Conformando trafico

Pablo Collado Soto

Ingeniería de Tráfico

1. Introducción

En esta práctica comenzaremos por estudiar el funcionamiento de un clasificador *WFQ* de manera práctica a través de la simulación de un escenario análogo al de la práctica anterior. Más tarde analizaremos un clasificador *WFQ* de manera teórica para terminar con el estudio de un conformador de tráfico según un cubo con goteo así como una función policía basada en el mismo algoritmo.

2. Estudiando un clasificador *WFQ* (2.1.1)

Antes de nada definiremos los flujos de nuestro escenario en la tabla 1. Debemos tener en cuenta que la velocidad de la interfaz de salida de los flujos de datos es $V_l = 10\ Mbps$ para toda la prueba. La configuración del router para este escenario se puede encontrar aquí.

2.1. Comentando las figuras

Observando la tasa de cada flujo en la figura 1 vemos que, en efecto, se garantiza que el `flujo 2` mantiene la tasa durante la prueba. Tengamos en cuenta que el flujo tiene una tasa de $1,5\ Mbps$ mientras que a este flujo se le otorgan $5\ Mbps$ al tener un peso del 50 % de la velocidad de la interfaz. Concluimos pues que las tasas son congruentes con lo esperado.

Analizando la latencia que aparece en la figura 2 veremos que a medida que se añaden más flujos a la prueba se empiezan a saturar los buffers asociados a cada flujo **exceptuando** el `flujo 2` ya que éste nunca superará la tasa que tiene asociada. En los instantes iniciales de la prueba solo contamos con los flujos `1 y 2` cuyo total agregado es de $9,5\ Mbps < V_l = 10\ Mbps$ con lo que se puede acomodar todo el tráfico al repartirse la capacidad no asociada a otros flujos con lo que la latencia es prácticamente $0\ s$. Cuando los siguientes flujos entran en acción su capacidad asociada no puede ser empleada para el primer flujo con lo que la latencia comienza a dispararse. Llegará un momento en el que todos los flujos excepto el `2` saturen su cola asociada que ocurre cuando llegan a una línea horizontal. Esta latencia máxima de cola es $85\ ms$ y $400\ ms$ para los flujos `1` y `3, 4 y 5`, respectivamente.

3. Garantizando una latencia mínima con *WFQ* (2.1.2)

Empleando los mismos flujos que en el apartado anterior vamos a priorizar uno de los flujos de manera que solo se saquen paquetes de otras colas si y solo si esta cola prioritaria está vacía. Como esto podría provocar inanición, la implementación práctica descartará el tráfico prioritario en exceso (según el porcentaje configurado) a no ser que las demás colas estén vacías. Ésto se traducirá en que

	Puerto	Comienzo [s]	Fin [s]	Tasa [Mbps]	Paquetes por segundo	SDU de UDP [B]
1	51151	0	20	8	800	1250
2	51152	5	20	1,5	150	1250
3	51153	10	20	4	400	1250
4	51154	15	20	2,4	240	1250
5	51155	15	20	1,6	160	1250

Tabla 1: Configuración de los flujos para estudiar *WFQ*.

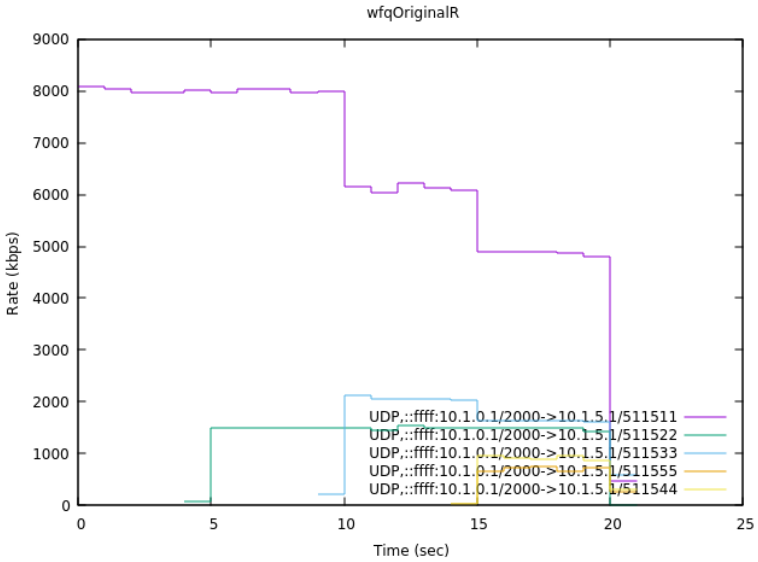

Figura 1: Tasa de cada flujo.

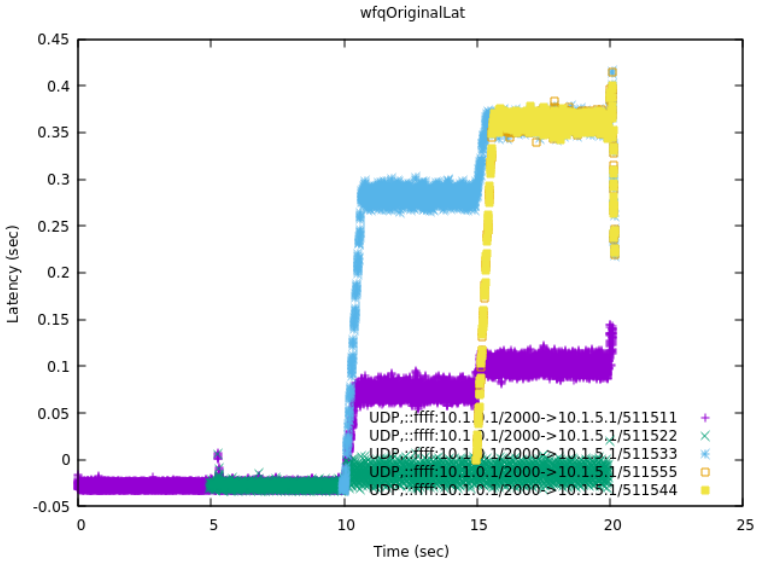

Figura 2: Latencia de cada flujo.

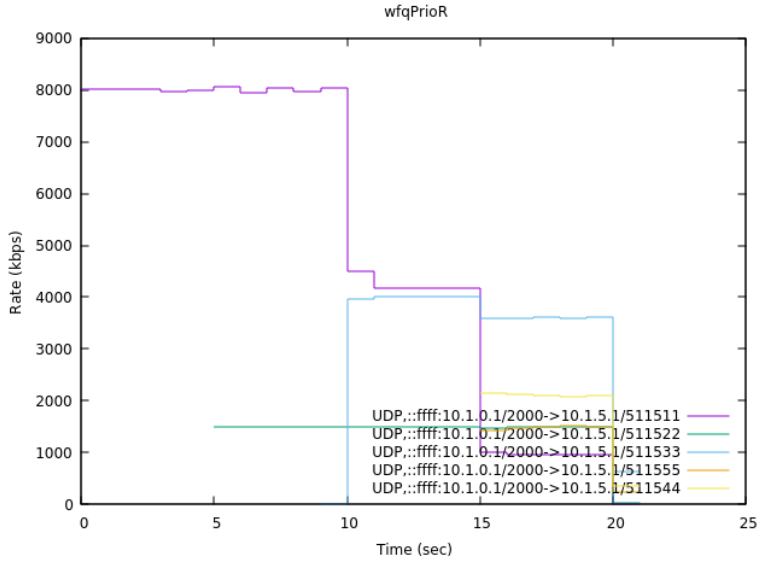

Figura 3: Tasa de cada flujo.

se mantendrá una latencia acotada pero se incrementarán las pérdidas del flujo prioritario a medida que se añadan más flujos a la prueba. La configuración del router para este caso se puede encontrar aquí, siendo la única diferencia con la anterior la línea `58`.

3.1. Comentando las figuras

Tal y como comentábamos al explicar la implementación de las prioridades observamos cómo la tasa del flujo prioritario (el `1`) empezará en los 8 *Mbps* iniciales hasta que se incorporan más flujos. Al hacerlo, la tasa se irá reduciendo hasta que en los últimos 5 *segundos* se aproxima a 1 *Mbps*, el 10 % de la velocidad del interfaz que configuramos. Todo esto se observa en la figura 3.

Si observamos la latencia en la figura 4 nos daremos cuenta de que, como antes, la latencia del flujo `2` se mantiene constante al tener asignado suficiente ancho de banda y que **además** la latencia del flujo `1` también está acotada al ser el flujo prioritario. Las demás tasas se comportan como comentábamos antes.

Si acabamos por observar las pérdidas en la figura 5 nos daremos cuenta de que las pérdidas de todos los flujos excepto el `2` experimentarán pérdidas hacia el final de la prueba. Si nos centramos en el flujo `1`, el prioritario, veremos que estas pérdidas comienzan a dispararse en cuanto el tercer flujo entra en acción y debemos empezar a descartar paquetes. Es interesante ver que el aumento de las pérdidas del flujo `1` se produce en el mismo instante en el que su tasa se disminuye.

4. Garantizando una latencia mínima con *WFQ* y vigilancia (2.1.3)

Empleando los mismos flujos que en los apartados anteriores vamos a añadir una "capa" de *vigilancia* que se encargará de que el tráfico sobrante del flujo prioritario sea descartado sin importar si las demás colas están vacías o no. Ésto nos podría evitar problemas con el tráfico extra ya que éste entraría en colas que estuvieran vacías en ese momento... Para lograrlo configuraremos una función *policía* en la *clase* asociada al flujo `1`. Que rechazará el tráfico sobrante por encima de la tasa asignada a dicha *clase*. Como hemos asignado al flujo prioritario un 10 % de la capacidad de la interfaz ($V_l = 10$ *Mbps*) éste contará con una tasa de 1 *Mbps*. Ésta tasa será la que utilizaremos como límite en la función *policía* que hemos de configurar. Dicha configuración se puede encontrar en la línea `64` de éste archivo.

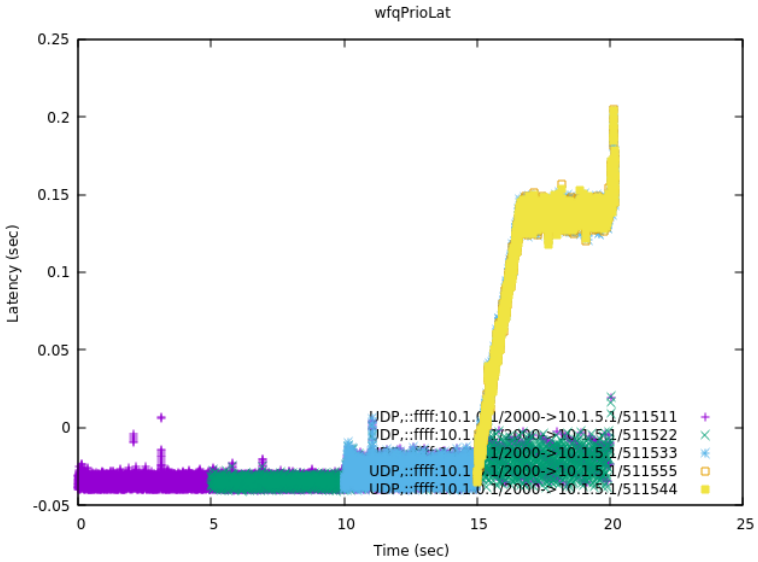

Figura 4: Latencia de cada flujo.

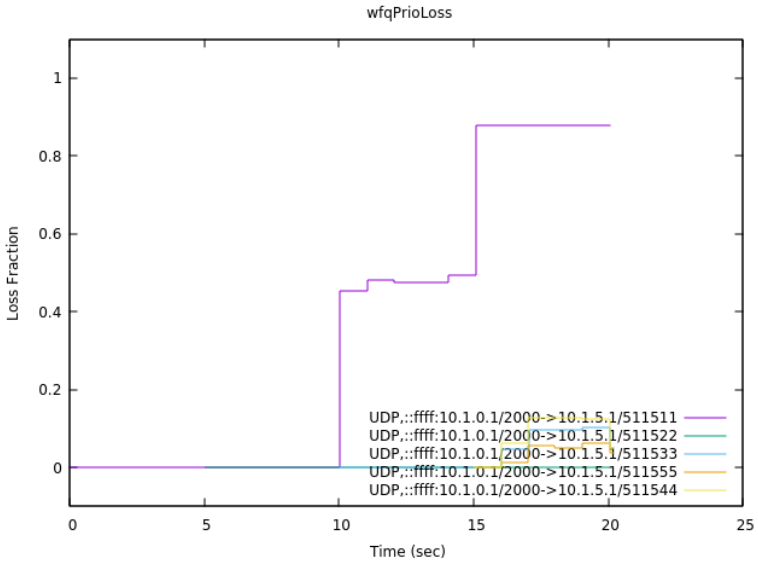

Figura 5: Pérdidas de cada flujo.

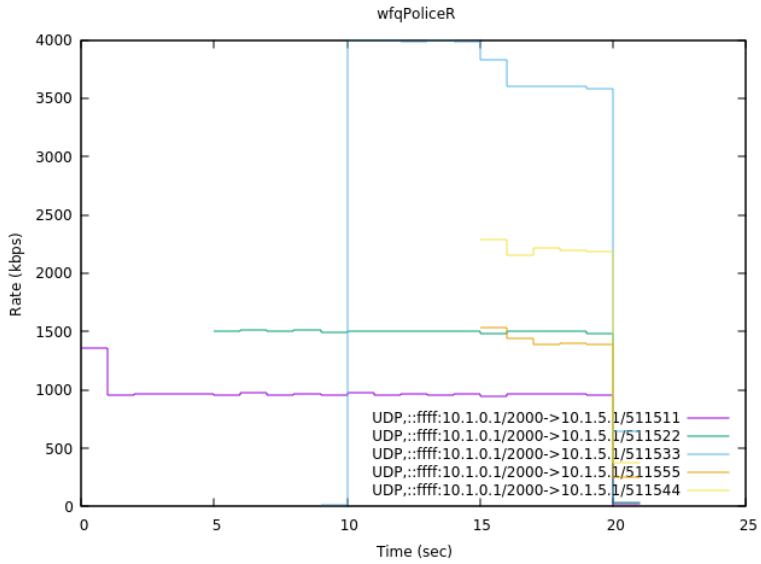

Figura 6: Tasa de cada flujo.

4.1. Comentando las figuras

Observando la figura 6 nos damos cuenta de que, a diferencia del caso anterior, la tasa del flujo prioritario se limita prácticamente desde el inicio de la prueba. Al no haber especificado el valor de `BE`, la cantidad de datos que pueden rebasar la `CIR` momentáneamente (el tamaño de un cubo de créditos asociado a la función de *policiía*), creemos que un valor por defecto distinto de 0 puede ser la causa del pico inicial superior a 1 $Mbps$. En cualquier caso, podemos considerar que la tasa está limitada a 1 $Mbps$ durante la duración de la prueba, cosa que diferencia este caso y el anterior totalmente.

Si ahora estudiamos la figura 7 veremos que es prácticamente idéntica a la del caso anterior. La única diferencia con el escenario sin función de *policía* es que limitamos la tasa del flujo prioritario desde el principio. No obstante, como el flujo prioritario era "desalojado" por los demás hasta que la tasa asociada era la mínima garantizada (el 10 %) no observamos diferencias en la latencia en ningún caso.

Finalmente podemos analizar las pérdidas de cada flujo en la figura 8. La diferencia con el caso anterior es que la pérdida de paquetes del flujo prioritario es muy elevada desde el primer instante además de ser constante. La "simetría" con la tasa del mismo flujo se aprecia de nuevo al igual que en el caso anterior: como empezamos a descartar tráfico desde prácticamente el principio la tasa es elevada desde ese mismo instante. La relación que existía en el caso anterior se mantiene pero es diferente a este caso porque la evolución de la tasa del flujo prioritario es también diferente al haber aplicado al función *policía*.

5. Cálculos teóricos sobre un planificador

En esta sección resolvemos el ejercicio con resultados propuesto en el guión. En nuestro caso contamos con un enlace de velocidad $V_l = 10\ Mbps$ y 5 $flujos$ de 2, 1, 2, 3 y 4 $Mbps$ cuya capacidad asignada es 15 %, 40 %, 10 %, 10 % y 25 %, respectivamente. Incluimos las tablas 2-4 así como los grupos de ecuaciones 1-12 con cada una de las iteraciones. Los resultados aparecen en la tabla 5.

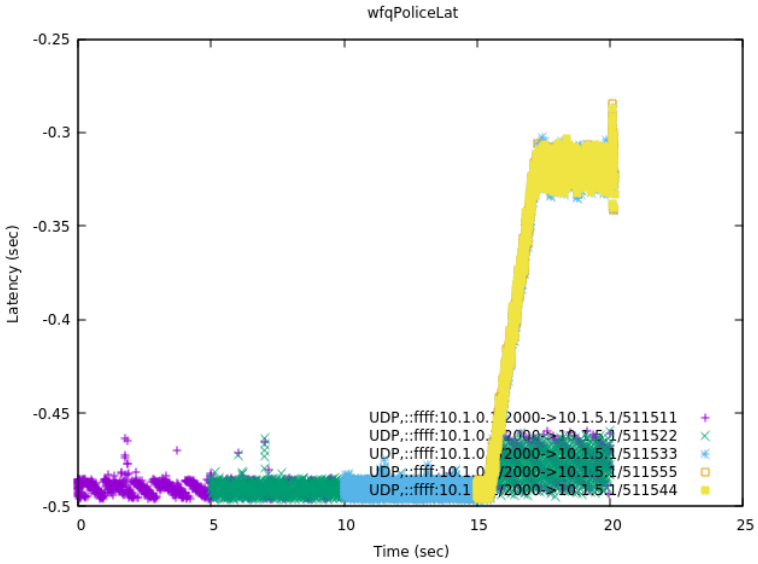

Figura 7: Latencia de cada flujo.

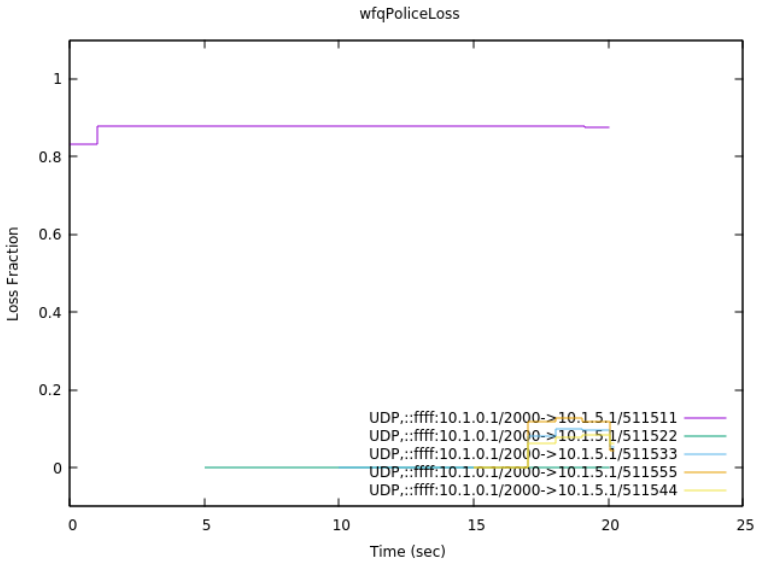

Figura 8: Pérdidas de cada flujo.

Peso (%)	Tráfico Inicial [Mbps]	C. min. g. [Mbps]	C. Asig. [Mbps]	T. Pend. [Mbps]
15	2	1,5	1,5	0,5
40	1	4	1	0
10	2	1	1	1
10	3	1	1	2
25	4	2,5	2,5	1,5

Tabla 2: Primera iteración.

Peso (%)	T. Pend. I. [Mbps]	C. min. g. [Mbps]	C. Asig. [Mbps]	T. Pend. F. [Mbps]
15	0,5	0,75	0,5	0
	0		0	0
10	1	0,5	0,5	0,5
10	2	0,5	0,5	1,5
25	1,5	1,25	1,25	0,25

Tabla 3: Segunda iteración.

Peso (%)	T. Pend. I. [Mbps]	C. min. g. [Mbps]	C. Asig. [Mbps]	T. Pend. F. [Mbps]
	0		0	0
	0		0	0
10	0,5	0,056	0,056	0,444
10	1,5	0,056	0,056	1,444
25	0,25	0,139	0,139	0,111

Tabla 4: Tercera iteración.

$$C_{min_i} = V_l \cdot \frac{w_i}{\sum_{i=1}^{5} w_i} \tag{1}$$

$$C_{min_1} = 10\ Mbps \cdot \frac{15}{100} = 1,5\ Mbps \tag{2}$$

$$C_{min_2} = 10\ Mbps \cdot \frac{40}{100} = 4\ Mbps \tag{3}$$

$$C_{min_3} = 10\ Mbps \cdot \frac{10}{100} = 1\ Mbps \tag{4}$$

$$C_{min_4} = 10\ Mbps \cdot \frac{10}{100} = 1\ Mbps \tag{5}$$

$$C_{min_5} = 10\ Mbps \cdot \frac{25}{100} = 2,5\ Mbps \tag{6}$$

$$C_{sobrante} = 10 - (1,5 + 1 + 1 + 1 + 2,5) = 3\ Mbps \tag{7}$$

$$C_{min_1} = 3\ Mbps \cdot \frac{15}{60} = 0,75\ Mbps \tag{8}$$

$$C_{min_3} = 3\ Mbps \cdot \frac{10}{60} = 0,5\ Mbps \tag{9}$$

$$C_{min_4} = 3\ Mbps \cdot \frac{10}{60} = 0,5\ Mbps \tag{10}$$

$$C_{min_5} = 3\ Mbps \cdot \frac{25}{60} = 1,25\ Mbps \tag{11}$$

$$C_{sobrante} = 3 - (0,5 + 0,5 + 0,5 + 1,25) = 0,25\ Mbps \tag{12}$$

$$C_{min_3} = 0,25\ Mbps \cdot \frac{10}{45} = 0,056\ Mbps \tag{13}$$

$$C_{min_4} = 0,25\ Mbps \cdot \frac{10}{45} = 0,056\ Mbps \tag{14}$$

$$C_{min_5} = 0,25\ Mbps \cdot \frac{25}{45} = 0,139\ Mbps \tag{15}$$

Con la tercera iteración recogida en la tabla 4 terminamos el reparto ya que ninguno de los flujos restantes recibe tanta capacidad como tráfico pendiente. Por lo tanto, el reparto queda:

Flujo	Capacidad asignada [Mbps]
1	$1,5 + 0,5 = 2$
2	1
3	$1 + 0,5 + 0,056 = 1,556$
4	$1 + 0,5 + 0,056 = 1,556$
5	$2,5 + 1,25 + 0,139 = 3,889$

Tabla 5: Resultados tras el reparto.

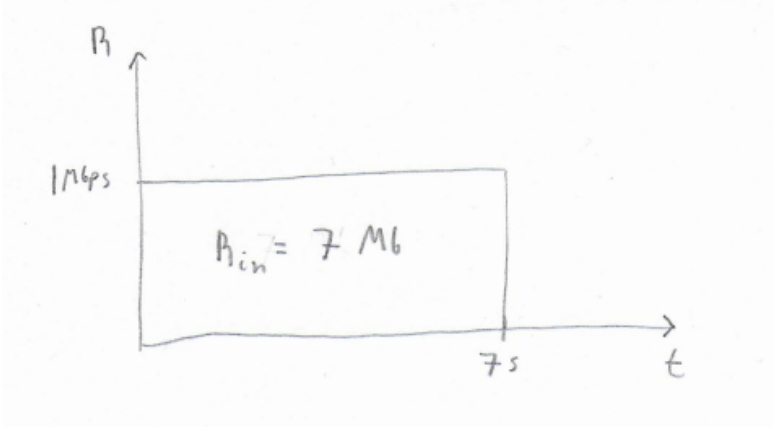

Figura 9: Entrada de ambos sistemas.

6. Cubos con créditos (2.3)

En esta sección estudiamos el conformado de tráfico con un *cubo de créditos* para después aplicar una función *policía*.

6.1. Prueba de conformado (2.3.1)

Incluimos la entrada de la ráfaga descrita en esta sección y la siguiente, así como la salida esperada para el *cubo de créditos* en las figuras 9 y 10, respectivamente.

La configuración del encaminador se incluye en éste archivo. Como la prueba solo contiene un flujo empleamos la clase por defecto del encaminador para facilitar la sintaxis de configuración.

Incluimos la tasa de salida de la ráfaga conformada en la figura 11. Esperamos que la tasa caiga hasta la velocidad media en $t = 1,5$ s pero dado que la granularidad de `trpr` es de 1 s veremos la caída en $t = 2$ s. Asimismo, observamos que se introduce un retardo que "estira" la ráfaga de entrada hasta la marca de los 9 s. Si observamos la gráfica de la latencia en la figura 12 veremos que ésta empieza a crecer una vez que se vacía el cubo de créditos en $t = 1,5$ s. Al igual que antes, no veremos

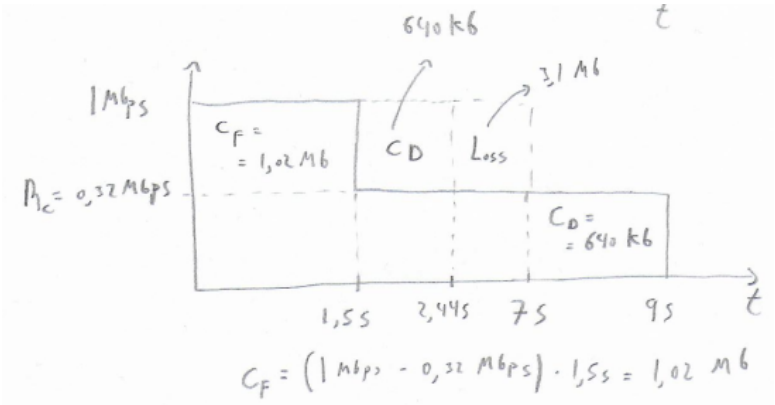

Figura 10: Salida esperada del *cubo de créditos*.

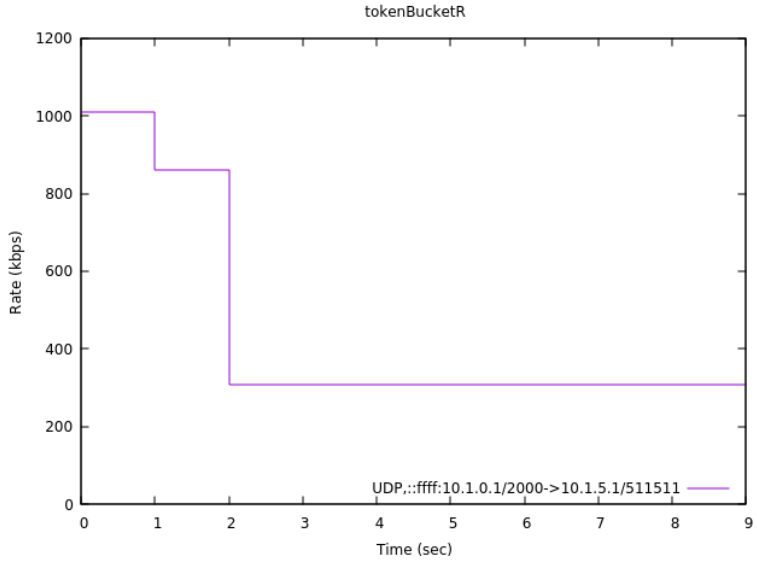

Figura 11: Tasa del flujo a la salida del conformador.

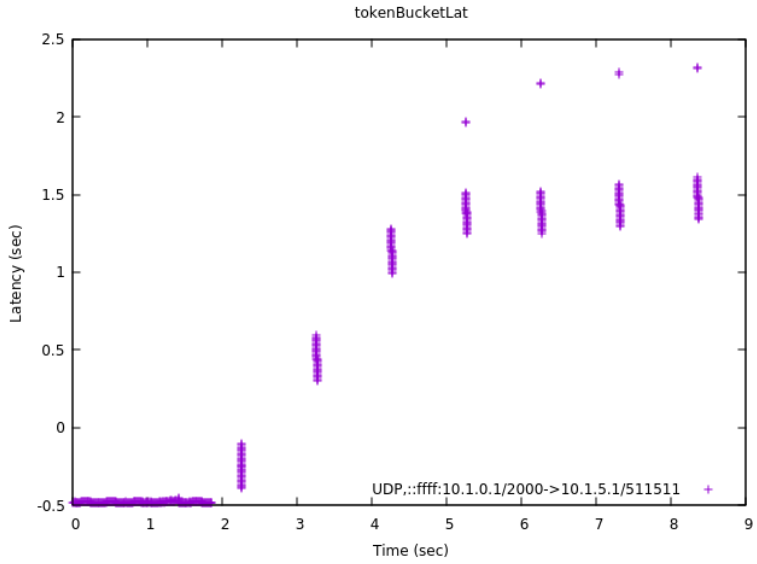

Figura 12: Latencia del flujo a la salida del conformador.

ésto reflejado en la gráfica hasta $t = 2\ s$. Asimismo, vemos que esta latencia se "aplana" en un valor de 2 s, retardo que esperábamos tal y como recoge la teoría. Al comparar este conformado con la sección siguiente observaremos que el hecho de descartar el tráfico en exceso en vez de almacenarlo en un buffer para luego extraerlo incrementará las pérdidas pero la latencia permanecerá constante. Reseñamos que el valor del buffer de datos de entrada real es prácticamente igual a 640 kb ya que los cálculos teóricos y los resultados generados son bastante parecidos. Además, la granularidad tan poco exacta de `trpr` así como la variabilidad intrínseca al proceso de simulación dificultan tremendamente poder hacer cálculos exactos.

6.2. Prueba con función policía (2.3.2)

Al igual que antes incluimos las gráficas con la tasa del flujo tras aplicar la función policía así como la latencia en las figuras 13 y 14, respectivamente. La configuración del encaminador para este apartado se encuentra en éste archivo.

En la figura 13 observamos que la ráfaga no acaba en $t = 9\ s$ sino que lo hace a los 7 segundos ya que no almacenamos una cantidad de datos igual al buffer de datos (C_D) para luego transmitirlos: directamente descartamos ese exceso. Ésto se traduce en una latencia prácticamente constante tal y como se refleja en la figura 14. La comparación entre ambas técnicas ya se llevó a cabo en la sección

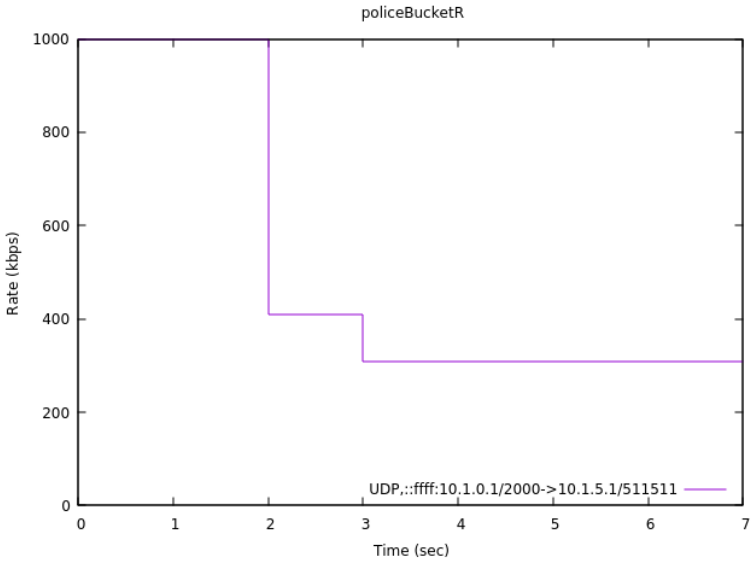

Figura 13: Tasa del flujo a la salida de la función policía.

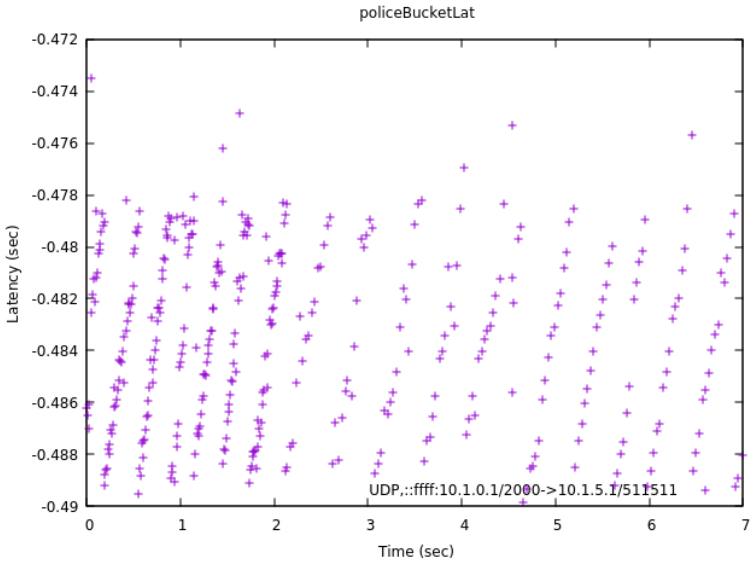

Figura 14: Latencia del flujo a la salida de la función policía.

anterior, pero se resume en que al descartar el tráfico no se introduce una latencia adicional en la función policía.

7. Conclusión

Tal y como se ve en la práctica hemos aprendido a configurar y analizar el comportamiento del algoritmo *WFQ* así como de dos técnicas de conformado.