

Reparto justo con pesos (*max-min weighted fair*)

Max-min *Weighted* Fair Share

- En ocasiones se desea una asignación preferente a unos flujos frente a otros
- Se asocia esta preferencia con unos pesos w_1, w_2, \dots, w_n
- Extensión:
 - Los recursos se asignan en orden de demanda creciente, normalizada por el peso
 - Ningún cliente recibe más de lo que solicita
 - Aquellos cuya demanda no se pueda satisfacer se reparten el remanente del recurso en proporción a sus pesos



Max-min WFS (Ejemplo)

- Recurso: 20. Demandas: 4, 2, 10 y 8. Pesos: 2.5, 4, 0.5 y 1
- Normalización de los pesos:
 - Que el menor valga 1
 - Pesos normalizados: 5, 8, 1 y 2
- En vez de 4 clientes es como si hubiera $5+8+1+2 = 16$
- $C/n = 20/16 = 1.25$
- La asignación a cada uno sería:
 - $(5 \times 1.25 =)$ 6.25, $(8 \times 1.25 =)$ 10, $(1 \times 1.25 =)$ 1.25 y $(2 \times 1.25 =)$ 2.5
 - El cliente 1 obtiene 6.25 pero solicitaba 4 luego sobra 2.25
 - El cliente 2 obtiene 10 pero solicitaba 2 luego sobra 8
 - El cliente 3 obtiene 1.25 pero solicita 10 (insuficiente)
 - El cliente 4 obtiene 2.5 pero solicita 8 (insuficiente)
- Ha sobrado $2.25 + 8 = 10.25$ a repartir entre los clientes 3 y 4
 - Sus pesos ya están normalizados (1 y 2). $C/n = 10.25 / 3 = 3.417$
 - El cliente 3 obtiene 3.417 adicional, en total $1.25 + 3.417 = 4.667$ (insuficiente)
 - El cliente 4 obtiene 6.834 adicional, en total $2.5 + 6.834 = 9.334$, sobra 1.334
 - Lo que sobra del cliente 4 se asigna al cliente 3 y así recibe $4.667 + 1.334$
- Asignación final: 4, 2, 6, 8

upna

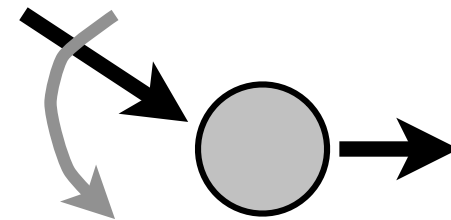
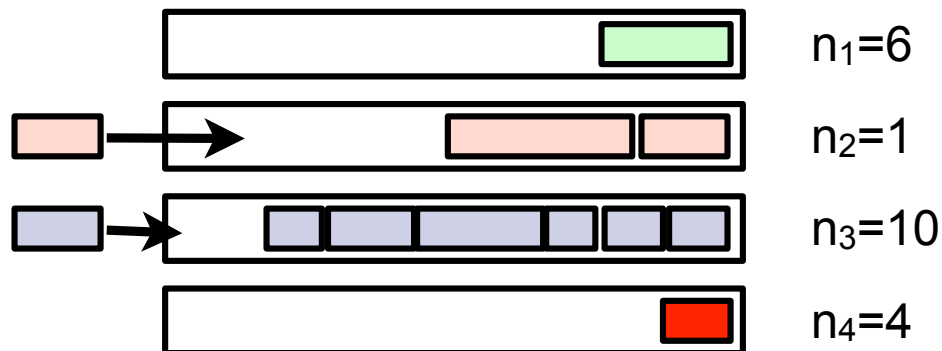
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Tecnologías Avanzadas de Red
Área de Ingeniería Telemática

Weighted Round Robin

Weighted Round Robin (WRR)

- Opera por “turnos”. Un peso para cada clase
- Se normaliza el peso por el tamaño medio de paquete en la clase $\frac{\phi(i)}{s_i}$
- Normaliza el resultado para que sean enteros
- En cada turno visita cada cola (en RR) y sirve tantos paquetes como su peso normalizado
- Ejemplo:
 - Pesos: 0.03, 0.05, 1 y 0.5
 - Tamaños medios: 50, 500, 1000 y 1200 bytes
 - Renormalizados según tamaños medios: 0.0006, 0.0001, 0.001 y 0.0004
 - Renormalizados a enteros: 6, 1, 10, 4



Weighted Round Robin (WRR)

Problemas

- Necesita saber el tamaño medio de paquete de cada clase
- Más sencillo si los paquetes son de tamaño constante
- Es justo solo por encima de la escala de la duración del turno
- Ejemplo:
 - Enlace T3 (45Mbps)
 - 500 PVCs ATM con peso 1 y 500 PVCs con peso 10
 - Supongamos que todos los PVCs tiene tráfico
 - Un turno requiere enviar: $500 \times 1 + 500 \times 10 = 5500$ celdas → 51.82 ms
 - Por debajo de una escala de 50ms unos PVCs reciben más que otros
- El retardo que sufre una clase depende del número de clases que haya
- Hay implementaciones que lo combinan con una cola de prioridad

upna

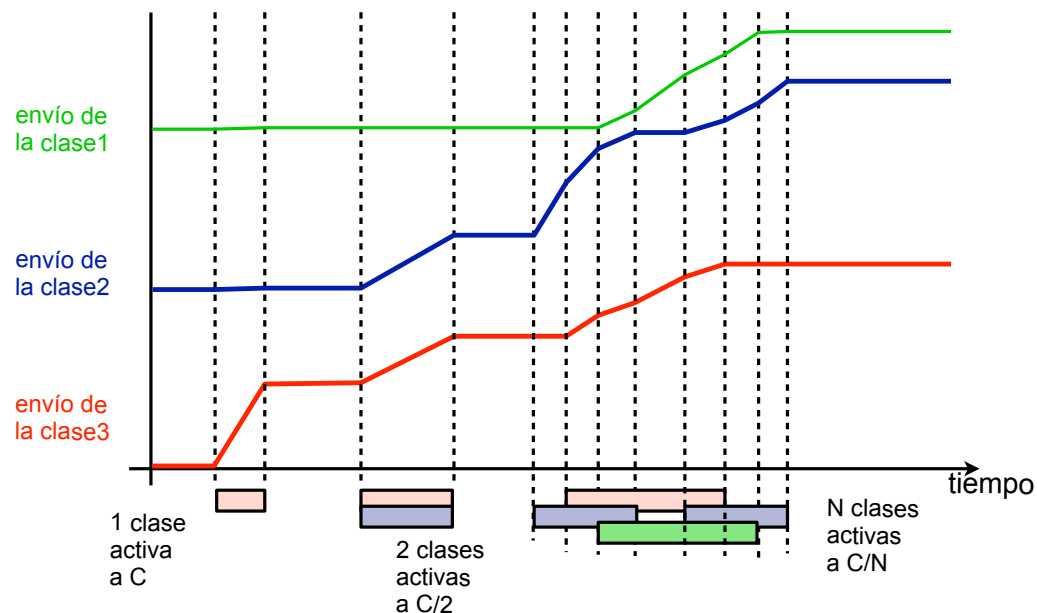
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Tecnologías Avanzadas de Red
Área de Ingeniería Telemática

Weighted Fair Queueing

GPS

- Partiendo de *Processor Sharing* se asocia un peso $\phi(i)$ a cada cola y entonces la cantidad de servicio es proporcional al mismo
- Ofrece *weighted max-min fairness* y lo llamamos *Generalized Processor Sharing (GPS)*
- En cualquier caso, en la realidad no podemos servir fluidos sino que servimos paquetes así que solo podremos aproximarlos
- Round Robin es una aproximación a PS
- WRR es una aproximación a GPS



$$c_i = C \frac{\phi(i)}{\sum \phi(i)}$$



WFQ

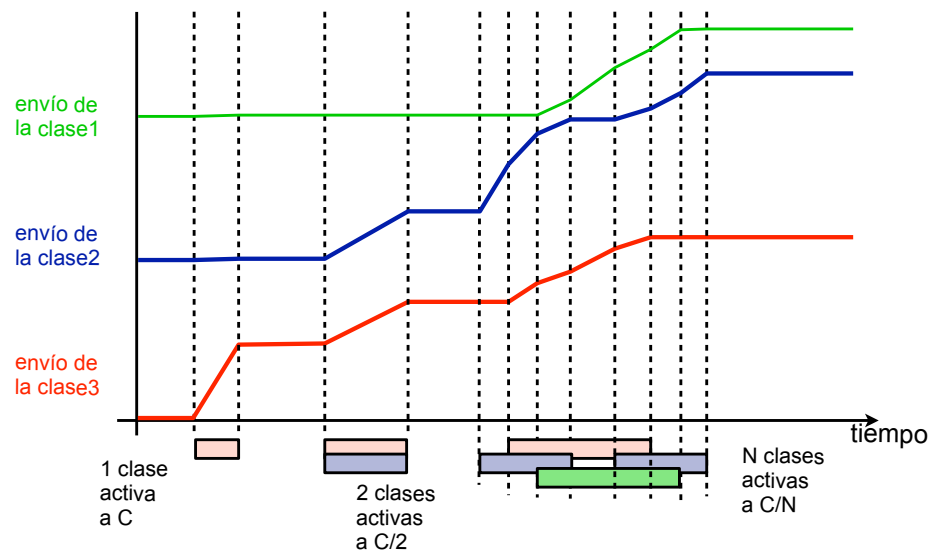
- *Weighted Fair Queueing*
- Aproximación de GPS (*Generalized Processor Sharing*) para el caso de paquetes
- Equivalente a PGPS (*Packet-by-packet Generalized Processor Sharing*)
- No requiere conocer el tamaño medio de paquete
- Emplea un reloj virtual
- Calcula el final virtual en que se enviaría cada paquete en el caso ideal GPS
- Se envían en orden de tiempo final virtual
- Más complejo de implementar
- Puede ofrecer *worst-case bounds*
- En resumen
 - Simular GPS
 - Enviar los paquetes en el mismo orden que GPS



WFQ

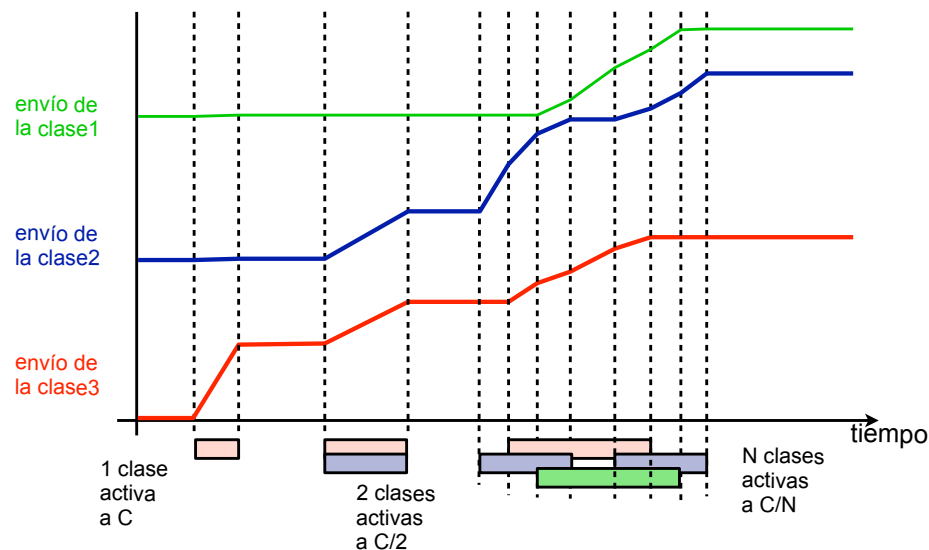


- Se simulan “turnos” (*rounds*)
- Supongamos que no hay pesos
- Supongamos que GPS no sirve fluido perfecto sino bit-a-bit
- **En cada turno se envía 1 bit de cada flujo**
- El número de turno (*round number*) es el número de turnos bit-a-bit que se han completado en un instante
- Así, cuantos más flujos activos simultáneos hay, más despacio se incrementa el turno con el tiempo
- Ignoramos el servir bit-a-bit si definimos el round number como un valor que crece inversamente proporcional al número de flujos activos



WFQ

- Un paquete k del flujo i que llega en el instante t
- Su *finish number* $F(i,k,t)$:
 - Si flujo está inactivo: el *round number* actual + el tamaño en bits
 - Si flujo está activo: $\max[F(i,k-1,t), \text{round_number}] + \text{tamaño en bits}$
- Una vez calculado para un paquete no hay que recalcularlo ante nuevas llegadas
- Si llega a una cola llena se descartan paquetes en orden decreciente de finish number hasta que quepa



WFQ

- Calcular el round number es complejo
- Hay que hacerlo para cada paquete que llega y por cada uno que se envía
- En el caso con pesos a la hora de calcular el finish number:
 - Si flujo inactivo: el *round number* actual + tamaño / peso
 - Si flujo activo: $\max[F(i,k-1,t), \text{round_number}] + \text{tamaño} / \text{peso}$
- y el round number se incrementa con el inverso de la suma de los pesos
- Existen variantes para simplificar este cálculo:
 - Self-Clocked Fair Queuing (SCFQ)
 - Start-Time Fair Queuing

upna

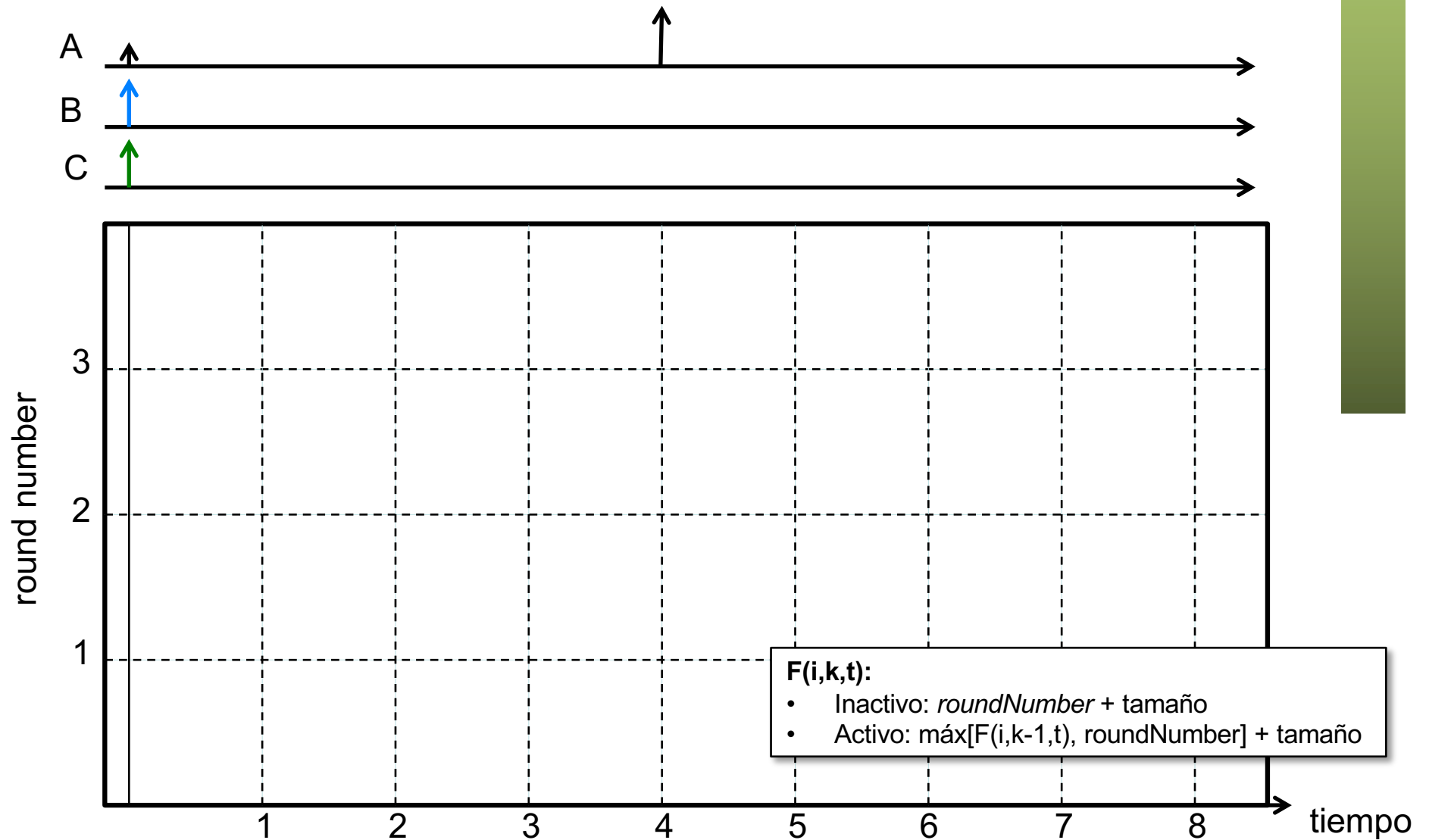
Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Tecnologías Avanzadas de Red
Área de Ingeniería Telemática

WFQ: Ejemplo

WFQ (Ejemplo)

- Enlace a 1 unidad/s
- Llegadas de tamaños 1, 2 y 2 unidades en $t=0$ y de tamaño 2 unidades en $t=4$



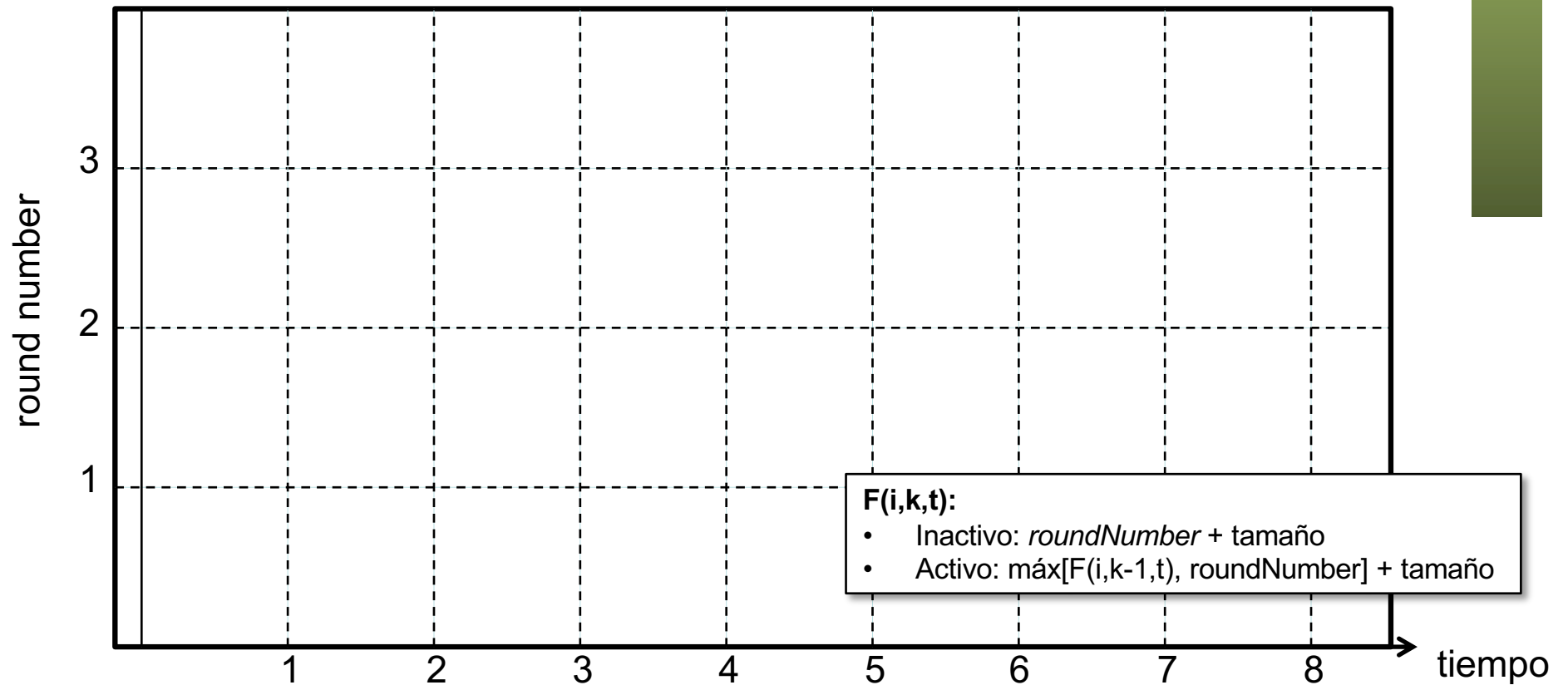
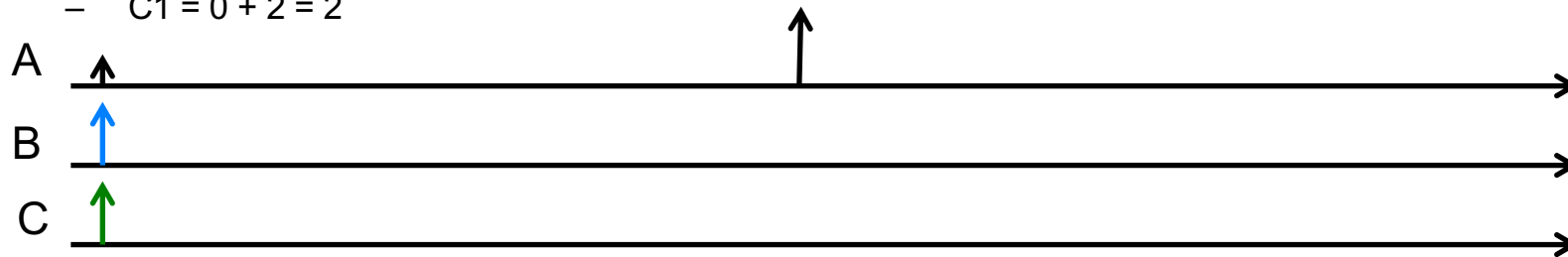
WFQ (Ejemplo)

- Finish numbers:

- $A1 = 0 + 1 = 1$

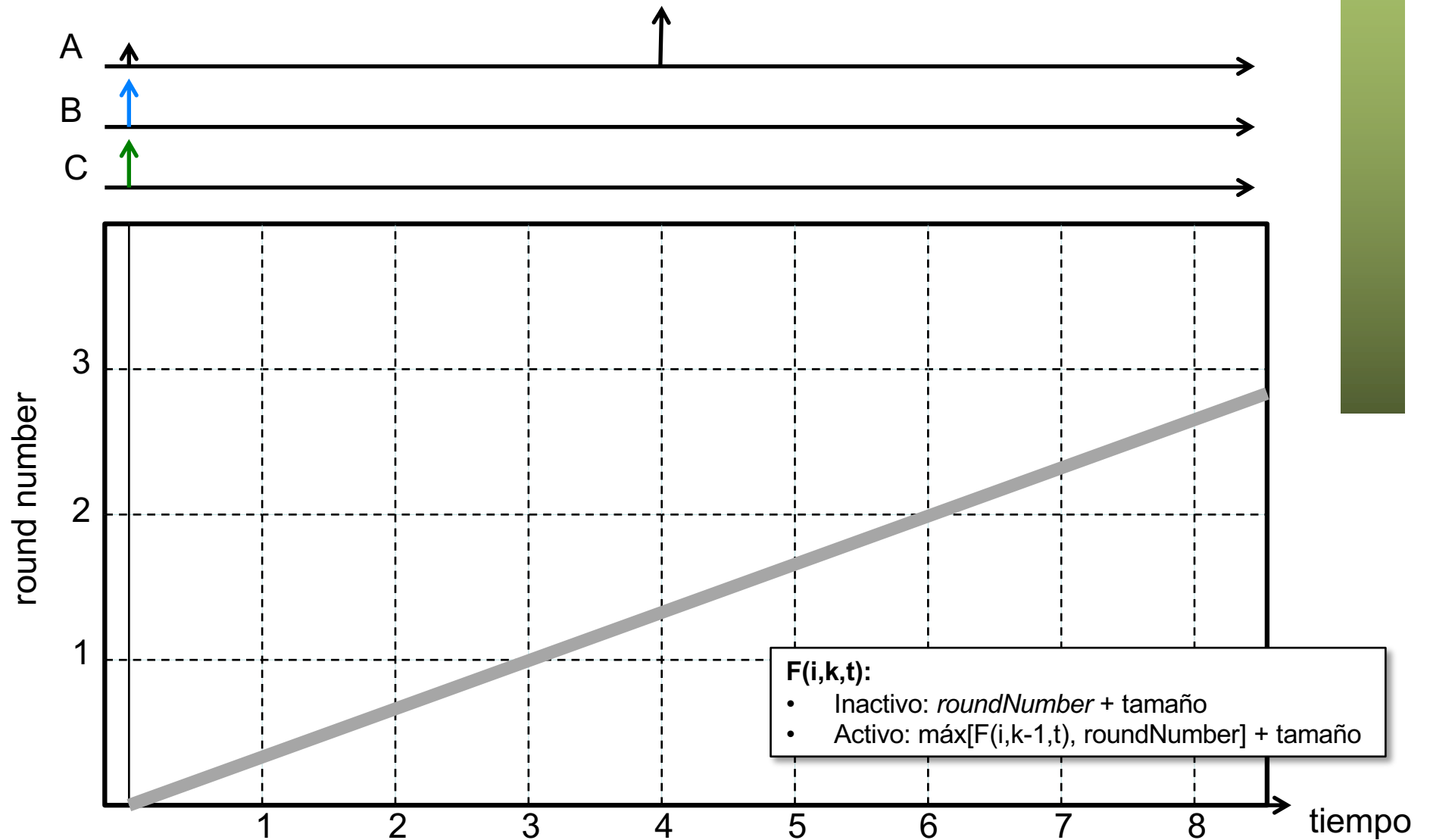
- $B1 = 0 + 2 = 2$

- $C1 = 0 + 2 = 2$



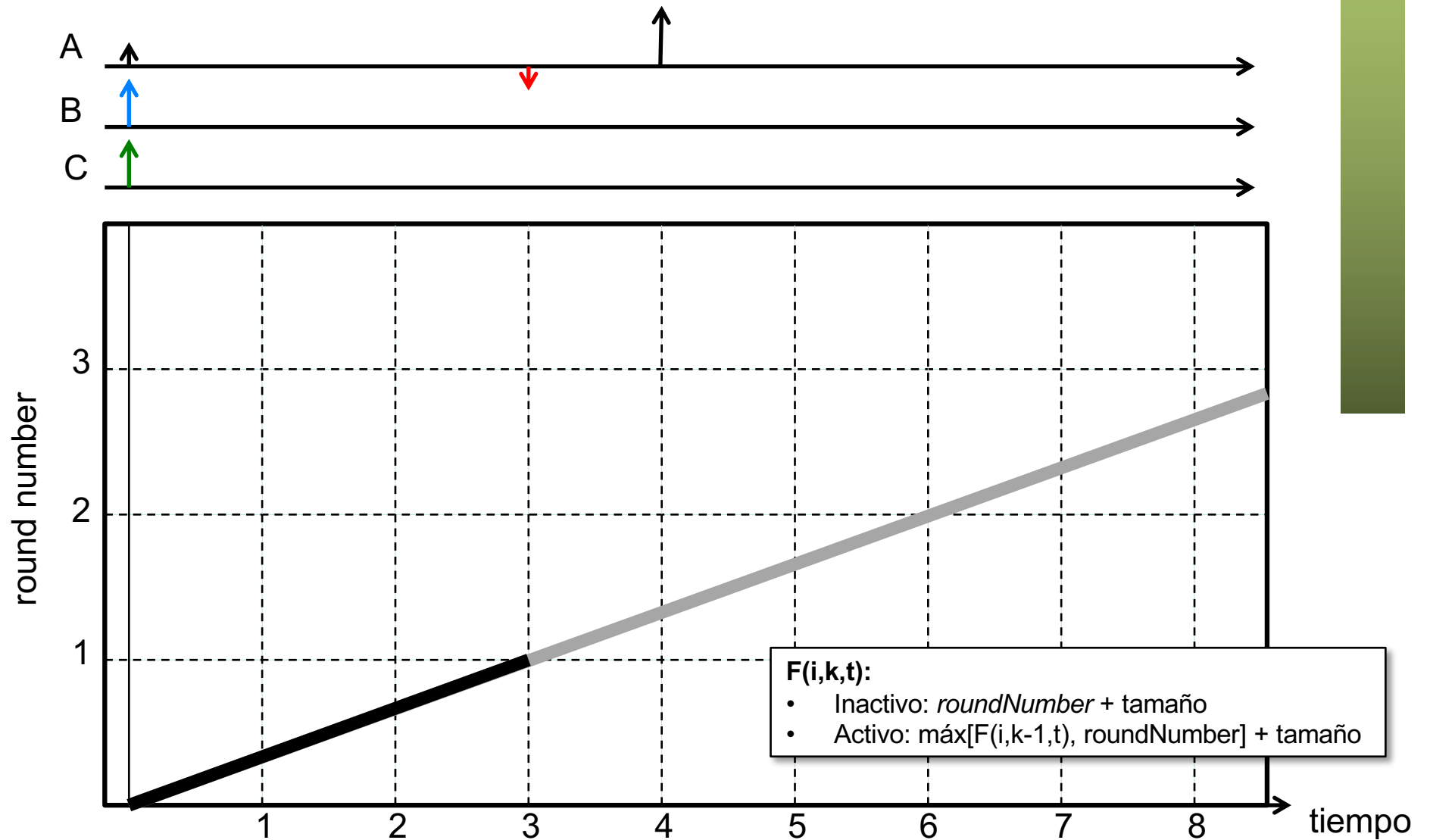
WFQ (Ejemplo)

- Hay 3 flujos a enviar simultáneamente
- El round number se incrementa a $C/3 = 1/3$ por unidad de tiempo



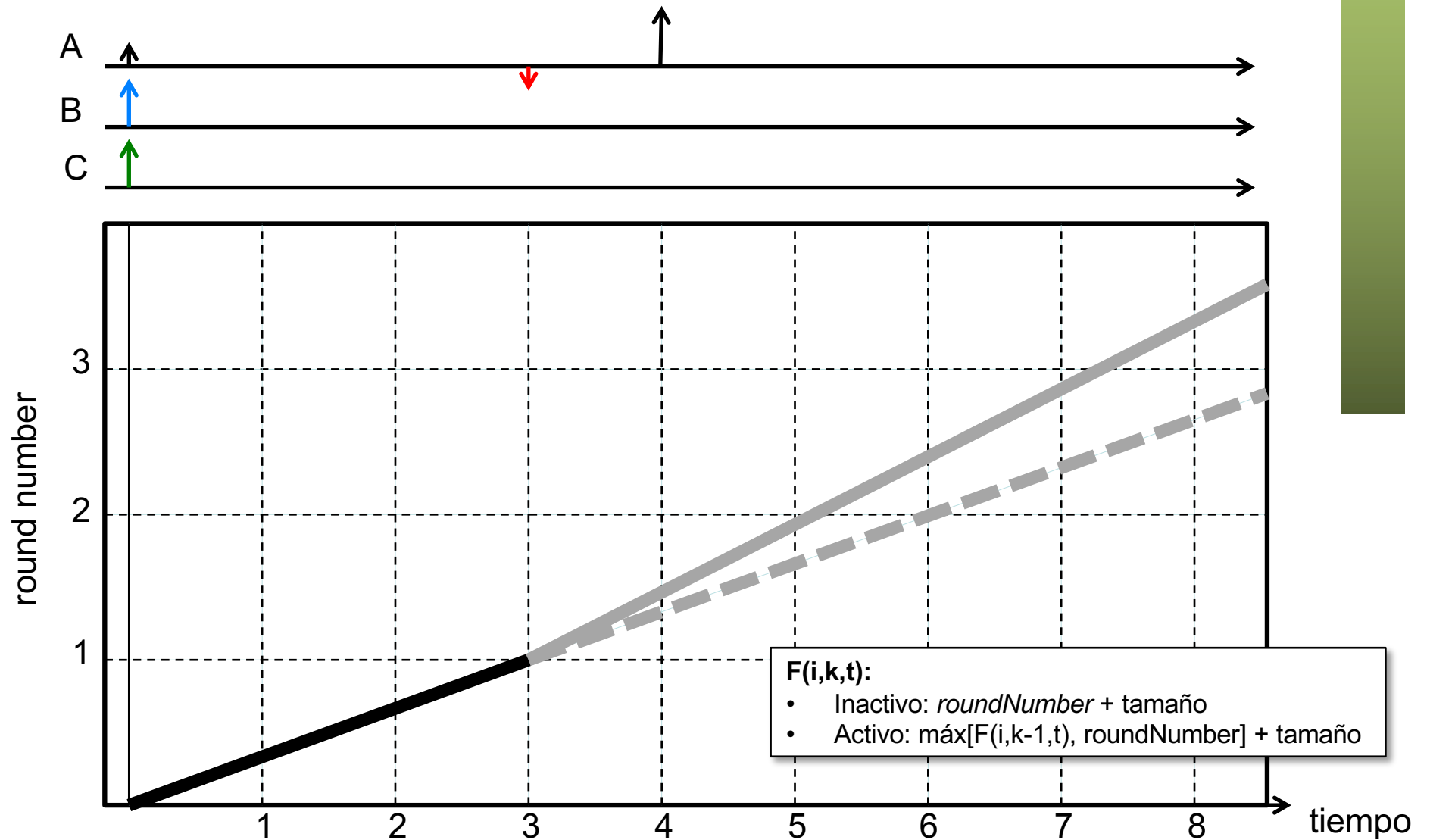
WFQ (Ejemplo)

- En el instante $t=3$ se han servido 3 bits, eso es uno por flujo y por lo tanto termina el round 1 y termina de enviarse A1



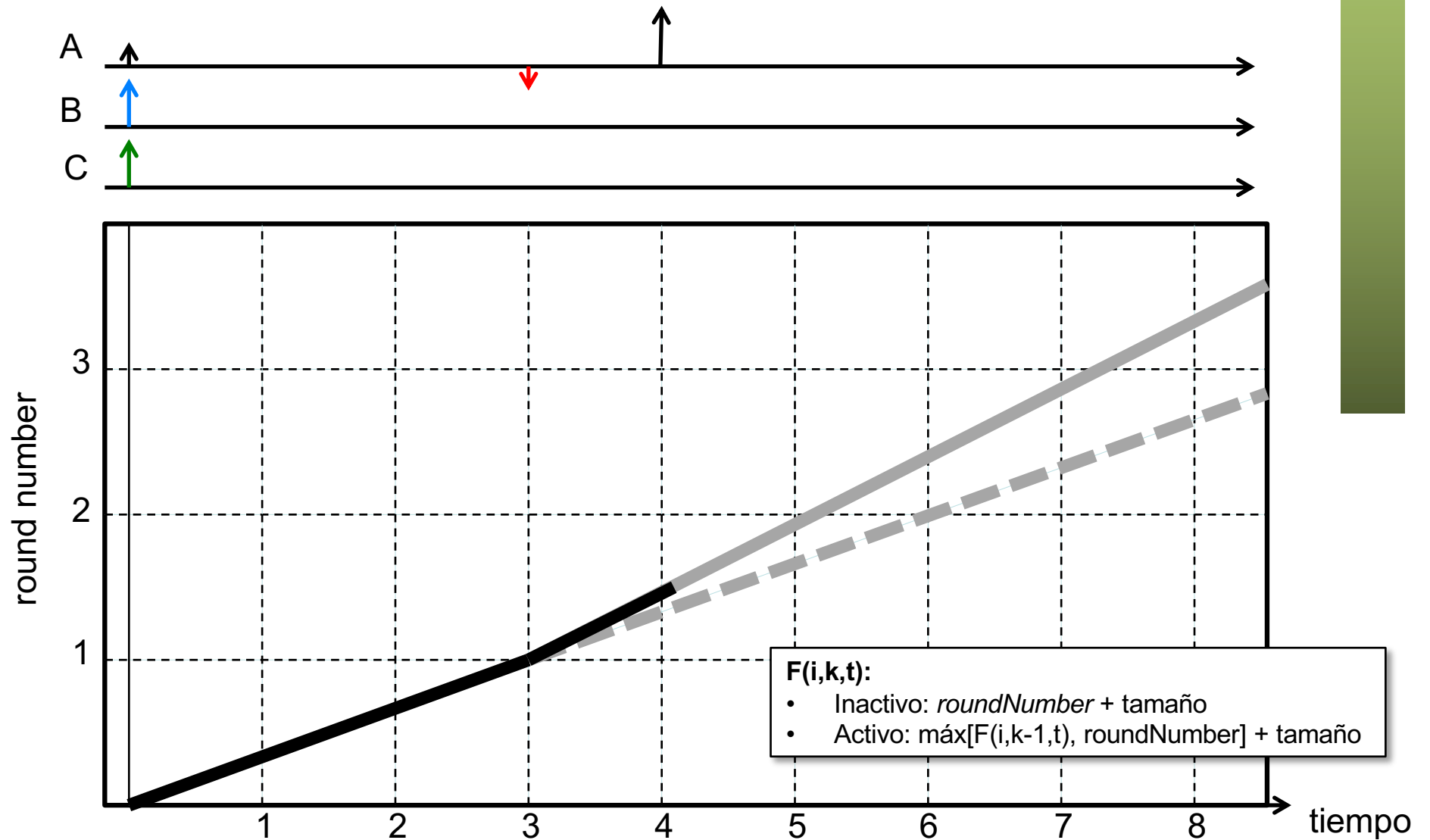
WFQ (Ejemplo)

- A partir de ahí se siguen sirviendo B1 y C1 con finish number = 2
- Al haber dos flujos activos crece el round number a 1/2



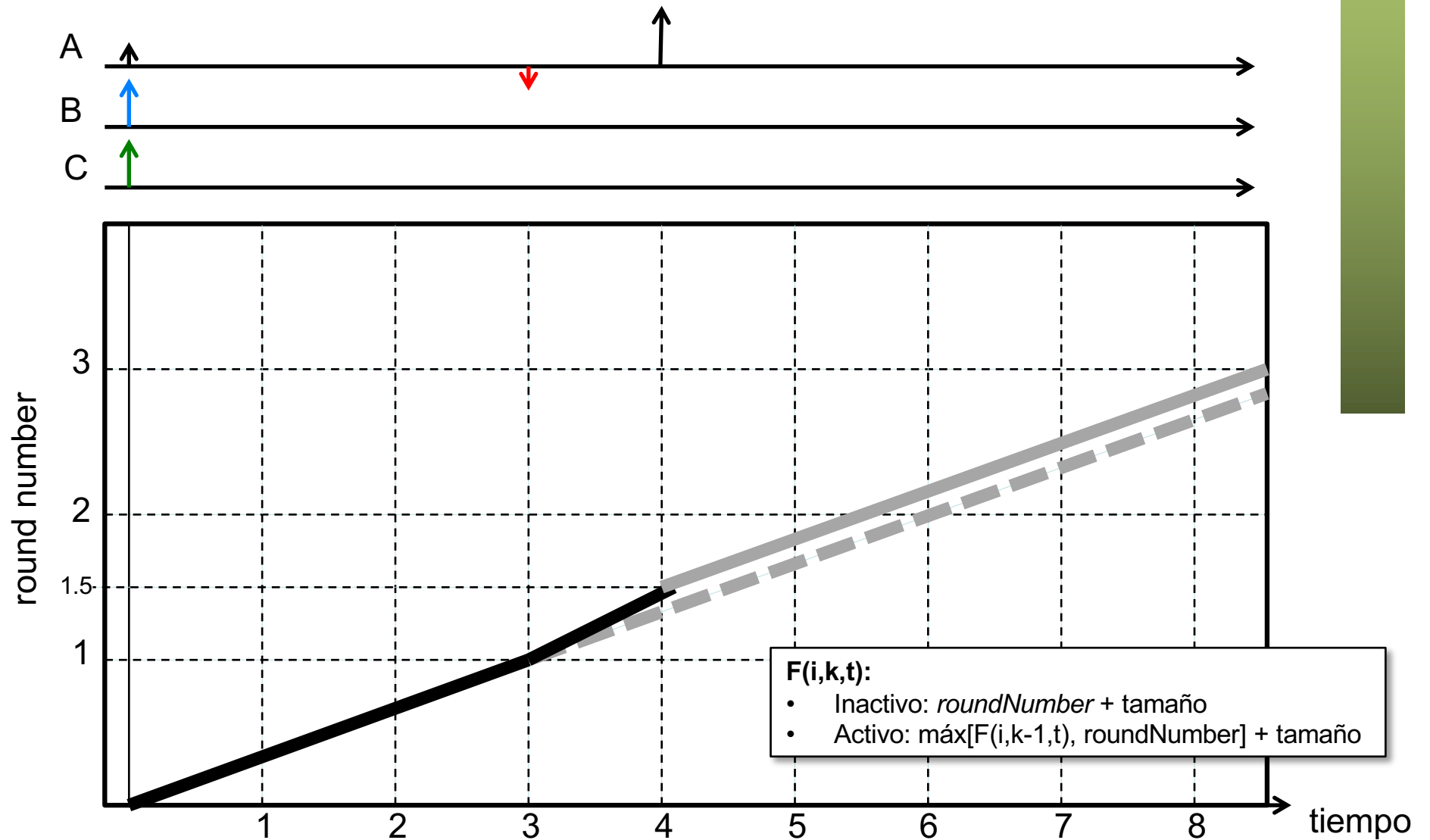
WFQ (Ejemplo)

- B1 y B2 terminarían de enviarse al alcanzar round number = 2 (t = 5) pero llega antes A2



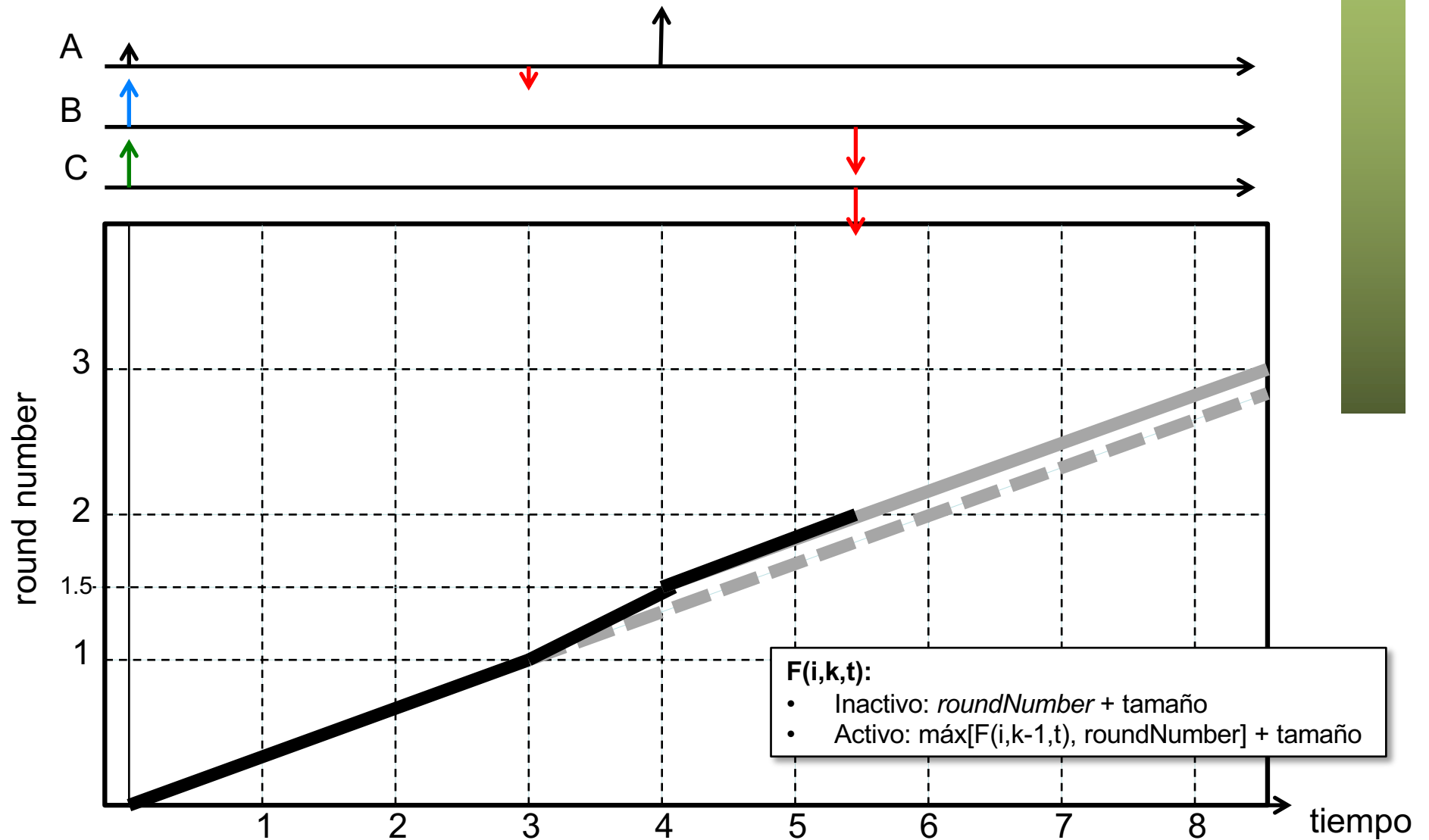
WFQ (Ejemplo)

- Finish number de A2 es $1.5 + 2 = 3.5$
- A partir de $t=4$ vuelve a haber 3 flujos simultáneos



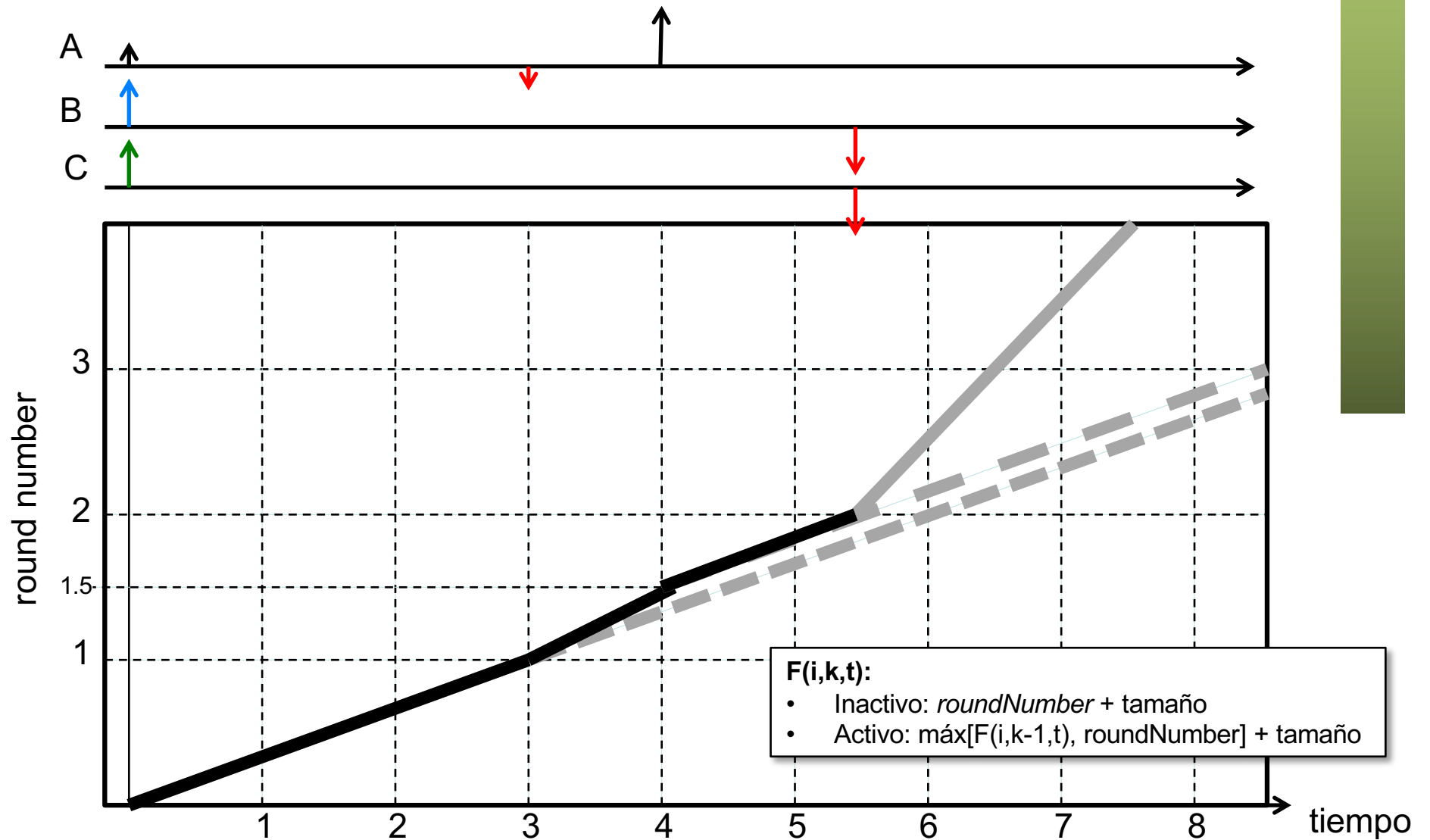
WFQ (Ejemplo)

- Se alcanza el round number 2 en $t = 4 + 0.5/(1/3) = 5.5$
- Entonces se completaría el envío GPS de B1 y C1



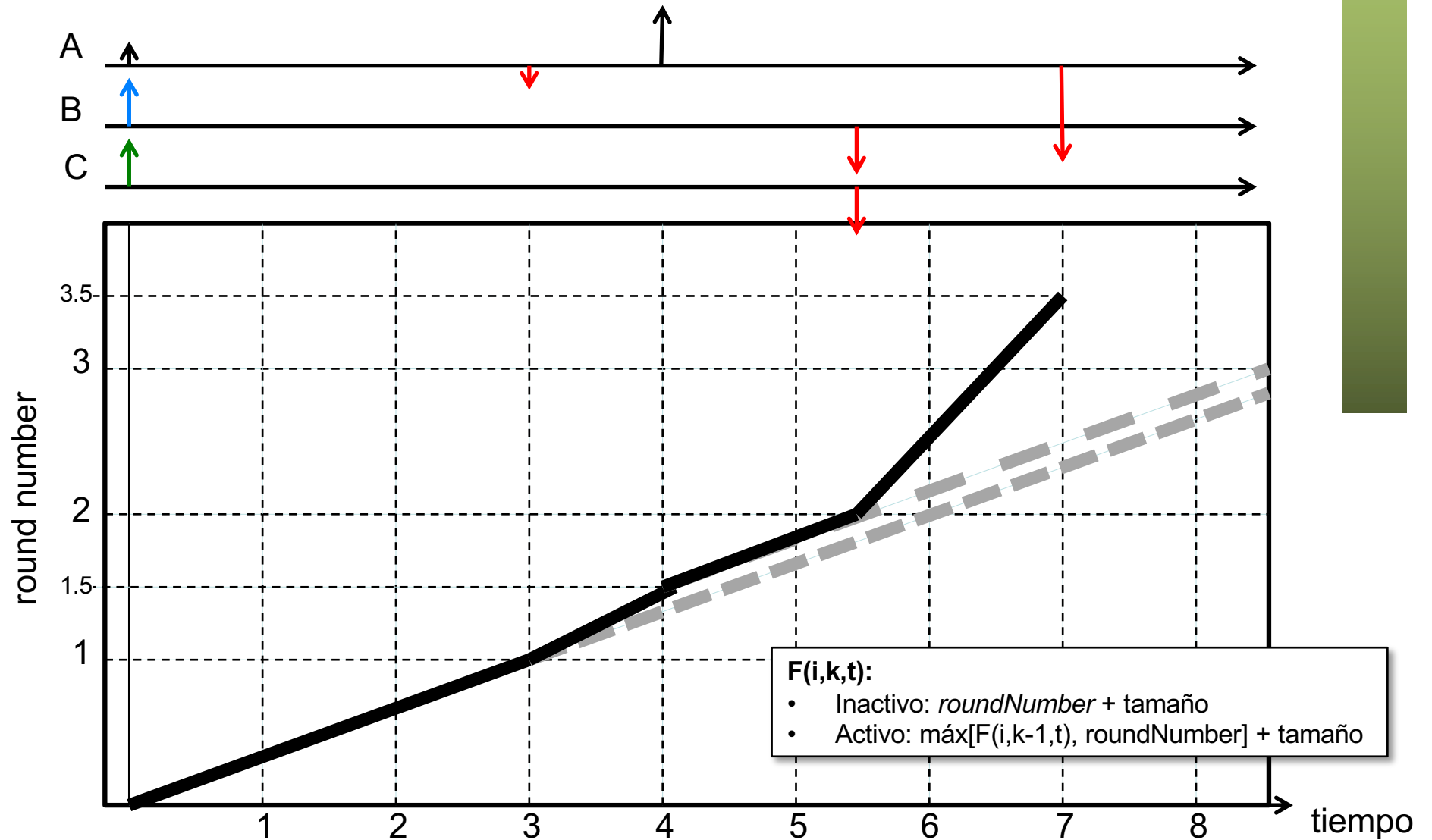
WFQ (Ejemplo)

- Queda solo una fuente activa luego ahora se avanza a 1 round por unidad de tiempo



WFQ (Ejemplo)

- Queda solo una fuente activa luego ahora se avanza a 1 round por unidad de tiempo
- En $t = 7$ se alcanza el round number 3.5 y termina de enviarse A2



WFQ (Ejemplo)

- Los *Finish numbers* han resultado ser:
 - $A1 = 1$
 - $B1 = 2$
 - $C1 = 2$
 - $A2 = 3.5$
- Los paquetes se envían en orden de sus finish numbers
- En caso de empate se envían los empatados en cualquier orden