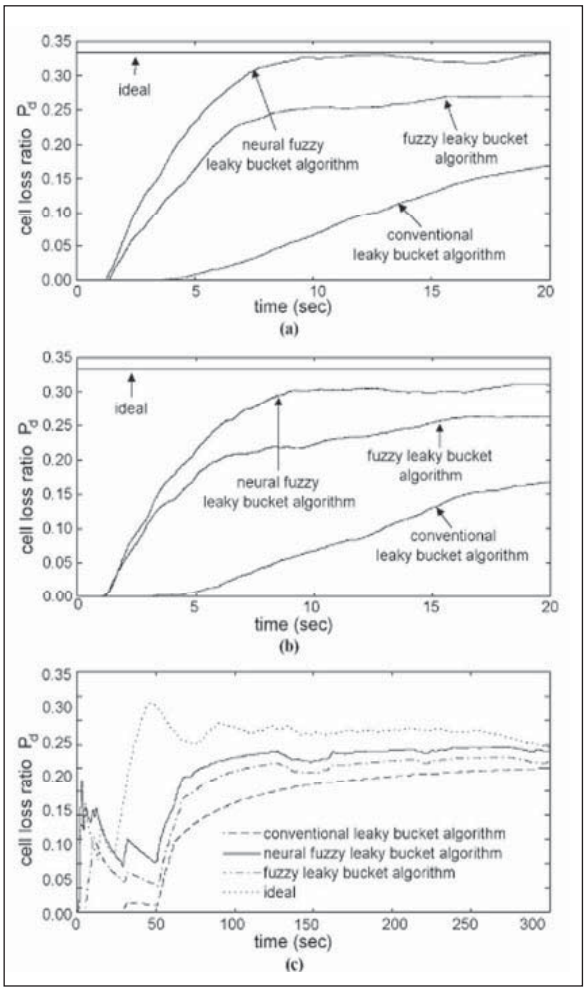

Fig. 2. Respuesta del algoritmo Leaky Bucket convencional, el algoritmos Leaky Bucket difuso y el algoritmo Leaky Bucket neuro difuso sobre: (a) fuentes de tráfico MMDP;(b) fuentes de tráfico MMBP; y (c) fuentes de tráfico de video MPEG (tomado de [30]).

es adherido al algoritmo Leaky Bucket convencional. Tanto en FIC como el NFIC, eligen apropiadamente la tasa media de celdas a largo plazo y la tasa media de celdas a corto plazo como variables de entrada para determinar de forma inteligente el valor de incremento. En la Fig. 2 se observan los resultados de simulación, los cuales superan el algoritmo convencional. De manera similar, en [33] se propone un esquema adaptable de control difuso de tráfico basado en

el algoritmo Leaky Bucket con el fin de resolver el problema de la congestión del tráfico en las redes inalámbricas.

En [12] se propone el algoritmo Token-Bank Leaky Bucket, para grupos de conexiones en redes ATM. El mecanismo explora la multiplexación estadística de conexiones múltiples en el grupo y permite compartir el ancho de banda no utilizado por las conexiones que lo necesitan. Adicionalmente, establece un límite de celdas de datos excesivos que una fuente puede enviar, con el fin de proteger las fuentes de buen comportamiento frente a las fuentes maliciosas.

3. EL ALGORITMO TOKEN BUCKET

El algoritmo Token Bucket, también encontrado en la literatura como TBF (Token Bucket Filter) [34], [35], es una disciplina de colas sencilla que permite el paso de paquetes que no exceden una tasa de transferencia límite impuesta administrativamente, pero acepta ráfagas cortas que excedan dicha tasa. Su operación, como lo ilustra la Fig.3 se basa en la emisión de tokens generados por un reloj cada ΔT segundos, de manera que si llega un paquete de longitud l (bytes) y hay al menos l tokens en el buffer, el paquete es enviado y se eliminan l tokens.

La Fig. 4(a) ilustra tanto el buffer de entrada de datos como el buffer de tokens, los cuales se realimentan mutuamente con el fin de ofrecer a la salida la misma tasa, tanto para los tokens como para los paquetes en dependencia de los estados de ambas colas.

La Fig. 4(b), muestra los eventos de llegada y partida de paquetes, así como la llegada de tokens, representada por los círculos grises. De acuerdo con la tasa de llegadas y el estado de ocupación del buffer de tokens, se pueden presentar los siguientes casos en la implementación de este algoritmo [30]:

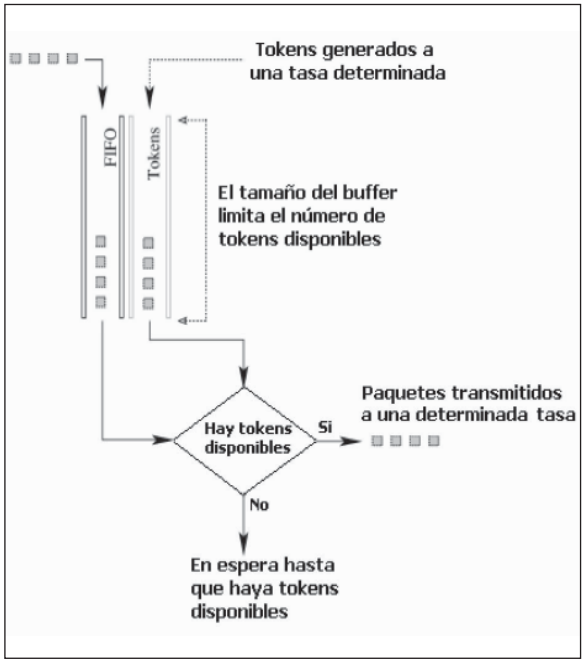

Fig. 3. Algoritmo TBF (tomado de [36]).

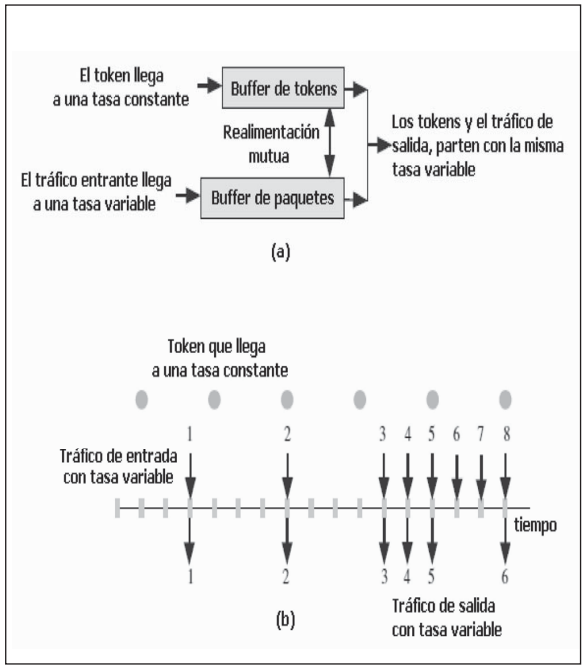

Fig. 4. El algoritmo Token Bucket suaviza el tráfico entrante con tasas variables almacenándolo y regulando la tasa máxima de salida de tráfico del buffer. (a) Buffer de paquetes; (b) Llegada y salida de paquetes (tomado de [30]).

a.) No llegan datos y los tokens son almacenados en el buffer, representando un ahorro para la fuente. Esto es similar a la situación que se presenta antes de la llegada del paquete 1 en la Fig. 4(b).

b.) Los datos llegan a una tasa menor que la tasa a la que se emiten los tokens. En esos casos, los datos serán aceptados y el exceso de tokens será almacenado en el buffer de tokens como ahorro para la fuente. Esto es similar a lo que ocurre con la llegada de los paquetes 1, 2 y 3 en la Fig. 4(b). Llegan más tokens que datos y el buffer de tokens comienza a llenarse.

c.) Los datos llegan a una tasa más alta que la tasa a la que son emitidos los tokens. En esos casos, los datos serán aceptados siempre y cuando haya tokens en el buffer. Esto es similar a lo que sucede con la llegada de los paquetes 4 y 5 en la Fig. 4(b). Aunque los paquetes 4 y 5 no viajan a través de tokens, han llegado gracias a que el buffer no estaba vacío.

d.) Los datos llegan pero no hay tokens presentes. Solo una porción de los datos será aceptada a una tasa igual a la que son generados los tokens. El exceso de datos puede ser descartado o etiquetado como no conforme. Esto es similar a lo que ocurre con la llegada de los paquetes 6, 7 y 8 en la Fig. 4(b). Los paquetes 6 y 7 llegan cuando el buffer de tokens está vacío, razón por la cual son almacenados en el buffer de paquetes y cuando llega un token, el paquete 6 sale a través de él.

En este contexto, si se tiene una ráfaga de longitud S segundos, un buffer con una capacidad de C bytes, una tasa de llegada de tokens de ρ bytes/seg, y una tasa máxima de salida de M bytes/seg, se puede calcular el número máximo de bytes en una ráfaga de salida mediante la Ec. (1),

$$C + \rho S = MS \tag{1}$$

Donde MS representa la cantidad de bytes en una ráfaga de longitud S segundos a velocidad máxima [37].

Precisamente, en [38] se investiga cómo obtener los parámetros de este algoritmo a partir de los patrones de tráfico observados en los flujos de una red de computadores en dos casos. En el primero de ellos, el flujo es entregado inmediatamente sin incurrir en retrasos o pérdidas. En el segundo, se añade una cola para almacenar temporalmente los paquetes que no pueden ser entregados de inmediato debido a la ausencia de tokens en el buffer. En este último caso, la cola suaviza el tráfico y los parámetros del algoritmo resultan menos exigentes. Además, se calcula el retardo que presenta cada paquete en la cola y es utilizado para ajustar el tamaño de la misma. Otra manera de obtener los parámetros fundamentales del algoritmo Token Bucket, como se expone en [39], consiste en la formulación de un problema de optimización con el fin de minimizar el retardo para una fuente de tráfico particular.

3.1 Variaciones del algoritmo Token Bucket

Siguiendo los principios básicos del algoritmo Token Bucket, se propone en [16] el algoritmo DTB (Dynamic Token Bucket), el cual asigna un buffer de tokens para cada flujo activo. La capacidad de este buffer es igual a la tasa instantánea F(t), que depende de dos parámetros: el número de flujos activos N(t) y la capacidad del enlace,

$$F(t) = \frac{C}{N(t)} \tag{2}$$

A partir de la Ec.(2) es posible visualizar, que si se asignan pesos a cada uno de los flujos de una red, se puede implementar un modelo de servicios diferenciados. De esta manera, el algoritmo DTB permite que los flujos puedan ser

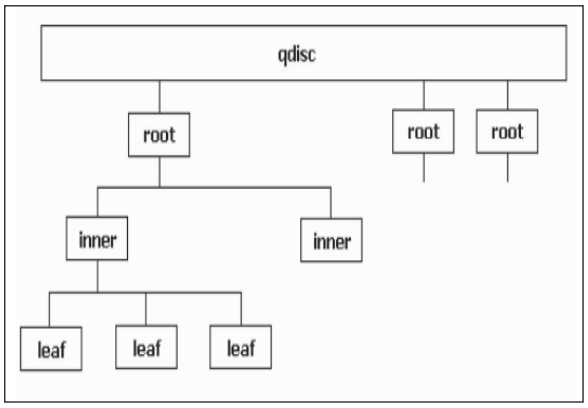

Fig. 5. Jerarquía de clases en HTB.

pre-configurados para recibir ancho de banda diferenciado, contrario a otros mecanismos que buscan una asignación equitativa de ancho de banda. Sin embargo, este algoritmo no es muy eficaz en la asignación equitativa del ancho de banda remanente.

Por otra parte, se encuentra el algoritmo HTB (Hierarchical Token Bucket), el cual implementa una disciplina de colas basada en clases (qdisc) para distribuir equitativamente el ancho de banda, bajo el sistema operativo Linux [40]. Así pues, en HTB se distinguen tres tipos de clases: root, inner y leaf. Las clases root constituyen la parte superior de la jerarquía y todo el tráfico pasa a través de ellas. Las clases inner tienen clases padre e hijas, y las clases leaf son clases terminales que tienen clases padre pero no clases hijas [41], tal como se ilustra en la Fig. 5.

Bajo este esquema, el tráfico proveniente de las clases superiores es clasificado a través de filtros para luego ser inyectado en las clases leaf. Esta clasificación puede hacerse por tipo de servicio, dirección IP e incluso por dirección de red, haciendo fácilmente diferenciables tanto tipos de tráfico como prioridades, para darle a cada uno el tratamiento adecuado. Posteriormente, se programa el tráfico clasificado. Para ello HTB ajusta el throughput empleando el principio básico del algoritmo Token Bucket.

En otro aspecto, la relación entre el algoritmo Token Bucket y el modelo estadístico de dependencia de rango largo (LRD long-range dependence) es ampliamente estudiada en [42] y en [43] se emplea como base para este mismo estudio el proceso estocástico movimiento browniano fraccional (FBM –Fractional Brownian Motion–).

4. EL ALGORITMO VIRTUAL SCHEDULING

El algoritmo Virtual Scheduling (VS) gestiona el tráfico en redes ATM monitoreando la tasa de llegadas de las celdas. Cuando una celda llega, el algoritmo calcula el tiempo teórico de llegada (en inglés, TAT) de la próxima celda [30], de acuerdo con la Ec. (3):

$$TAT = \frac{1}{\lambda_a} \tag{3}$$

Donde λ_a: es la tasa promedio esperada. TAT es medido para encontrar la diferencia en tiempos de llegadas de los encabezados de dos celdas consecutivas.

Por tanto, asumiendo que la diferencia de tiempo entre la celda actual y la próxima celda es *t*, la celda es considerada como conforme si *t* satisface la siguiente desigualdad Ec. (4)

$$t \geq TAT - \Delta \tag{4}$$

Donde Δ es un valor pequeño de tiempo para permitir las pequeñas variaciones en la tasa de datos. La celda se considera como no conforme, cuando el tiempo de llegada de celdas satisface la desigualdad Ec.(5)

$$t < TAT - \Delta \tag{5}$$

En la Fig. 6 se puede observar los diferentes casos de llegadas de celdas en VS. En Fig. 6(a) se muestra una celda conforme debido a que el tiempo satisface la desigualdad Ec. (4). La Fig. 6(b) muestra otro caso de celda conforme, debido a que el tiem-

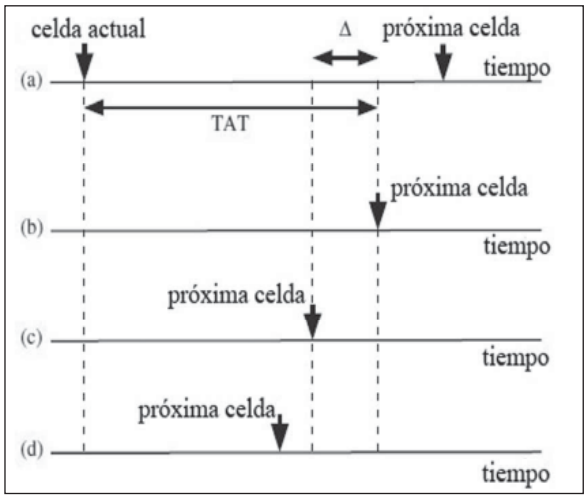

Fig. 6. Casos de llegadas de celdas en el algoritmo VS. (a) $t > TAT$ y la celda es conforme; (b) $t = TAT$ y la celda es conforme; y (c) $t = TAT - \Delta$ y la celda es conforme; (d) $t < TAT - \Delta$ y la celda no es conforme (tomado de [30]).

po de llegada sigue satisfaciendo la desigualdad Ec. (4). La Fig. 6(c) también muestra una celda conforme, ya que el tiempo de llegada satisface la igualdad en Ec. (4). La Fig. 6(d) muestra una celda no conforme debido a que el tiempo de llegada no satisface la desigualdad Ec. (4).

5. COMPARACIÓN ENTRE LOS ALGORITMOS LEAKY BUCKET, TOKEN BUCKET Y VIRTUAL SCHEDULING

Los algoritmos Leaky Bucket, Token Bucket y Virtual Scheduling (VS) tienen un objetivo común: regular la tasa media y la variabilidad del tráfico de entrada a la red. Sin embargo, existen entre ellos diferencias fundamentales que los hacen más o menos viables para una aplicación determinada de acuerdo con las condiciones del tráfico que se quiere controlar. Estas diferencias se sintetizan en la tabla 1.

Como se describió en la sección II, el algoritmo Leaky Bucket impone una tasa promedio de salida fija, sin tener en cuenta el número de ráfagas presentes en el tráfico. Sin embargo, son muchas las aplicaciones en las cuales es deseable incrementar la tasa de salida cuando se detecta la llegada de ráfagas repentinas. En estos casos es ideal la implementación del algoritmo Token Bucket, que presenta una ventaja adicional cuando se llena el buffer, ya que se descartan los tokens pero no los paquetes, contrario al comportamiento del algoritmo Leaky Bucket en el que se descartan los paquetes tan pronto como se llena el buffer.

Tabla 1 Diferencias entre los algoritmos de gestión de tráfico.

	Leaky Bucket	**Token Bucket**	**Virtual Scheduling (VS)**
Parámetros fundamentales	• Tasa de datos λ_a • Número de paquetes N	• Tasa de generación de tokens λ • Tamaño del búfer C	• Tasa promedio esperada λ_a • Valor de tiempo reducido Δ para permitir pequeñas variaciones en la tasa de datos. • Tiempo de llegada teórico TAT
Tipo de tráfico de salida	Tasa Fija	Tasa Flexible	Conforme o no Conforme
Comportamiento cuando se llena el buffer	Descarta paquetes	Descarta tokens pero no paquetes	N/A
Aplicable en tiempo real	No	Sí	Sí

Por otra parte, la gran utilidad del algoritmo Virtual Scheduling radica en la posibilidad de generar una implementación de manera más sencilla y directa, que incluso con el mismo algoritmo Leaky Bucket.

6. APLICACIONES

Son muchas las aplicaciones obtenidas a partir de estos algoritmos. En [44] por ejemplo, se introduce el concepto de Shaped Variable Bit Rate (SVBR). SVBR es un control de tráfico preventivo que permite la codificación del tráfico de video directamente en la red, regulando el tráfico impredecible de ráfagas grandes por medio del algoritmo Leaky Bucket. Inspirados en este trabajo, en [45] se desarrolla el sistema Evalvid-RASV. Este sistema trabaja en el concepto VBR (lazo abierto de codificación de vídeo), pero está diseñado para que no se produzcan ráfagas de tráfico sin compromisos, sin retrasos adicionales.

En [46] se adopta el algoritmo Token Bucket para controlar el volúmen de tráfico en una red WiMAX. Para ello, se ajustan parámetros tales como la tasa de tokens y el tamaño del buffer de acuerdo con las características de cada clase de tráfico. Los resultados de simulación muestran una mejora considerable en el desempeño de la red luego de aplicar esta técnica, ya que disminuye el retardo promedio para el tráfico en tiempo real como RTPS (Real Time Polling Service). Asimismo se reduce la probabilidad de pérdida de datos para tráfico en tiempo real y para servicios en tiempo no real como NRTPS (Non-Real-Time Polling Service), diseñado para soportar el flujo de servicios que requieren normalmente ráfagas de datos de tamaño variable, como es el caso del gran ancho de banda que maneja FTP.

Bajo este mismo enfoque, se propone en [47] un programador de paquetes cuya arquitectura está basada en un diseño cross-layer para redes móviles WiMAX, el cual combina el concepto de función de utilidad (empleado en economía), para satisfacer el retardo máximo de paquetes de los flujos de los servicios en tiempo real y el algoritmo Token Bucket para soportar los requerimientos mínimos de desempeño de los flujos de los servicios en tiempo no real.

En [48] se incorpora un conformador de tráfico Token Bucket para proveer calidad de servicio en una LAN wireless IEEE802.11 bajo un ambiente compuesto por 11 estaciones móviles, solución que puede ser implementada en un entorno real, ya que no modifica la capa MAC 802.11. En contraste en [41] se expone un mecanismo para proveer calidad de servicio en WLAN, pero en lugar de enfocarse en la capa MAC, la solución está orientada a la capa IP con HTB para controlar la manera como se entregan los paquetes a la capa MAC. También basado en HTB, se define en [49] un programador denominado TWHTB (The Wireless Hierarchical Token Bucket), capaz de diferenciar del servicio de transporte ofrecido por un punto de acceso IEEE 802.11 teniendo en cuenta la calidad del enlace de radio experimentado por las diferentes estaciones móviles.

Otras áreas como la inteligencia computacional, también se han integrado a este campo. Es así como en [50] se emplea lógica difusa para controlar la tasa de generación de tokens en un esquema Token Bucket, aplicado a una red ATM de alta velocidad. De esta manera se obtiene un menor retardo, se reducen las pérdidas y se incrementa el rendimiento del sistema. Asimismo, en [51] se desarrolla un predictor Token Bucket lógico difuso que es aplicado a dos arquitecturas promisorias como lo son los servicios diferenciados (DiffServ) y MPLS (Multiprotocol Label Switching), de manera que el requerimiento de ancho de banda real se puede predecir y la realimentación es transmitida a los mecanismos de control de admisión.

En cuanto al algoritmo Virtual Scheduling se refiere, en [27] se proponen dos protocolos inde-

pendientes para regulación de tráfico, estos son LGBP (Loose Generic Bucket Policer) y SGBP (Strict Generic Bucket Policer), ambos basados en el algoritmo GCRA (Generic Cell Rate Algorithm), cuya primera versión fue VS [52].

7. CONCLUSIONES

El algoritmo Leaky Bucket original presenta algunas desventajas como la baja eficiencia y difícil aplicación en tiempo real para fuentes de tráfico con ráfagas. Por tanto, en la literatura se encuentran mejoras al algoritmo, algunas de ellas basadas en mecanismos de inteligencia computacional que permiten hacer predicción de tráfico.

El algoritmo Token Bucket permite acumular tokens hasta un tamaño máximo de buffer n, lo cual significa que se pueden enviar a la vez ráfagas de máximo n paquetes. De esta manera se obtiene a la salida una tasa variable con la que se puede responder de forma rápida a ráfagas de entrada inesperadas.

Aunque el intervalo máximo de ráfaga en el algoritmo Token Bucket puede regularse mediante una selección cuidadosa del parámetro ρ, pueden presentarse ráfagas de gran extensión. Una opción para equilibrar estos intervalos, consiste en implementar un algoritmo Leaky Bucket con una tasa mayor que ρ pero menor que la tasa máxima de la red, a la salida de un algoritmo Token Bucket, obteniendo de esta manera una tasa de tráfico uniforme.

En general, los tres algoritmos presentados se emplean normalmente para llevar a cabo las funciones de control y conformación de tráfico en los switches ATM, y son ampliamente utilizados en diferentes campos, incluso han sido complementados con técnicas de inteligencia computacional incrementando su eficiencia y las prestaciones que brindan a las redes de comunicaciones.

REFERENCIAS

[1] J. Tunner, "New directions in communications (or which way to the information age)," *IEEE Comunication Magazine*, vol. 24, no. 10, pp. 17 – 24, Oct. 1986.

[2] P. Boyer, F. Guillemin, M. Servel and M. Coudreuse, "Spacing cells protects and enhances utilization of atm network links," *IEEE Network*, vol. 6, no. 5, pp. 38 – 49, 1992.

[3] J. Man-Yeong, K. Dong-Yong and P. Hong-Shik, "Implementation of a peak cell rate policer using the virtual scheduling algorithm," *IEEE International Conference on Communications, Conference Record, Converging Technologies for Tomorrow's Applications*, vol. 2, pp. 762 – 766, 1996.

[4] P. Gallay, J. Majos and M. Servel, "A 622 mb/s 256 k atm resource management circuit," *IEEE International Solid-state Circuits Conference*, San Francisco. Feb. 1999.

[5] M. Norashidah and N. Fisal, "Fuzzy logic token bucket bandwidth predictor for assured forwarding traffic in a diffserv-aware mpls internet", *Proceedings of the First Asia International Conference on Modelling & Simulation*. Washington. Mar. 2007.

[6] P. Giacomazzi, L. Musumeci, G. Saddemi and G. Verticale, "Optimal selection of token bucket parameters for the admission of aggregate flows in ip networks," *IEEE*

Global Telecommunications Conference. GLOBECOM. San Francisco. Dec. 2006.

[7] R. Haalen and R. Malhotra, "Improving TCP performance with bufferless token bucket policing: A TCP friendly policer," *15th IEEE Workshop on Local & Metropolitan Area Networks. LANMAN*, Princeton, June. 2007.

[8] C. Lee, J. Kwak and D. Jeong, "Enhanced token bucket policer using reduced fair queuing for ethernet access networks," *IET Communications*, vol. 1, no. 6, pp. 1248 – 1255, Dec. 2007.

[9] M. Butto, E. Cavallero and A. Tonietti, "Effectiveness of the leaky bucket policing mechanism in atm networks," *IEEE Journal on selected areas in communications*, vol. 9, no. 3, pp. 335 – 342, Apr. 1991.

[10] J. Lee and C. Un, "Performance of dynamic rate leaky bucket algorithm," *Electronics Letters IEEE*, vol. 29, no. 17, pp. 1560 – 1561, Aug. 1993.

[11] P. Giacomazzi, L. Musumeci, G. Saddemi and G. Verticale, "Analytical methods for resource allocation and admission control with dual-leaky bucket regulated traffic," *IEEE International Conference on Communications*, Glasgow, Aug 2007.

[12] L. Sheng and S. Wen, "The token-bank leaky bucket mechanism for group connections in atm networks," *Proceedings International Conference on Network Protocols*, Washington, Oct. 1996.

[13] S. Heckmuller and B. Wolfinger, "Analytical modeling of token bucket based load transformations," *International Symposium on Performance Evaluation of Computer and Telecommunication Systems. SPECTS*, Edinburgh, June, 2008.

[14] B. Kovacs, "Mathematical remarks on token bucket," *17th International Conference on Software, Telecommunications & Computer Networks. SoftCOM*, Hvar, Sept. 2009.

[15] E. Park and C. Choi, "Adaptive token bucket algorithm for fair bandwidth allocation in DiffServ networks," *IEEE Global Telecommunications Conference. GLOBECOM*, vol. 6, pp. 3176 – 3180, Dec. 2003.

[16] J. Kidambi, D. Ghosal and B. Mukherjee, "Dynamic token bucket (DTB): a fair bandwidth allocation algorithm for high-speed networks", *Computer Communications and Networks, 1999. Proceedings. Eight International Conference on*, Boston, Oct. 1999.

[17] H. Kazemian and L. Meng, "Neuro-fuzzy control for mpeg video transmission over bluetooth", *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, pp. 761 – 771, Oct. 2006.

[18] D. Perez and J. Valenzuela, "Performance of a token bucket traffic shaper on a real IEEE 802.11 test-bed," *IEEE 16th International Symposium on Personal, Indoor and Mobile Radio Communications. PIMRC*, Berlin, Sept. 2005.

[19] J. Ben-Othman, L. Mokdad, "Improving QoS for UGS, rtPS, nrtPS, BE in WIMAX networks," *International Conference on Communications and Information Technology (ICCIT)*, Aqaba, March 2011.

[20] I. Liu, F. Takawira and H. Xu, "A hybrid token-cdma mac protocol for wireless ad

hoc networks," *IEEE Transactions on Mobile Computing*, vol. 7, no. 5, pp. 557 – 569, May. 2008.

[21] T. Tsai and C. Wang, "Routing and admission control in ieee 802.16 distributed mesh networks," *International Conference on Wireless and Optical Communications Network. WOCN*, Singapore, Aug. 2007.

[22] C. Li and P. Wai, "A token bucket method for packet injection control in deflection-routed optical networks," *15th Opto Electronics and Communications Conference (OECC2010) Technical Digest*, Sapporo, July. 2010.

[23] E. Gunduzhan, "An elastic traffic policer for MPEG video streams," *IEEE International Conference on Multimedia and Expo. ICME*, Amsterdam, July. 2005.

[24] J. Pitts and J. Schormans, "An analytical study of precedence and QoE for voice service in degraded IP networks," *IEEE Military Communications Conference, MILCOM*, San Diego, Nov. 2008.

[25] D. Balan and D. Potorac, "Linux htb queuing discipline implementations," *First International Conference on Networked Digital Technologies. NDT*, Ostrava, July 2009.

[26] M. Feng, H. Zhang, D. Wang and Z. Sun, "Design and implementation of high-speed token bucket based on FPGA," *2nd International Conference on Information Science and Engineering (ICISE)*, Dec. 2010.

[27] C. Hu, H. Saidi, P. Yan and P. Min, "A protocol independent policer and shaper design using virtual scheduling algorithm," in *IEEE 2002 International Conference on Communications, Circuits and Systems and West Sino Expositions*, IEEE, vol. 1, pp. 791– 795, Jul. 2002.

[28] J. Park, K. Ha, S. Lee and K. Lee, "Performance evaluation of NDIS-based four-layer architecture with virtual scheduling algorithm for IEEE 802.11b," *Control, Automation and Systems, 2007. ICCAS '07. International Conference on*, Seoul, Oct. 2007.

[29] S. Lee, J. Park, K. Ha and K. Lee, "Wireless networked control system using ndis-based four-layer architecture for IEEE 802.11b," *Factory Communication Systems, 2008. WFCS 2008. IEEE International Workshop on*, Dresden, May. 2008.

[30] F. Gebali, *Analysis of Computer and Communication Networks*. Victoria: Springer, 2008.

[31] L. Du-Hern, S. Yoan and K. Young-Han, "A variable rate leaky bucket algorithm based on a neural network prediction in ATM networks," *Proceedings of International Conference on Information, Communications and Signal Processing, ICICS*, Sept. 1997.

[32] C. Chung-Ju, Y. Chung-Hsun, C. Chih-Sheng and L. Li-Fong, "Intelligent leaky bucket algorithms for sustainable-cell-rate usage parameter control in atm networks," *IEEE Transaction On Multimedia*, vol. 6, no. 5, pp. 749 – 759, Oct. 2004.

[33] L. Somchai and C. F. Chun, "An adaptive fuzzy control traffic shaping scheme over wireless networks," *Proceedings of IEEE Asia-Pacific Conference on Communications 2007*, Bangkok, Oct. 2007.

[34] T. Takase, N. Komuro, S. Sakata, S. Shioda and T. Murase, "QoS control for wire-

less LAN using receiving opportunity control based on token bucket filter," in *2011 IEEE Consumer Communications and Networking Conference (CCNC)*, Las Vegas, NV, USA, Las Vegas, Jan. 2011, [En línea]. Disponible en: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5766658

[35] S. Park, J. Oh, K. Kim and J. Jang, "Reconfigurable bandwidth controller for responding the DDoS attacks using token bucket mechanism," in *The 7th International Conference on Advanced Communication Technology, 2005, ICACT 2005.* Phoenix Park, 2005. [En línea]. Disponible en: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1461820

[36] M. Brown. *Traffic control HOWTO*. Oct. 2006. [En linea]. Disponible en: http://tldp.org/HOWTO/Traffic-Control-HOWTO/classless-qdiscs.html.

[37] A. Tanenbaum, *Computer networks* 4th ed., Amsterdam: Prentice Hall, 2003.

[38] P. P. Tang, T. Y. Tai, "Network traffic characterization using token bucket model," in *IEEE INFOCOM '99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings*, New York, Mar. 1999.

[39] D. Niyato, J. Diamond and E. Hossain, "On optimizing token bucket parameters at the network edge under generalized processor sharing (GPS) scheduling," in *IEEE Global Telecommunications Conference, 2005. GLOBECOM '05*, Dec. 2005.

[40] D. Ivancic, N. Hadjina and D. Basch, "Analysis of precision of the HTB packet scheduler," in *18th International Conference on Applied Electromagnetics and Communications, 2005. ICECom 2005*. Dubrovnik, Oct. 2005.

[41] J. Valenzuela, A. Monleon, I. San Esteban, M. Portoles and O. Sallent, "A hierarchical token bucket algorithm to enhance QoS in IEEE 802.11: proposal, implementation and evaluation," *Vehicular Technology Conference, 2004. VTC2004-Fall. 2004 IEEE 60th*, vol. 4, Sept. 2004.

[42] S. Bregni, R. Cioffi and P. Giacomazzi, "Queuing performance of Long-Range dependent traffic regulated by Token-Bucket policers," in *IEEE GLOBECOM 2008 - 2008 IEEE Global Telecommunications Conference*, New Orleans, Nov. 2008. [En línea]. Disponible en: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4698039

[43] G. Procissi, A. Garg, M. Gerla and M. Sanadidi, "Token bucket characterization of long-range dependent traffic," *Computer Communications*, vol. 25, no. 11-12, pp. 1009–1017, July. 2002. [En línea]. Disponible en: http://www.sciencedirect.com/science/article/pii/S0140366402000154

[44] H. Hamdi, J. W. Roberts and P. Rolin, "Rate control for VBR video coders in broad-band networks," *Selected Areas in Communications, IEEE*, vol. 15, no. 6, pp. 1040–1051, Aug. 1997.

[45] A. Suki, M. Arif, S. Hassan, O. Ghazali and S. AwangNor, "Evalvid-RASV: Shaped VBR rate adaptation stored video system," *2nd IEEE International Conference on Education Technology and Computer (ICETC)*, Shanghai, June 2010.

[46] S. Ghazal and J. Ben-Othman, "Traffic policing based on token bucket mechanism for WiMAX networks," *Communications*

(ICC), 2010 IEEE International Conference on, Cape Town, May. 2010.

[47] A. Nascimento, A. Gameiro and J. Rodríguez, "A joint utililitytoken bucket packet scheduling algorithm for IEEE 802.16e WiMAX networks," *Wireless Communication Systems, 2009. ISWCS 2009. 6th International Symposium on*, Tuscany, Sept. 2009.

[48] D. Perez and J. Valenzuela, "Performance of a token bucket traffic shaper on a real IEEE 802.11 test-bed," in *2005 IEEE 16th International Symposium on Personal, Indoor and Mobile Radio Communications*, Berlin, Sept. 2005. [En línea]. Disponible en: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1651777

[49] R. Garroppo, S. Giordano, S. Lucetti and E. Valori, "The wireless hierarchical token bucket: A channel aware scheduler for 802.11 networks," in *Sixth IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks*, University of Pisa June 2005. [En línea]. Disponible en: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1443505

[50] A. Aeron, "Fine tuning of fuzzy token bucket scheme for congestion control in high speed networks," *Computer Engineering and Applications (ICCEA), 2010 Second International Conference on*, vol. 1, pp. 170–174, Mar. 2010.

[51] N. Din and N. Fisal, "Fuzzy logic token bucket bandwidth predictor for assured forwarding traffic in a DiffServ-Aware MPLS internet," in *Proceedings of the First Asia International Conference on Modelling& Simulation*. IEEE Computer Society, Asia. Mar. 2007.

[52] P. Castillo, A. Soto, F. Flor, *Codificación y transmisión robusta de señales de video MEPG-2 de caudal variable sobre redes de transmisión asíncrona ATM*. España: Univ. de Castilla-La Mancha, 1999.