BSFQ: Bin Sort Fair Queueing

Shun Yan Cheung and *Corneliu S. Pencea*

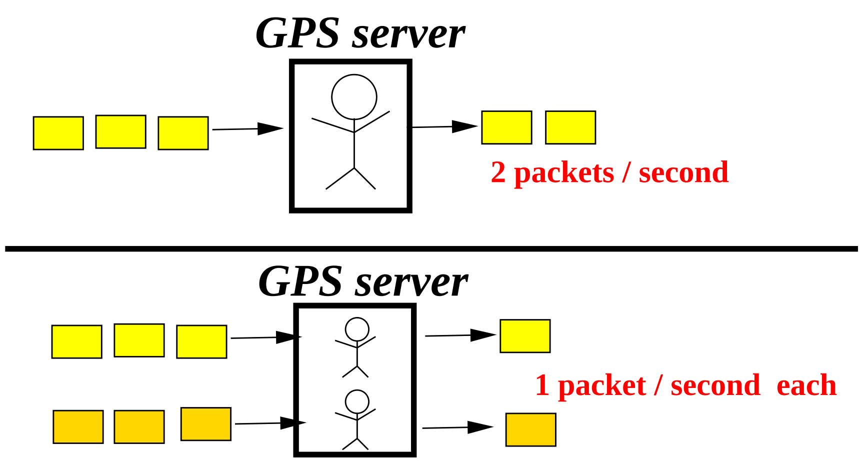
Emory University
Atlanta, Georgia 30322

Providing QoS to Flows

- Types of Quality of Service:
  - Throughput (rate) guarantee
  - Delay guarantee
  - Lossless guarantee
- Techniques to provide QoS:
  - Packet scheduling
  - Buffer management

Providing QoS through Scheduling

- Canonical approach: General Processor Sharing (GPS)



- provides rate guarantee naturally through flow isolation
- GPS also provides
  - delay guarantee (Parekh '93 & '94)
  - perfect fairness (Golestani '95)

Approximations of WFQ/GPS

- Packetized implementation of GPS: WFQ (and WF^2Q)
  - Schedule packets according their completion time in a (parallel) GPS system.
  - Very computationally expensive.
- Approximations of WFQ:
  - Frame-based Approaches
    - transmit packets in rounds (1 frame = 1 round)
    - Packet scheduling is constant time ($O(1)$)
  - Sorted-Priority Approaches
    - assign priority values to packets
    - needs sort operation in scheduling

Frame-based Techniques

- Deficit Round Robin (Shreedhar & Varghese '98)
  - Use deficit counter to carry over unused credit
  - O(1) runtime complexity
    — packet processing and flow addition/removal
  - Flow is bursty
- Uniform Round Robin (Matsufuru & Aibara '99)
  - Computes a "spreading" schedule, e.g., 1 2 1 3 1 4
  - Less bursty but flow addition/removal is $O(n)$ (n slots)
- Smoothed Round Robin (Guo '01)
  - Use weight spread sequence and weight matrix
  - Less bursty but flow addition/removal is $O(m)$
    (m = # weight spread sequences)

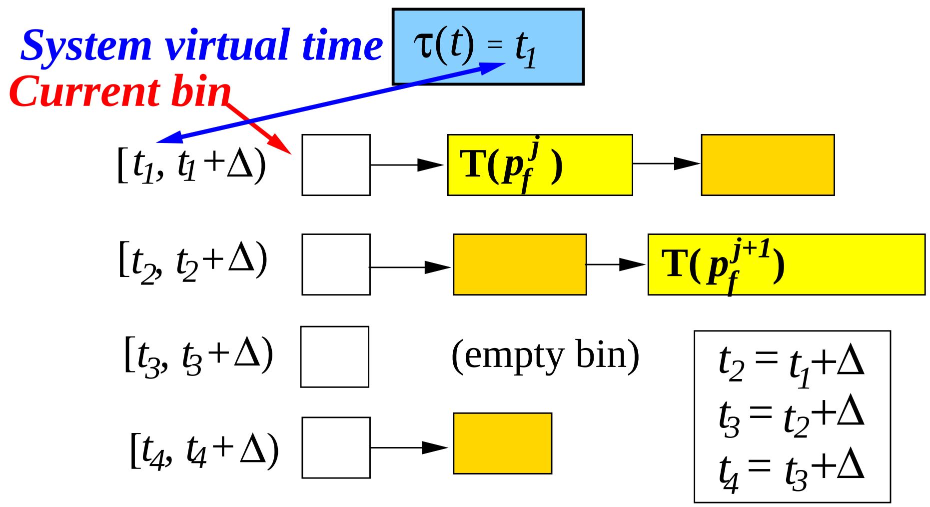
Sorted-Priority Techniques

- Virtual Clock (Zhang '90):
  - $T(p_f^j) = \max(A(p_f^j), T(p_f^{j-1})) + \frac{\ell_f^j}{r_f}$
  - Provides delay guarantee (Xie & Lam '95)
  - unfair...
- Leap Forward Virtual Clock (Suri et. al. '97)
  - Fixed unfairness in VC by advancing system clock is too far behind time stamp value of current packet serviced
- Self-Clocked Fair Queueing (Golestani '94):
  - $T(p_f^j) = \max(VT(A(p_f^j)), T(p_f^{j-1})) + \frac{\ell_f^j}{r_f}$
- Many others...
- Disadvantage of Sorted-Priority approach: packet processing > O(1)

Bin Sort Fair Queueing (BSFQ) Highlights

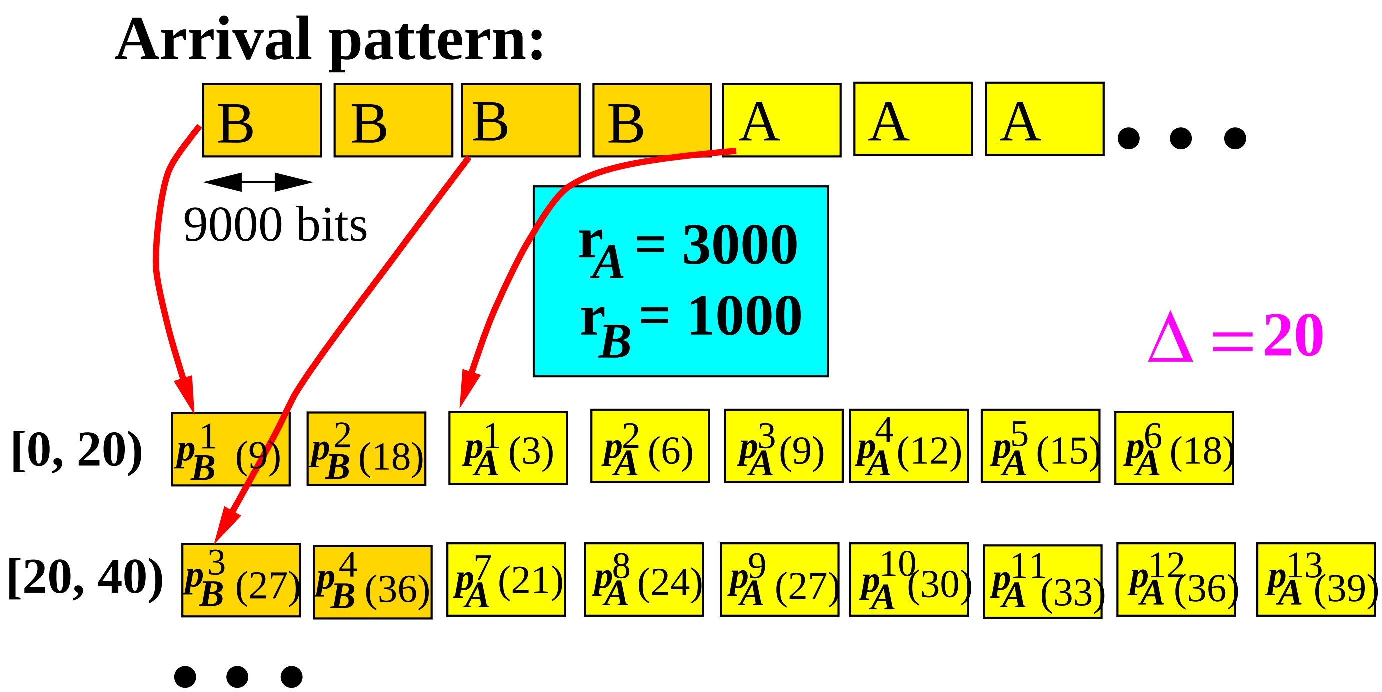
- Combines the strengths of frame-based and sorted-priority techniques
- BSFQ is a frame-based technique
- Unlike URR and SRR, BSFQ do not compute scheduling order when flow is added or removed
- BSFQ assigns time stamps (priority) to packets as in sorted-priority techniques
- But BSFQ do not use ordering/sorting operations
- BSFQ has $O(1)$ run time complexity for
  - Flow addition & removal
  - Packet processing

Bin Sort Fair Queueing Description



- Virtual time divided into disjoin intervals or time bins - interval size Δ
- Current bin = first non-empty bin
- System virtual time $\tau(t)$ = left interval time of current bin
- Packet time stamp: $T(p_f^j) = \max(\tau(A(p_f^j)), T(p_f^{j-1})) + \frac{\ell_f^j}{r_f}$
- Packet inserted into corresponding bin and serviced in FIFO order

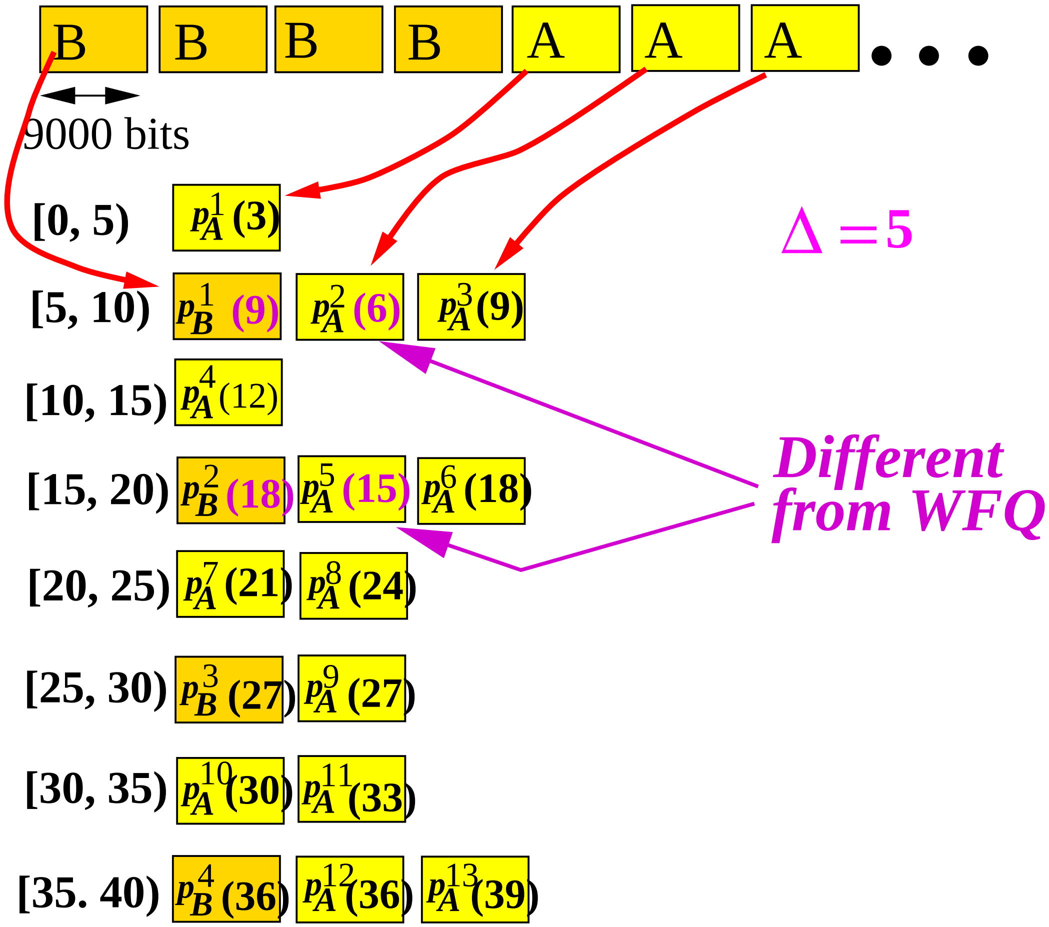
Bin Sort Fair Queueing Example 1: $\Delta = 20$



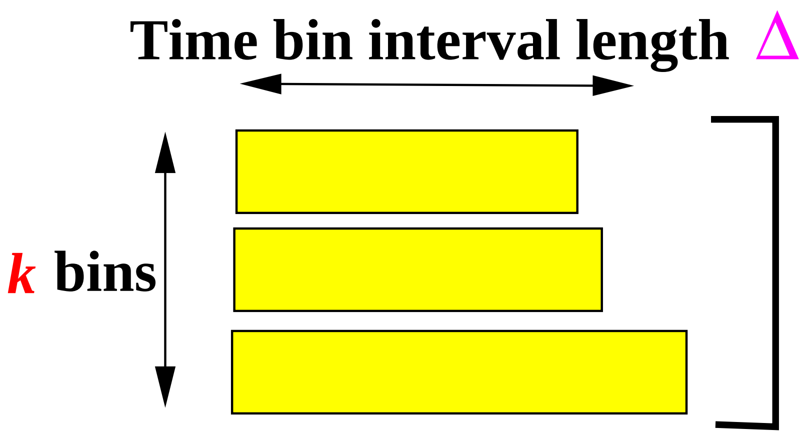
- Two flows A and B, reserved rate $r_A = 3000$ bps and $r_B = 1000$ bps
- $\Delta = 20$, fixed packet size 9000 bits

Bin Sort Fair Queueing Example 2: $\Delta = 5$

Arrival pattern:



BSFQ: Overheads and Effect of Δ



Max amount of data buffered in *k* bins:

$$k\Delta R + \sum_{\text{all flows } f} \ell_f^{max}$$

- Storage overhead increases linearly with the number of bins
- Δ = length of the virtual time interval of each (time) bin
- Using smaller values for Δ will:
  - Provide a better approximation for WFQ
  - Need a larger number of bins

BSFQ: Delay Guarantee

- The Guarantee Rate (GR) class schedulers: (Goyal et. al. '95)
  - Define: $GRC(p_f^j) = \max(A(p_f^j), GRC(p_f^{j-1})) + \frac{\ell_f^j}{r_f}$
  - Scheduler is in GR if departure time

$$\boxed{L(p_f^j) \leq GRC(p_f^j) + \beta}$$

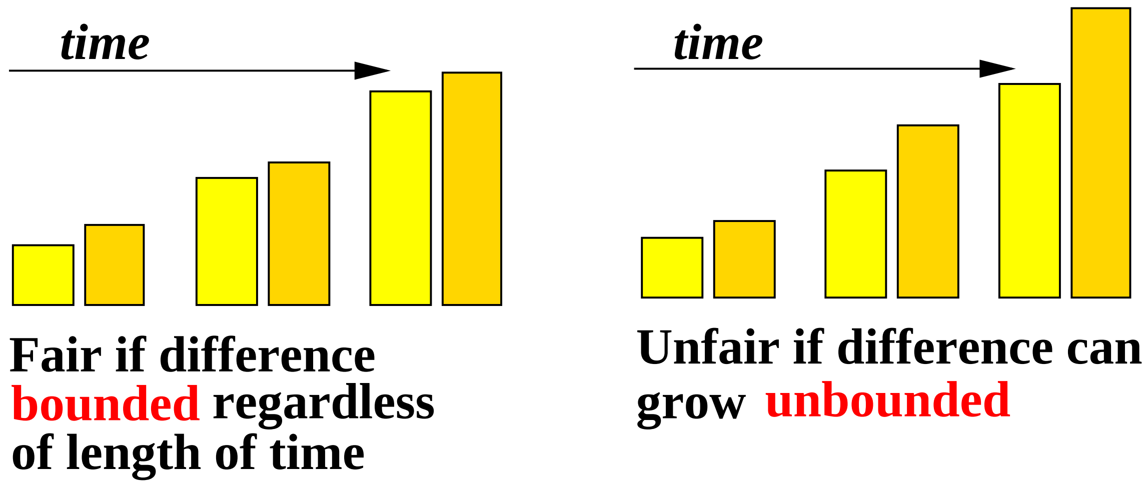
  for some constant β

- Theorem 1: Departure time of the BSFQ Scheduler is

$$\boxed{L_{BSFQ}(p_f^j) \leq GRC(p_f^j) + \Delta + \frac{\sum_{all\ flows\ f} \ell_f^{max}}{R}}$$

- R = output link's capacity
- ℓ_f^{max} = maximum packet size of flow f
- Δ is a (constant) design parameter of BSFQ

BSFQ: Fairness Guarantee

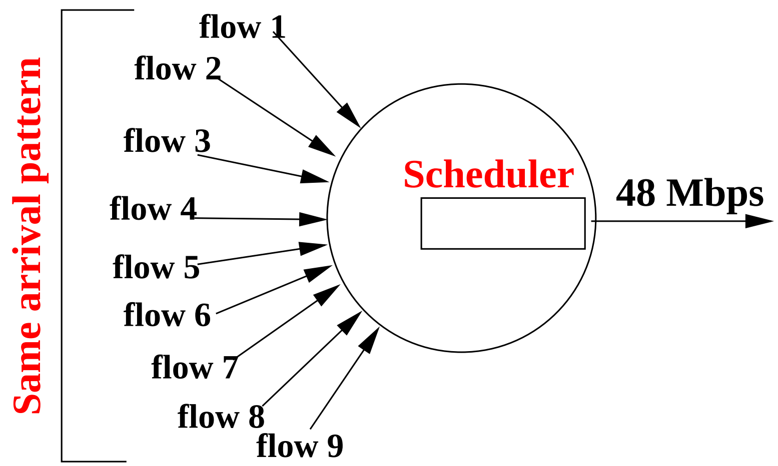


- Fairness condition used: (Golestani '95)
  - Scheduler is fair if difference of normalized work of two backlogged flows is bounded by a constant over any time interval
- Theorem 2: In BSFQ, if f and g are backlogged during $[t_1, t_2]$, then

$$\mid w_f(t_1, t_2) - w_g(t_1, t_2) \mid \leq 2\left(\frac{\ell_f^{max}}{r_f} + \frac{\ell_g^{max}}{r_g} + \Delta\right)$$

BSFQ: Performance Analysis

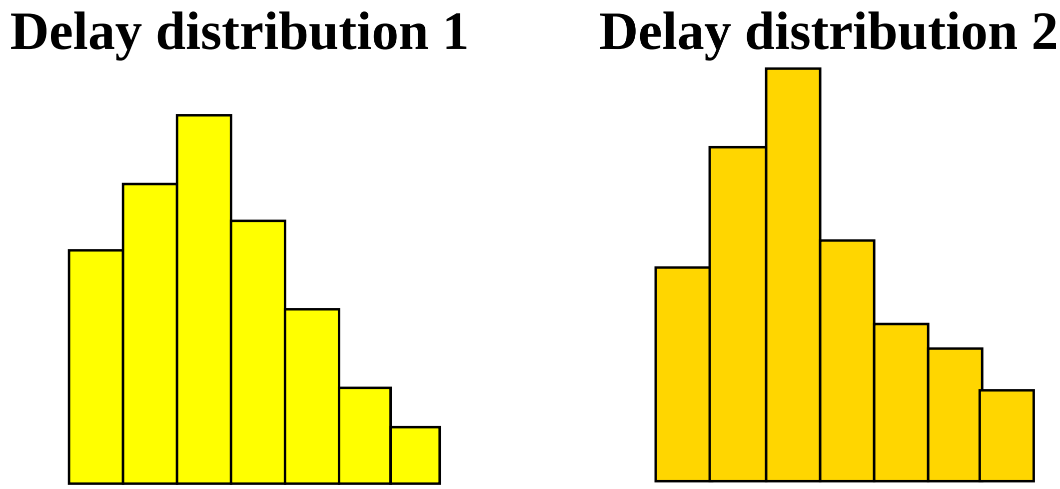
- Goal of the study:
  - Does BSFQ approximate WFQ ?
  - How good is the approximation ?
  - How does BSFQ compare to other approximation with similar overhead ?
- Approach:



Use the **same arrival** **pattern**

Compare packet delay distribution using different schedulers

Comparing two probability distributions



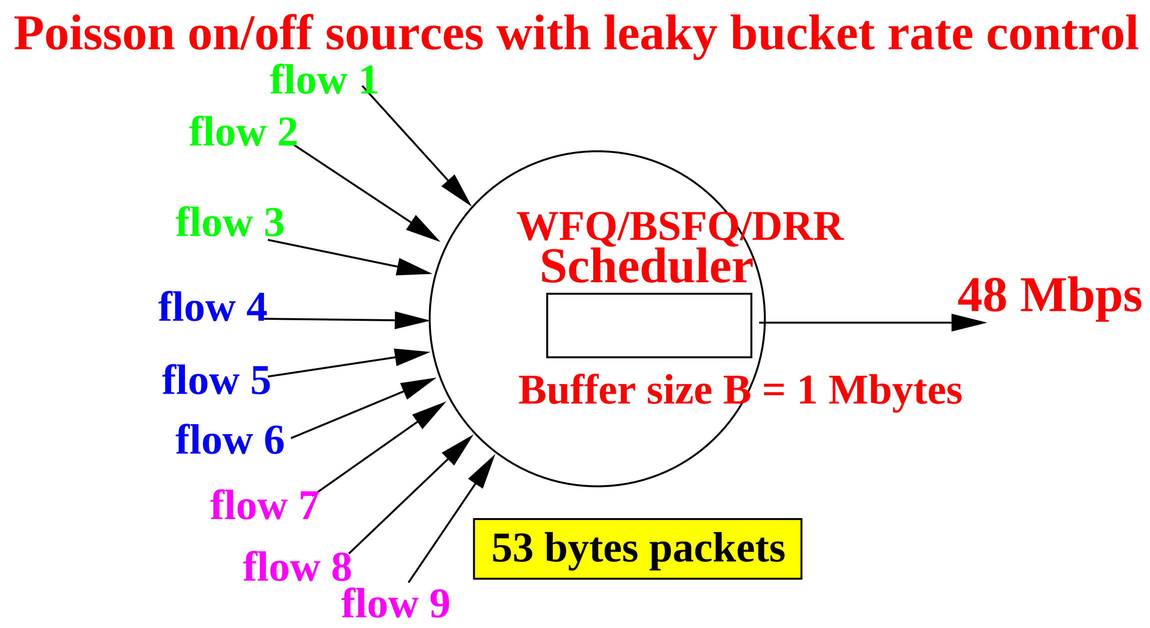
- The total variance is often used to measure difference between two probability distributions
- Given two probability distributions $p(x)$ and $\pi(x)$ over the same finite set Ω, the total variance is:

$$TV(p, \pi) = \frac{1}{2} \sum_{x \in \Omega} \mid p(x) - \pi(x) \mid$$

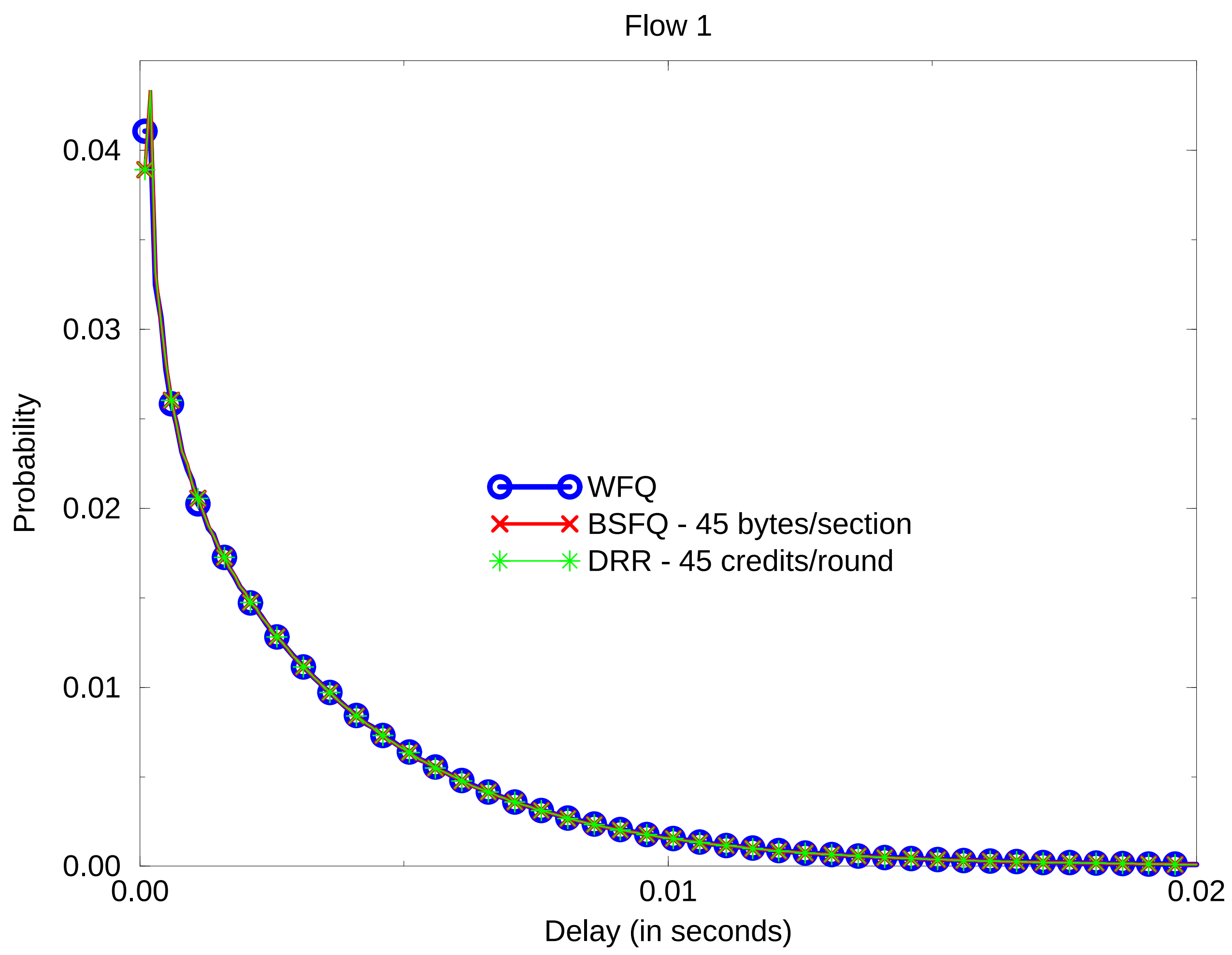
A Comparison Study

- Schedulers used in comparison study
  - WFQ - basis for the comparison study
  - DRR - has same runtime complexity as BSFQ for packet scheduling and flow addition/removal
- Setting parameters in BSFQ and DRR
  - Parameter Δ in BSFQ and quantum size Q in DRR affect runtime complexity and scheduler's performance
  - Criteria for setting Δ and Q:
    - Computation effort in each round (in both schemes) approximately equal
    - Pick Δ and Q such that BSFQ and DRR will transmit approximately the same number of packets in one round

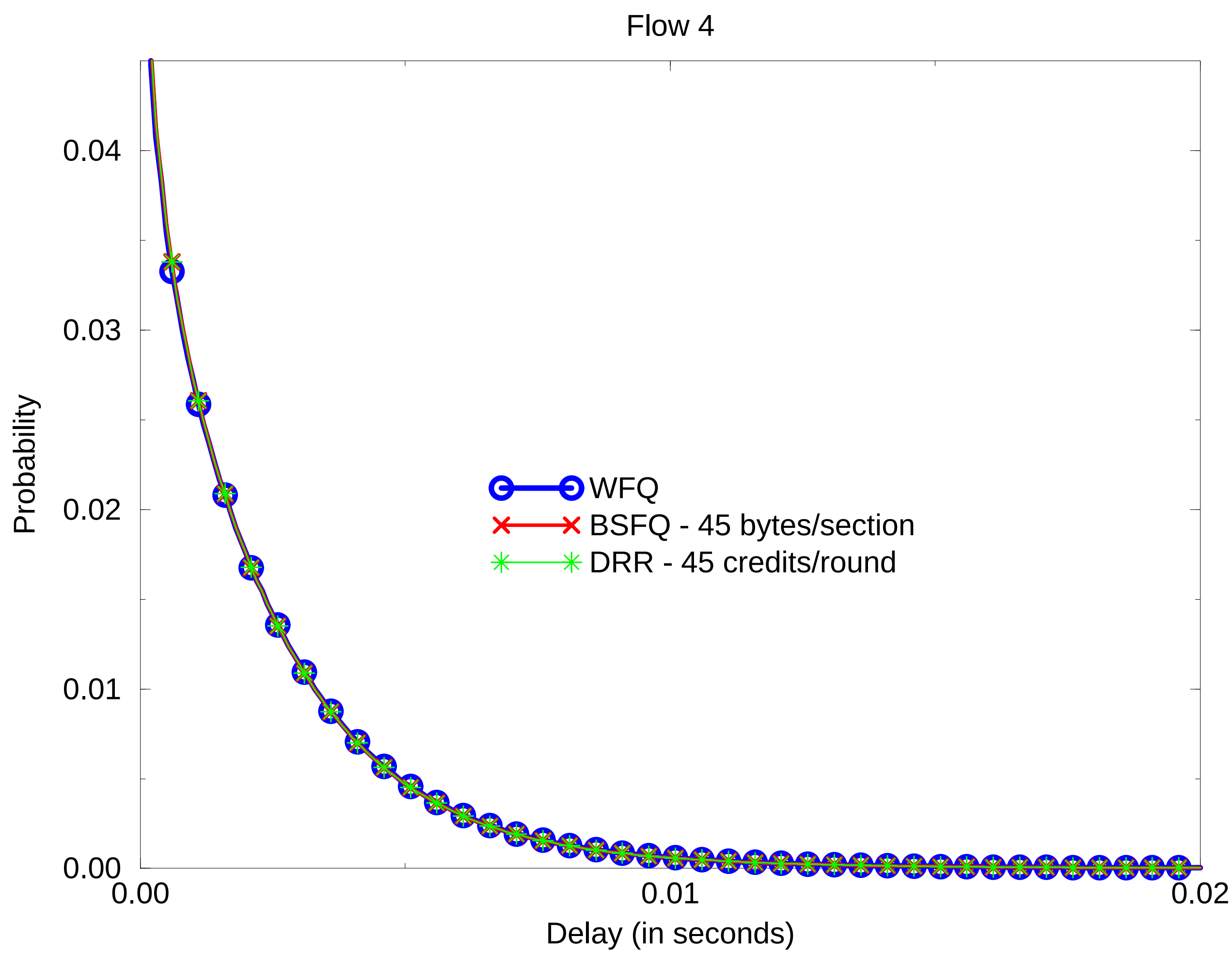
The Simulation Model



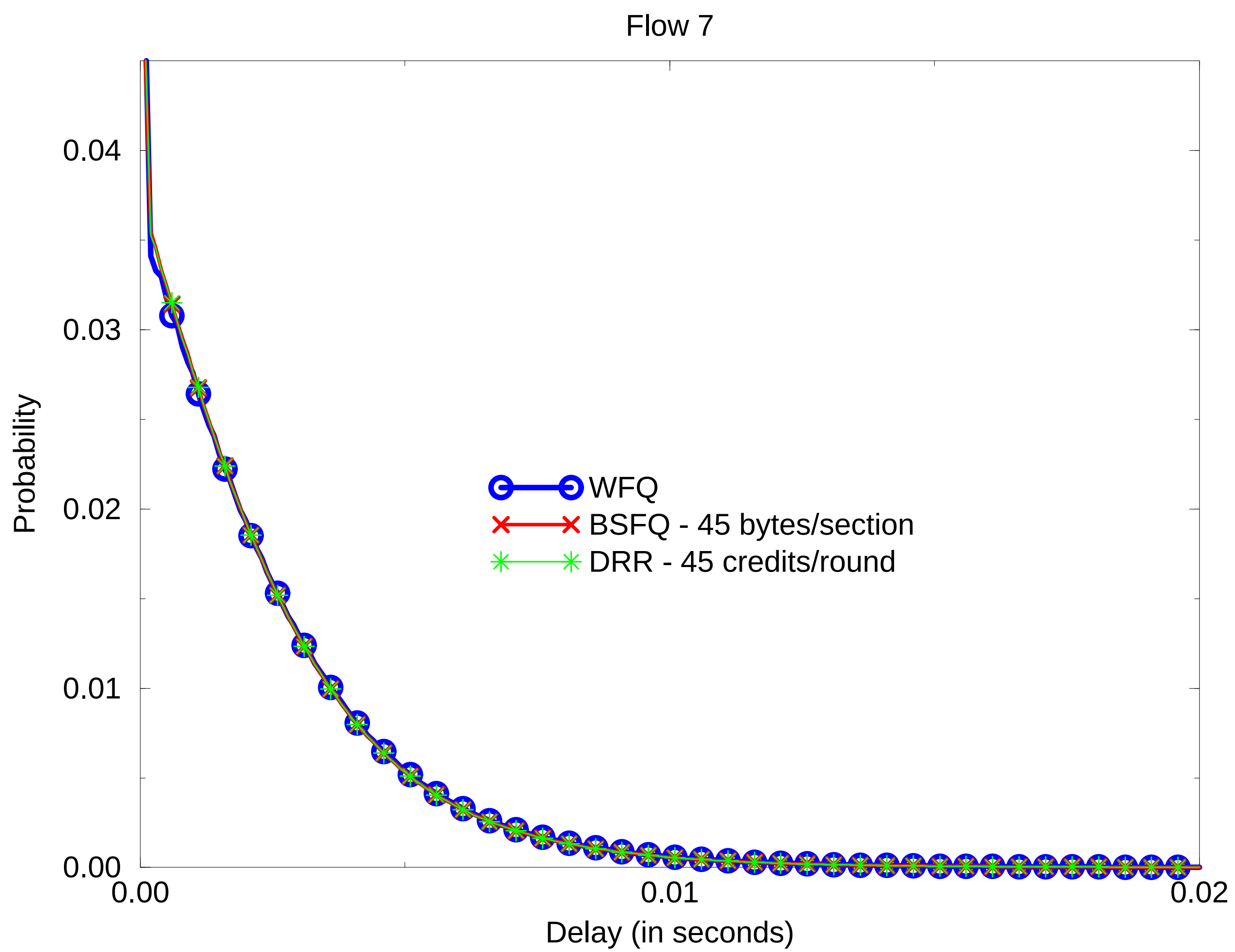
	Source rate		Leaky Bucket	
Flow	R_{peak}	R_{avg}	σ	ρ
1,2,3	16Mbps	2Mbps	50KB	2Mbps
4,5,6	25Mbps	5Mbps	100KB	5Mbps
7,8,9	40Mbps	8Mbps	200KB	8Mbps



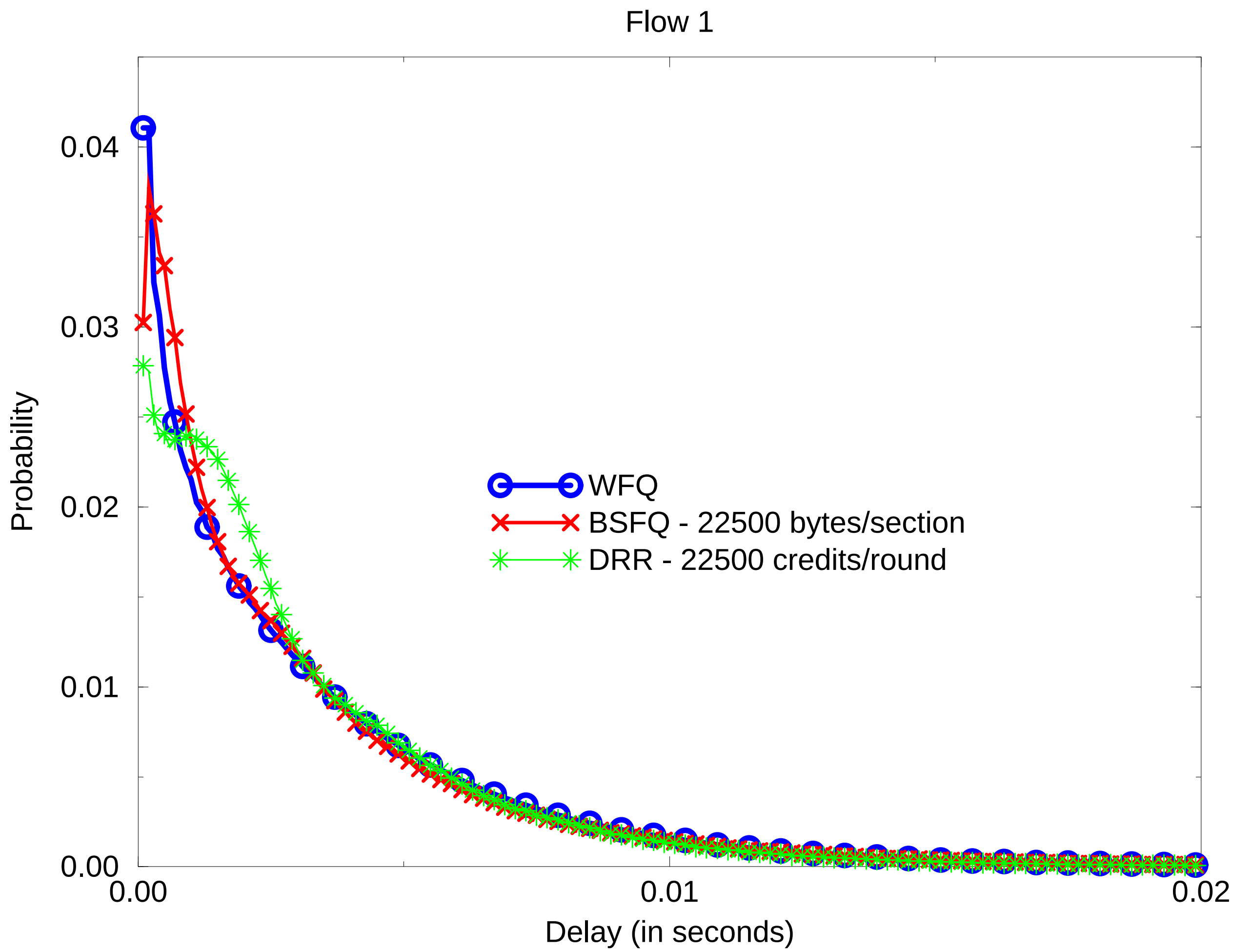
Delay distribution of Flows 1, 2 and 3, 45 bytes/round



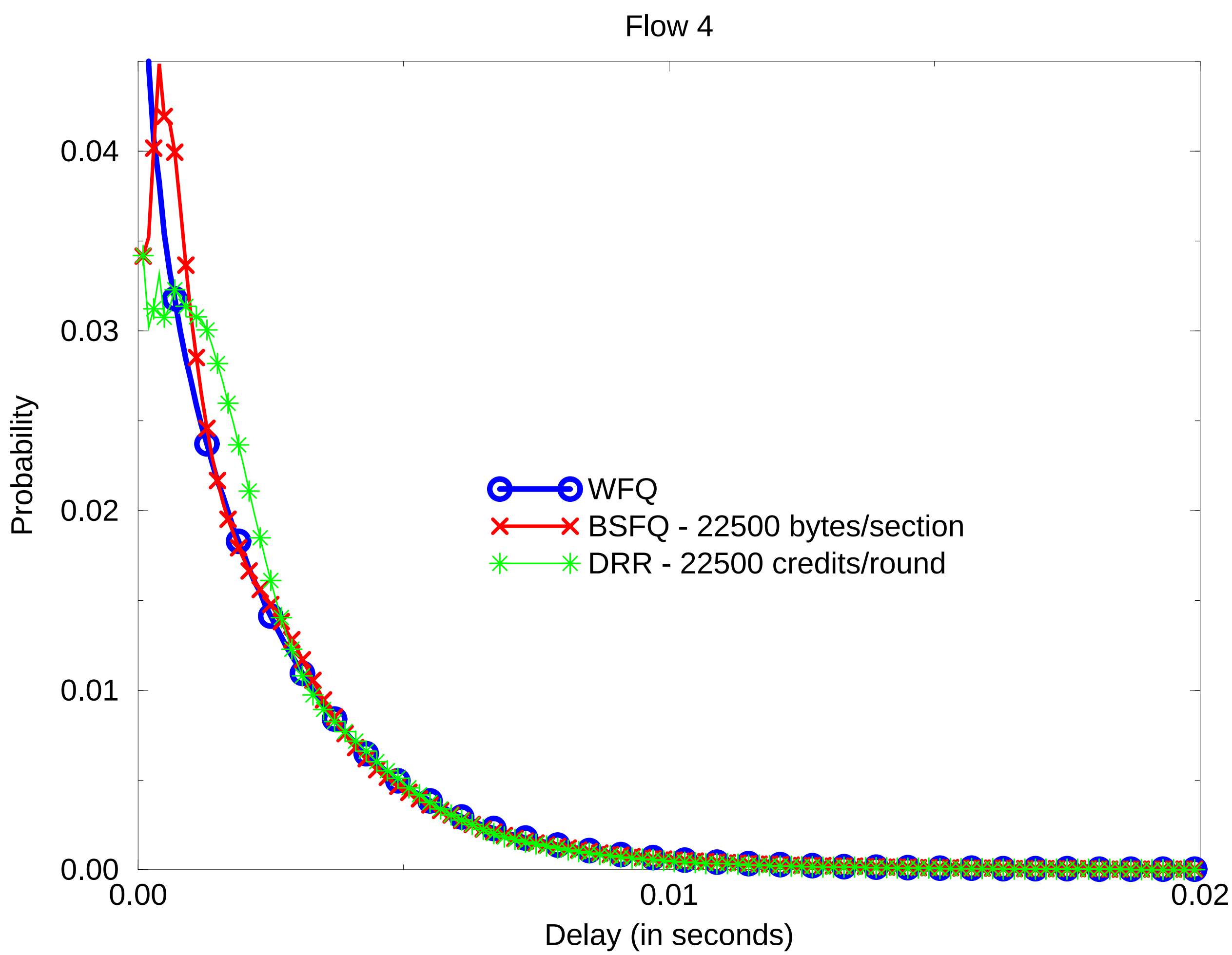
Delay distribution of Flows 4, 5 and 6, 45 bytes/round



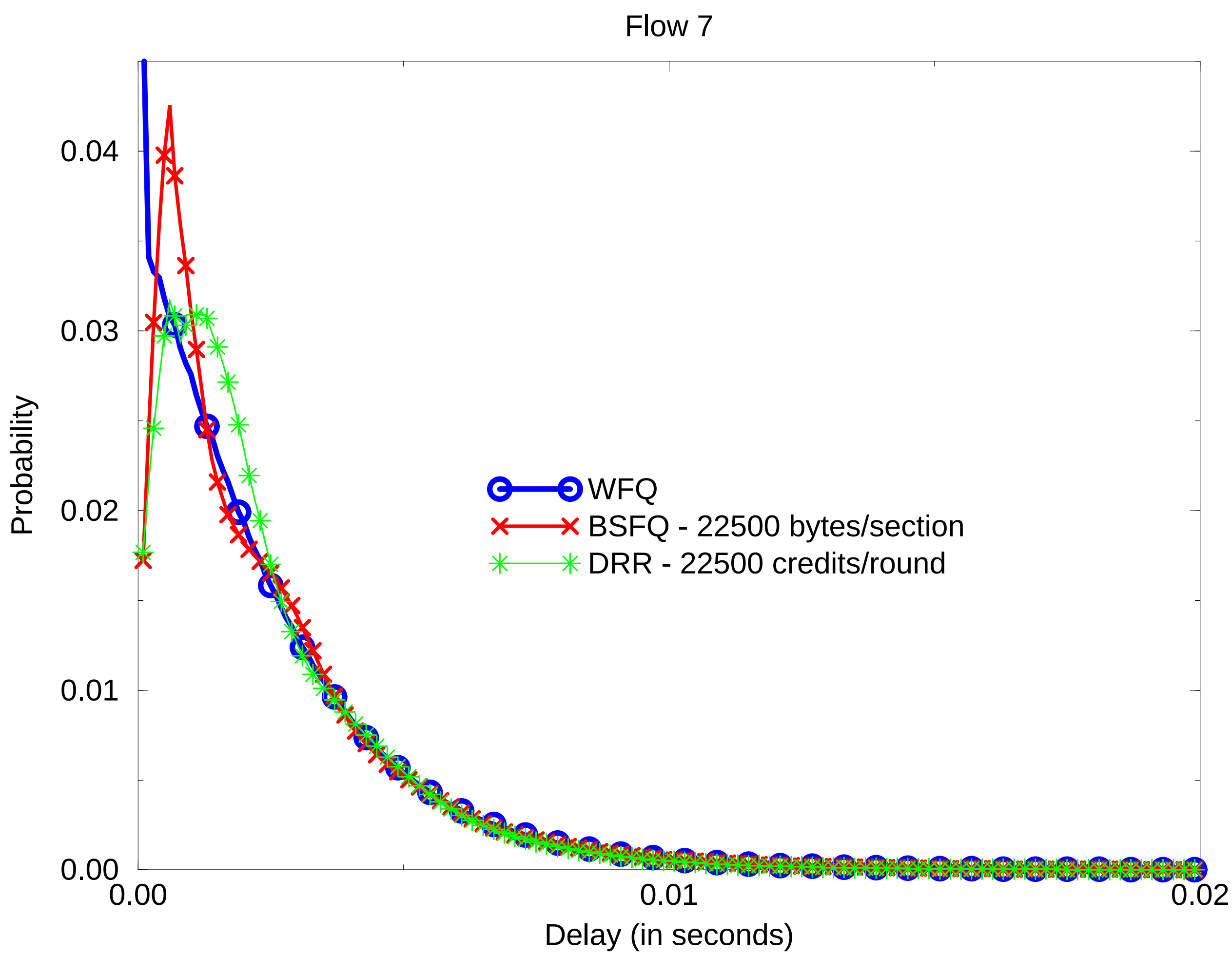
Delay distribution of Flows 7, 8 and 9, 45 bytes/round



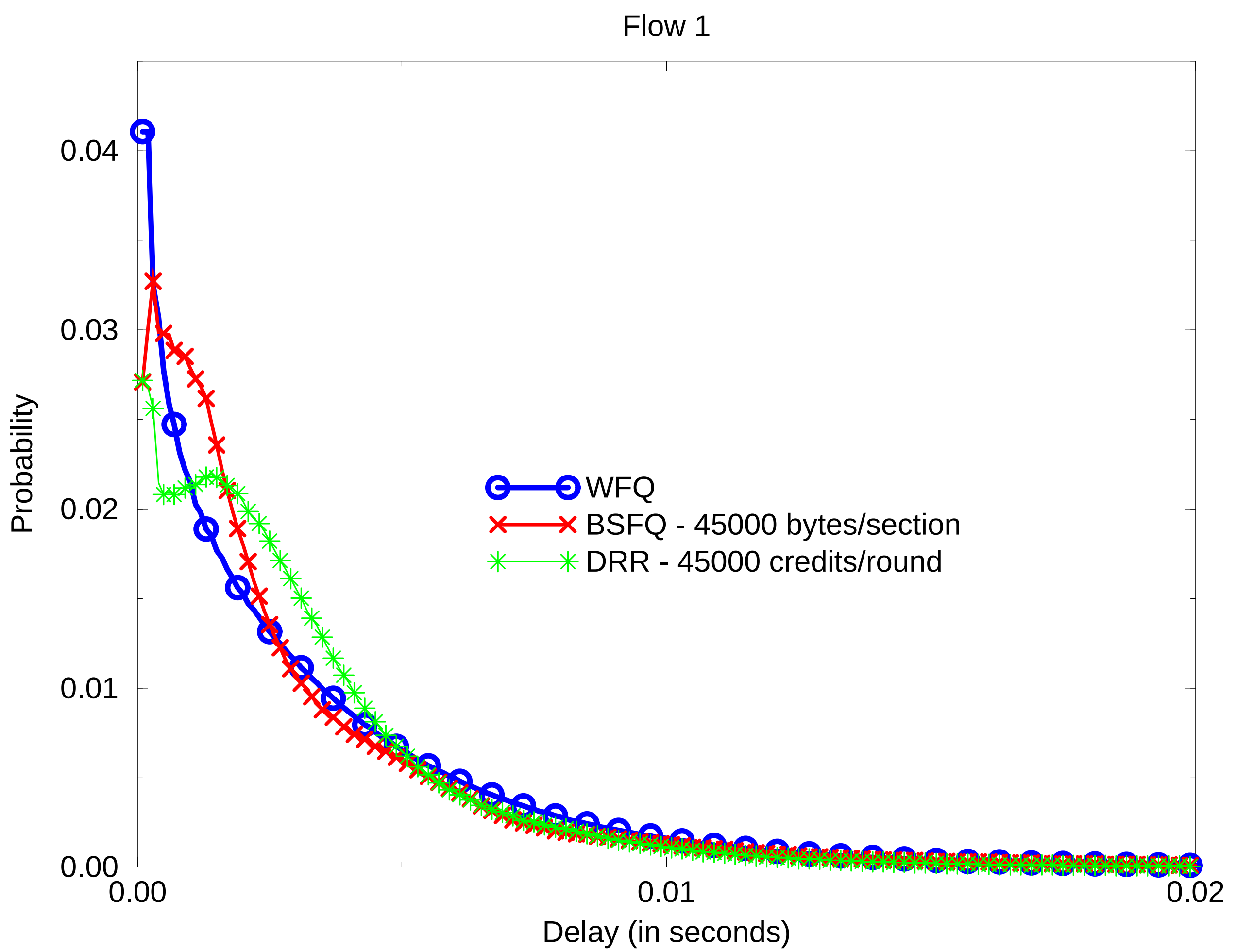
Delay distribution of Flows 1, 2 and 3, 22500 bytes/round



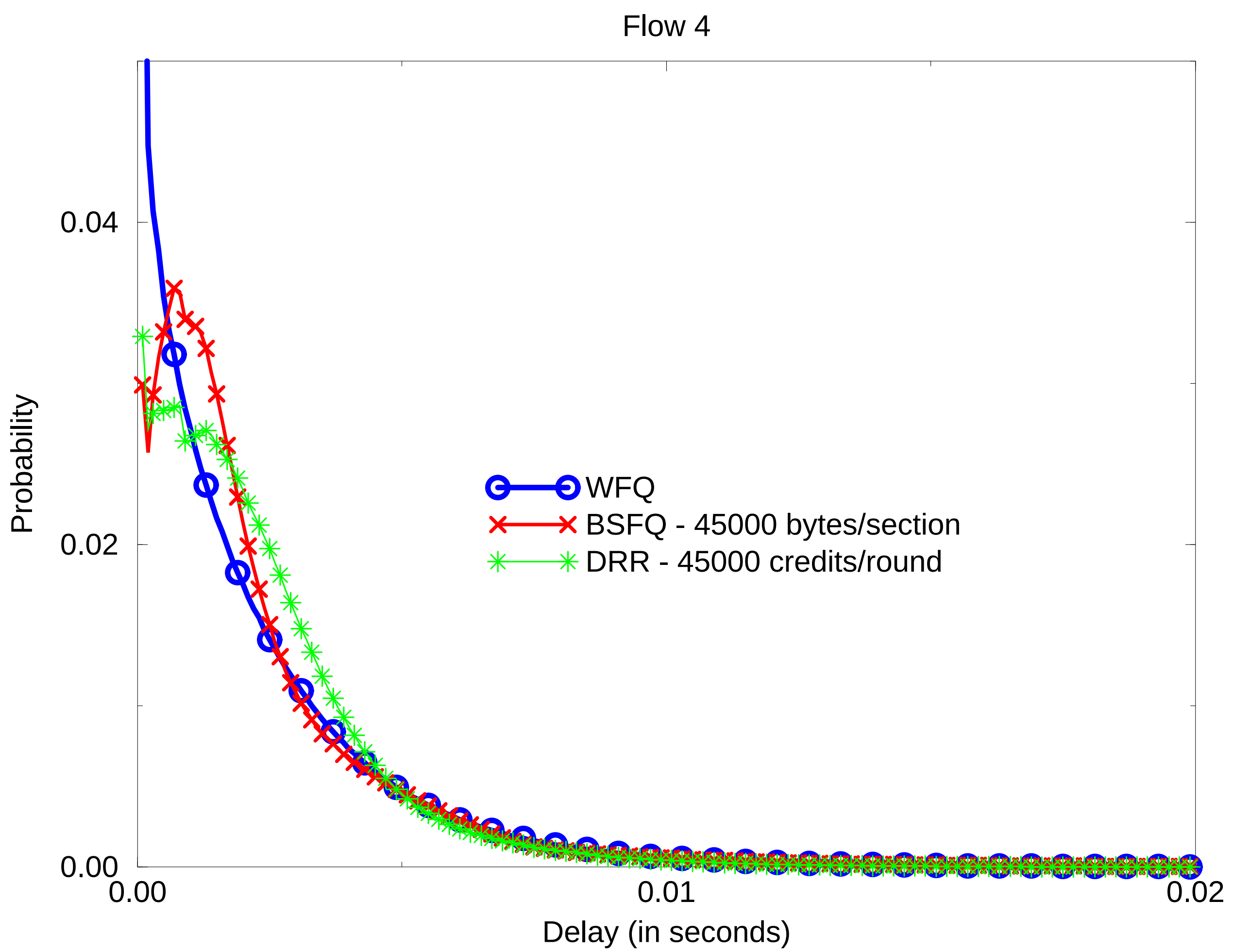
Delay distribution of Flows 4, 5 and 6, 22500 bytes/round



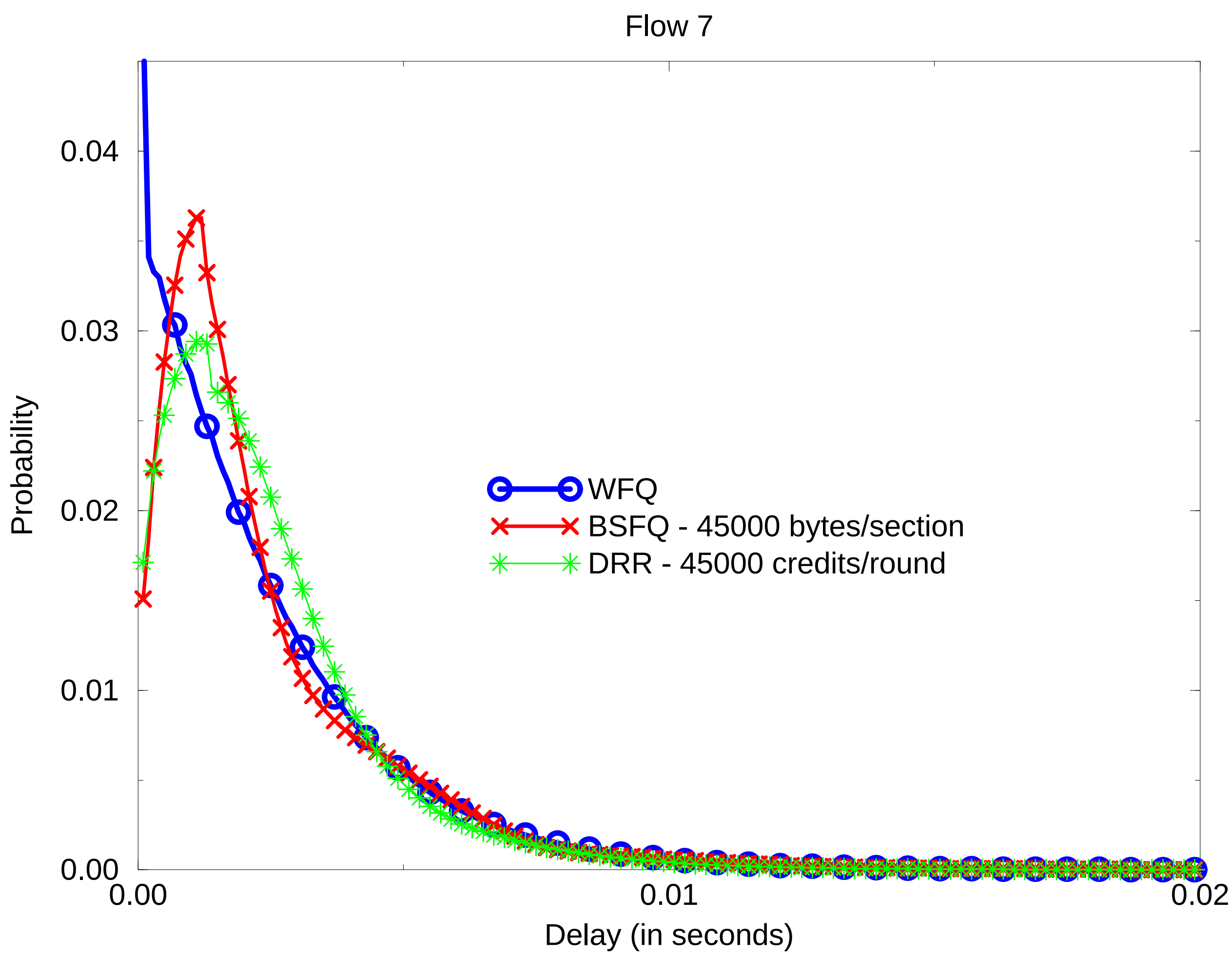
Delay distribution of Flows 7, 8 and 9, 22500 bytes/round



Delay distribution of Flows 1, 2 and 3, 45000 bytes/round



Delay distribution of Flows 4, 5 and 6, 45000 bytes/round



Delay distribution of Flows 7, 8 and 9, 45000 bytes/round

Summary of the experimental results

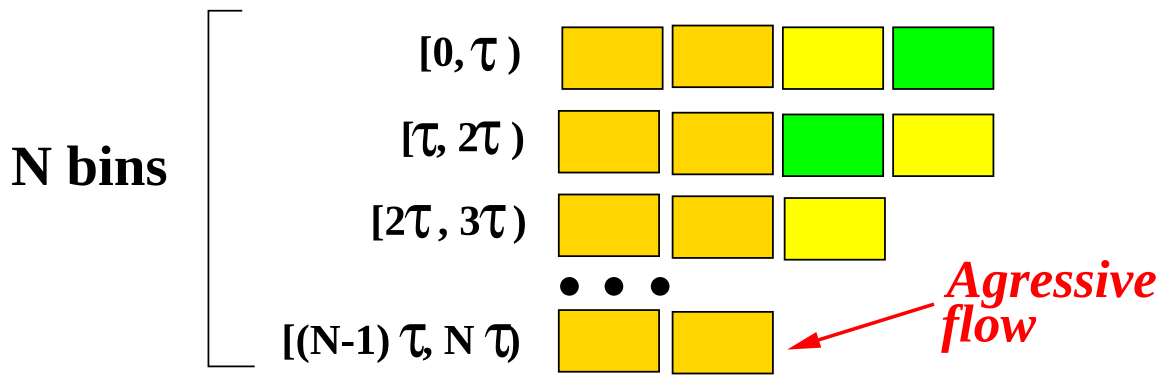
Total variance distances between BSFQ and WFQ

Parameter Setting	Flows 1,2,3	Flows 4,5,6	Flows 7,8,9
45 bytes/bin	0.0050	0.0048	0.0069
22500 bytes/bin	0.0430	0.0604	0.0620
45000 bytes/bin	0.0883	0.1028	0.0998

Total variance distances between DRR and WFQ

Parameter Setting	Flows 1,2,3	Flows 4,5,6	Flows 7,8,9
45 bytes/round	0.0048	0.0049	0.0067
22500 bytes/round	0.0738	0.0882	0.0739
45000 bytes/round	0.1180	0.1255	0.1095

Built-in Buffer Management Capability



- Packets of non-compliant flows will be assigned a bin number larger than N (number of bins)
- Such packets are dropped
- BSFQ has a built-in buffer management function

Buffer Management Example

- Introduce non-compiant flows:

	Source rate		Leaky Bucket	
Flow	R_{peak}	R_{avg}	σ	ρ
1,2,3	16Mbps	2Mbps	50KB	2Mbps
4,5,6	25Mbps	5Mbps	100KB	5Mbps
7,8,9	50Mbps	10Mbps	200KB	8Mbps

- Total data rate exceeds link capacity, buffer size 1 MByte
- Result:

	Fraction of packets dropped		
	Flow 1	Flow 4	Flow 7
BSFQ	0%	0%	9.89%
DRR	5.43%	4.52%	6.38%

Concluding Remarks

- BSFQ partitions virtual time into bins of equal size
- Assign time stamp to packets according to sorted-priority technique
- BSFQ combines the strengths of frame-based and sorted-priority scheduling methods
- BSFQ has O(1) runtime complexity for packet scheduling and flow addition/removal
- $\Delta \longrightarrow 0$, BSFQ becomes SCFQ
- $\Delta \longrightarrow \infty$, BSFQ becomes FIFO
- BSFQ has a built-in buffer management component