- 10. A Demers, S Keshav, S Shenker, "Analysis and Simulation of a Fair Queueing Algorithm," SIGCOMM 1989.

- 11. JCR Bennett, H Zhang, "WF^2Q: Worst-case fair weighted fair queueing," IEEE Infocom, 1996.

Weighted Fair Queueing (WFQ)

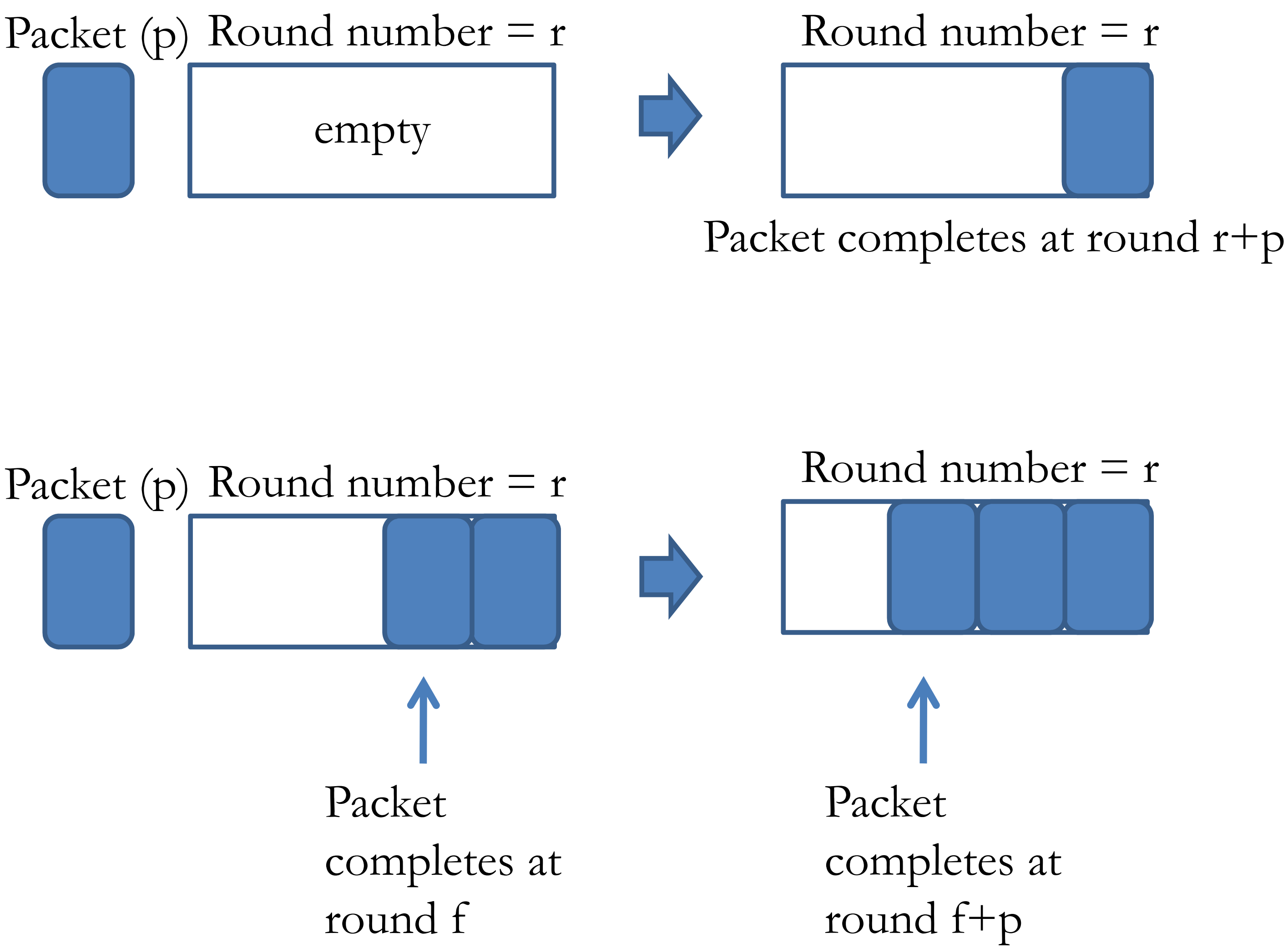
- Deals better with variable size packets and weights
- The idea is that assume GPS is fairest discipline
- Find the *finish time* of a packet, *had we been doing GPS*
- Then serve packets in order of their **finish times**
- The scheduler tries to emulate the order in which packets are processed by GPS

WFQ: first cut

- Suppose, in each *round*, the server served one bit from each active connection
  - begins with emulating bit-by-bit Round-Robin
- *Round number* is the number of rounds already completed
  - can be fractional
- Each round of service takes a variable amount of time
  - The more connections served, the longer the round takes

WFQ (cont'd)

- If a packet of length ***p*** arrives to an empty queue when the round number is ***R***, it will complete service when the round number is ***R + p*** => *finish number* is *R + p*
  - independent of the number of other connections!
- If a packet arrives to a non-empty queue, and the previous packet has a finish number of ***f*****,** then the packet's finish number is ***f+p***
- Serve packets **in order of finish numbers**

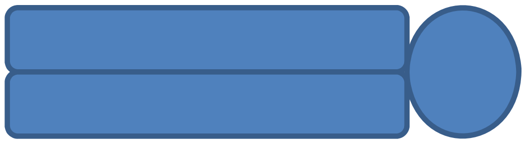

Packet (p)
Round number = r
empty
Round number = r
Packet completes at round r+p
Packet (p)
Round number = r
Round number = r
Packet completes at round f
Packet completes at round f+p

WFQ: computing the round number

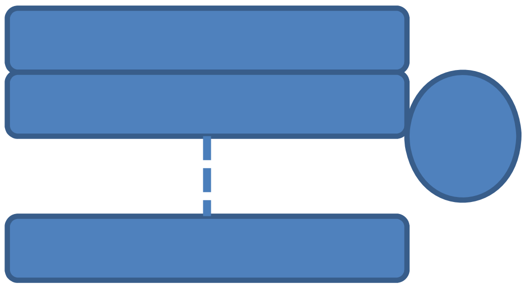
- Naively: round number = number of rounds of service completed so far
  - what if a server has not served all connections in a round?
  - what if new conversations join in halfway through a round?
- *Redefine* round number as a real-valued variable that **increases at a rate inversely proportional to the number of currently active connections**
- With this change, WFQ emulates GPS instead of bit-by-bit RR

Increase is round number is proportional to rate of service for each per queue/flow

1 queue/flow, round number increases at rate of 1 unit



2 queues/flow,s round number increases at rate of 1/2



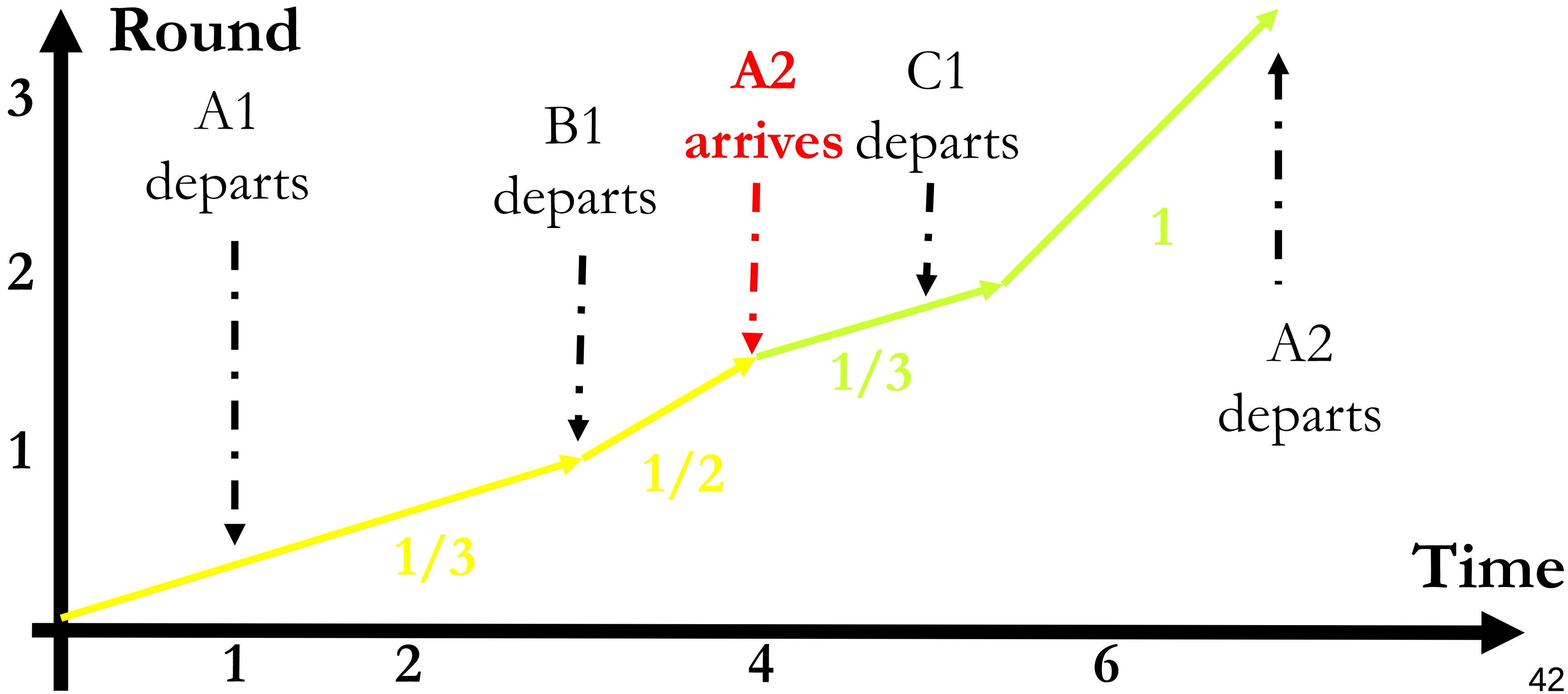
n queues/flow,s round number increases at rate of 1/n

WFQ implementation

- On packet arrival:
  - classify packet and look up finish number of last packet served (or waiting to be served)
    - O(1) to O(N)
  - re-compute round number
    - worst case O(N)
  - compute finish number
  - insert in priority queue sorted by finish numbers
    - O(logN)
  - if no space, drop the packet with largest finish number
- On service completion
  - select the packet with the lowest finish number

Example: FQ

- Three connections: A,B,C. At t=0, packet of size 1,2 and 2 arrives. (A1,B1,C1). Finish time: A1 = 1, B1 = C1 = 2.
- With GPS, at t=3, round 1 is completed, A1 departs, only 2 connections active
- At t=4, round is 1.5, A2 of size 2 arrives, finish time is (1.5+2) 3.5



Example: GPS

- Three connections: A,B,C. At t=0, packet of size 1,2 and 2 arrives. (A1,B1,C1). At t=4, A2 of size 2 arrives
- Using GPS:
  - At t=3, all packets get 1 bit of service
    - A1 departs
  - At t=4, B1 and C1 get 1.5 bits of service
    - A2 arrives
  - At t=5 1/2, B1 and C1 get 2 bits of service
    - A2 gets ½ bits of service
    - B1 and C1 depart
  - At t=7, A2 departs
  - Sequence of service = A1, {B1,C1}, A2
  - Departure time = 3, 5.5, 5.5, 7

Example

- FQ
  - Finish #:
    - A1 = 1, B1 = C1 = 2
    - A2 = 3.5
  - Sequence of service: A1, {B1,C1}, A2
  - Departure Time: 1, 3, 5, 7
- GPS
  - Sequence of service: A1, {B1,C1}, A2
  - Departure time: 3, 5.5, 5.5, 7

Example

A queue has service rate of 3 bit/s. Packet arrived are shown in the table below and a GPS scheduler is used.

Packet	A	B	C
Time	0	0.25	0.5
Size (bits)	2	1	1

Evaluation

- Pros
  - like GPS, it provides protection
  - can obtain worst-case end-to-end delay bound
  - gives users incentive to use intelligent flow control (and also provides rate information implicitly)
- Cons
  - needs per-connection state
  - iterated deletion is complicated (occurs during round number computation)
  - requires a priority queue

Light Load

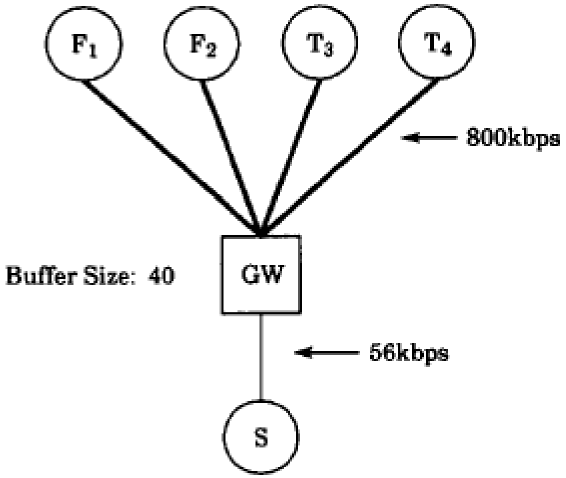


Table II. Scenario 1

Quantity	Policy	FTP 1	FTP 2	Telnet 3	Telnet 4
Throughput (packets)	G/FCFS	1746	1746	99	96
	G/FQ	1746	1746	102	94
	JK/FCFS	1747	1745	102	104
	JK/FQ	1746	1746	105	103
	DEC/DEC	1746	1746	97	98
	DEC/FQbit	1745	1746	83	88
Average round-trip time	G/FCFS	1·43	1·43	1·36	1·35
	G/FQ	1·43	1·43	0·079	0·091
	JK/FCFS	1·43	1·43	1·35	1·36
	JK/FQ	1·43	1·43	0·084	0·089
	DEC/DEC	0·286	0·286	0·206	0·218
	DEC/FQbit	1·38	1·39	0·088	0·074
Retransmitted packets	G/FCFS	0	0	0	0
	G/FQ	0	0	2	1
	JK/FCFS	0	0	0	0
	JK/FQ	0	0	0	0
	DEC/DEC	0	0	0	0
	DEC/FQbit	0	0	0	0
Dropped packets	G/FCFS	0	0	0	0
	G/FQ	0	0	0	0
	JK/FCFS	0	0	0	0
	JK/FQ	0	0	0	0
	DEC/DEC	0	0	0	0
	DEC/FQbit	0	0	0	0

High Load

Table III. Scenario 2

Quantity	Policy	FTP 1	FTP 2	FTP 3	FTP 4	FTP 5	FTP 6	Telnet 7	Telnet 8
Throughput (packets)	G/FCFS	18	1154	1159	3	1149	15	31	3
	G/FQ	178	838	591	600	615	621	96	98
	JK/FCFS	582	583	585	585	583	582	3	0
	JK/FQ	574	579	546	594	599	601	87	96
	DEC/DEC	582	582	582	582	582	582	90	99
	DEC/FQbit	582	582	582	582	582	582	83	89
Average round-trip time	G/FCFS	403	2·18	2·16	—	2·18	140	115	—
	G/FQ	16·8	3·31	4·88	4·83	4·53	4·47	0·079	0·078
	JK/FCFS	1·85	1·93	1·93	1·85	1·93	1·85	—	—
	JK/FQ	1·75	1·78	1·19	1·86	2·20	2·16	0·091	0·085
	DEC/DEC	0·859	0·859	0·859	0·859	0·859	0·859	0·783	0·778
	DEC/FQbit	1·59	1·59	1·59	1·59	1·59	1·59	0·088	0·089
Retransmitted packets	G/FCFS	43	10	7	6	9	17	25	5
	G/FQ	73	224	176	168	243	159	2	2
	JK/FCFS	57	57	57	57	57	57	6	0
	JK/FQ	83	80	60	64	61	61	0	0
	DEC/DEC	0	0	0	0	0	0	0	0
	DEC/FQbit	0	0	0	0	0	0	0	0
Dropped packets	G/FCFS	26	5	4	3	5	11	15	2
	G/FQ	33	139	106	88	167	98	0	0
	JK/FCFS	56	56	56	56	56	56	5	0
	JK/FQ	80	76	48	61	57	54	0	0
	DEC/DEC	0	0	0	0	0	0	0	0
	DEC/FQbit	0	0	0	0	0	0	0	0

Table IV. Scenario 3

Quantity	Policy	FTP 1	Telnet 2	Ill-behaved 3
Throughput (packets)	G/FCFS	3	11	3497
	G/FQ	3491	95	5
	JK/FCFS	0	0	3500
	JK/FQ	3489	110	6
	DEC/DEC	0	0	3500
	DEC/FQbit	3489	108	5
Average round-trip time	G/FCFS	1362	2·87	2·97
	G/FQ	0·716	0·080	903
	JK/FCFS	—	—	2·83
	JK/FQ	0·716	0·085	860
	DEC/DEC	—	—	2·85
	DEC/FQbit	0·626	0·077	918
Retransmitted packets	G/FCFS	7	139	0
	G/FQ	0	2	0
	JK/FCFS	2	0	0
	JK/FQ	0	0	0
	DEC/DEC	1	1	0
	DEC/FQbit	0	0	0
Dropped packets	G/FCFS	7	127	3504
	G/FQ	0	0	6995
	JK/FCFS	2	0	3500
	JK/FQ	0	0	6994
	DEC/DEC	1	1	3500
	DEC/FQbit	0	0	6994

WFQ Variants

- There are many WFQ variants that are easier to implement and provides different levels of performance bounds
  - SCFQ – self clock fair queueing (1994)
  - DRR – Deficit Round-Robin (1995)
  - W^2FQ – worst-case fair WFQ (1996)
  - and many, many more ....
- In practice, when WFQ variants are available on routers, the number of classes/flows supported tend to be small

How "fair" is WFQ

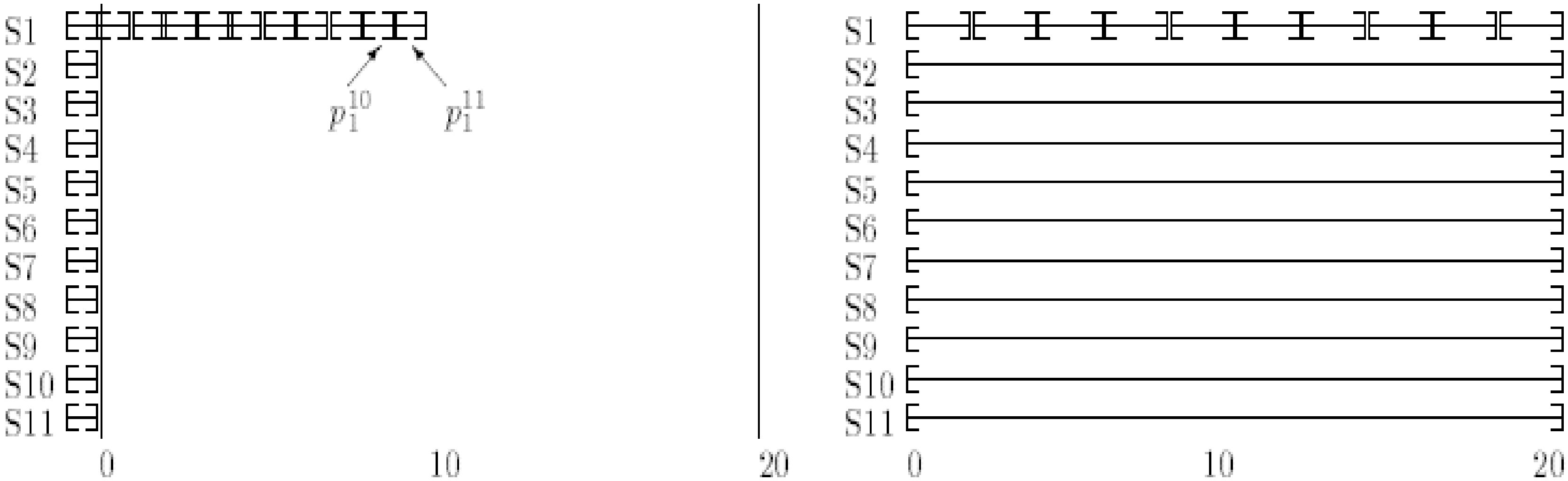
- Unweighted case:
  - if GPS has served *x* bits from connection A by time t
  - WFQ would have served at least *x - P* bits, where *P* is the largest possible packet in the network
  - However, WFQ could send *much more* than GPS would => absolute fairness bound > *P*

$$d_{i,WFQ}^{k} - d_{i,GPS}^{k} \leq \frac{L_{max}}{r} \quad \forall i, k \qquad (5)$$

$$W_{i,GPS}(0,\tau) - W_{i,WFQ}(0,\tau) \leq L_{max} \quad \forall i, \tau \qquad (6)$$

- *Is this fair enough?*

Motivation



(a) Packet Arrivals

(b) GPS Service Order

Figure 1: An Example

Weight of S1 = 10, all the rest are 1

Example of “unfairness”

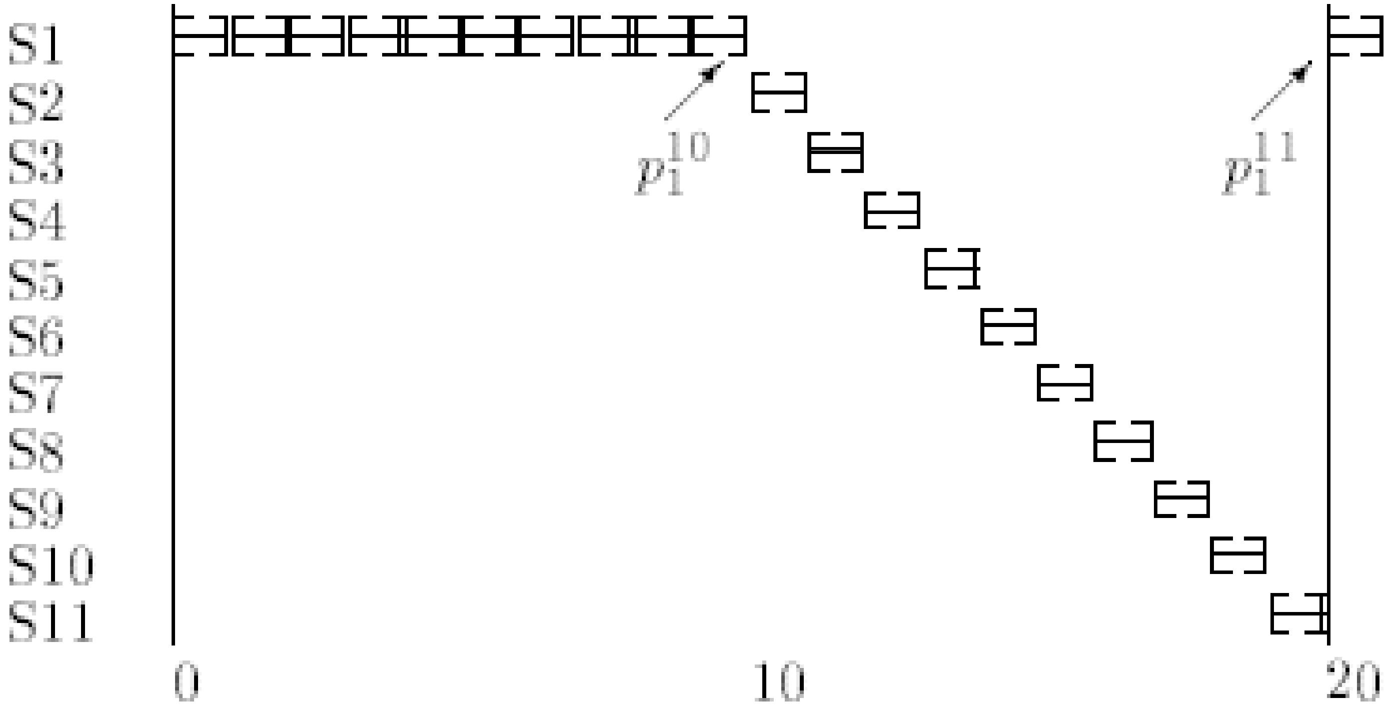


Figure 2: WFQ Service Order

W2FQ

- A packet can arrive later and yet be served earlier
- WFQ: choose among all packets, the first packet that would complete its service
- W2FQ: considers only packets that would have started (even completed) under GPS
  - Among these (eligible) packets, choose the packet that would have completed first

Example

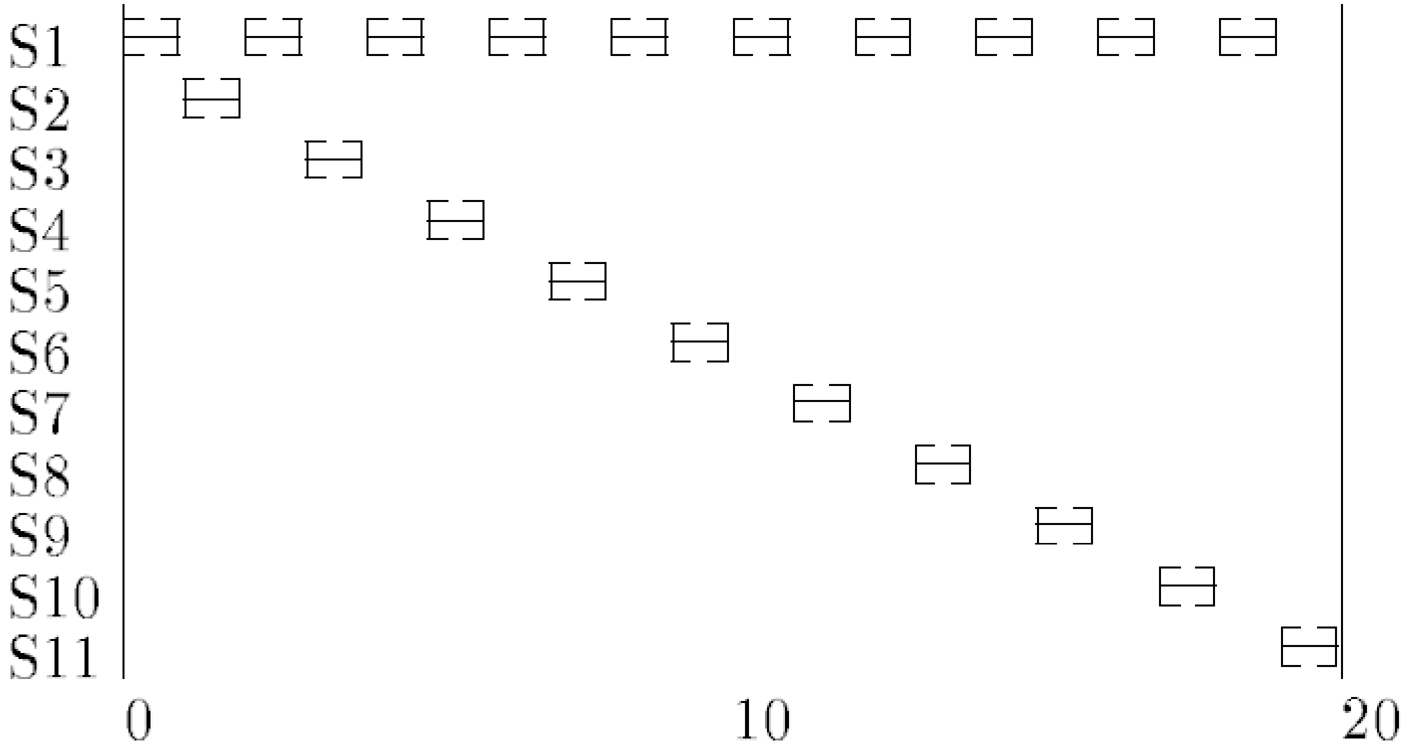


Figure 4: WF^2Q Service Order