A Survey of Scheduling Methods

Raj Jain

Raj Jain is now at
Washington University in Saint Louis
Jain@cse.wustl.edu
http://www.cse.wustl.edu/~jain/

- Goals
- Metrics
- Classification
- Scheduling Methods

Scheduling: Goals

- Sharing bandwidth
- Sharing bandwidth fairly
- Meeting bandwidth guarantees (min and max)
  $\Rightarrow$ Provide isolation between users
- Meeting loss guarantees (multiple levels)
- Meeting delay guarantees (multiple levels)
- Reducing delay variation

Goals (Cont)

- Guarantees require call admission control, policing, shaping, drop policies, buffer allocation, and scheduling
- These issues are, therefore, related.
- For example, zero-loss can be obtained by allocating PCR (but no multiplexing gain).

Scheduling: Methods

- FCFS
- Round Robin
- Priority Queueing
- Priority Queueing with Windows
- Generalized Processor Sharing (GPS)
- VirtualClock
- Weighted Fair Queueing (WFQ), WF2Q, WF2Q+
- Self-Clocked Fair Queueing (SCFQ)
- Stop and Go
- Rate Controlled Service Descipline (RCSD)

Scheduling Metrics

- Complexity of enqueue+dequeue processes
- Fairness: If two flows are backlogged, difference between their weighted throughputs is bounded
- Complexity of adding and releasing connections (or changing quotas in ABR)
- Delay bounds should not depend upon behavior of other flows, number of other flows, reservations of other flows

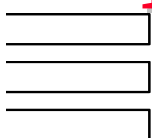
Scheduling Classification

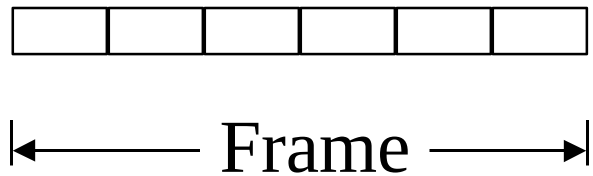
- Work conserving vs nonconserving
- Sorted priority vs frame based
- Control vs accomodate distortion

Work Conserving vs Nonconserving

- Conserving: Server not idle if there is work.
  - Produces lower average delay, higher delay var.
  - Produces high total throughput
  - Examples: GPS, WFQ, VirtualClock
- Nonconserving: Better for multiple hops
  - May produce lower worst case end-to-end delay
  - May produce higher network throughput
  - Reshaping at every hop $\Rightarrow$ additive hop delays
  - Examples: Stop & Go

Sorted Priority vs Frame Based





- Sorted Priority: Virtual time for each flow/packet
  - Generally has a O(log(V)) complexity
  - Examples: VirtualClock, WFQ
- Frame Based: Time split into fixed/variable frames
  - Each flow reserves the time per frame
  - Delay and bandwidth allocations are dependent
    - Examples: Stop and Go uses constant frame size. DRR, WRR allow variable frame size

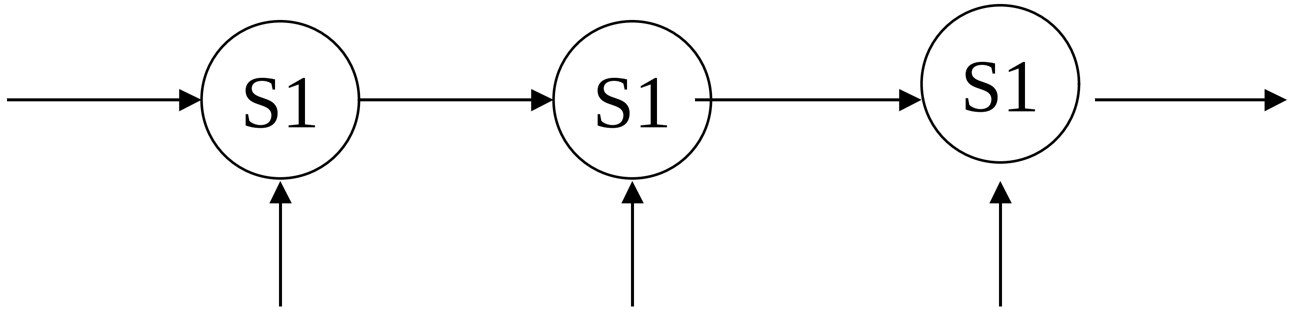
Control Distortion vs Accomodate distortion

- Burstiness of the traffic increases along the path ⇒ Need more resources
- Control Distortion: Reshape at each hop ⇒ Non-work conserving
  - Example: Stop and Go, HRR, Jitter EDD
- Accomodate Distortion: Do not reshape
  - Example: VirtualClock, Fair Queueing, GPS, Delay EDD

FCFS

- Unfair
- No isolation among users

Round Robin



- Parking lot problem: Distance sources get lower
- Classify incoming traffic into flows (Src-Dest pairs)
- Round-robin among flows
- Known Problems:
  Ignores packet length ⇒ Fair Queueing
- Ref: Nagle

Priority Queueing

- Also known as head of line (HOL)
- Priority 0 through n-1
- Priority 0 is always serviced first.
- Priority i is serviced only if 0 through i-1 are empty
- Highest priority has the lowest delay, highest throughput, lowest loss
- Lower priority classes may be starved if higher priority are overloaded

Priority Queueing with Windows

- Maximum n_i packets from ith priority during a single round
- Come back to higher priority unless n_i packets have been served
- Guarantees non-starvation but increases the delay for higher priorities
- Large $n_i \Rightarrow$ Priority queueing.
  Small $n_i \Rightarrow$ Round robin

Priority with Windows (Cont)

- n_i's determine min bandwidth allocations and delays
- Quantitative relationships between n_i and delays or loss not provided.
- VLSI design implemented
- Refs: El-Gebaly et al and Sabaa et al.

VirtualClock

- Goals: Provide average reserved throughput R_i b/s
- Provides isolation between users
- Upon packet arrival:
  - $VirtualClock_i = Max\{wall\ clock\ time, VirtualClock_i\}$
  - Timestamp the packet with $VirtualClock_i$
  - $VirtualClock_i = VirtualClock_i + packet\ size/R_i$
- Transmit packets in order of increasing timestamps

VirtualClock (Cont)

- Possible to implement with one timestamp per flow rather than one per packet
- Known Problems: Flows do not accumulate credits
  - Flows using idle bandwidth are penalized later $\Rightarrow$ Virtual Clock is Unfair $\Rightarrow$ Several proposed fixes, e.g., Time-shift scheduling
  - No CAC policy $\Rightarrow$ no delay bounds
  - Need to implement priority queues $\Rightarrow$ O(log V) complexity, V=# of VCs
- Refs: Zhang, Srinivasan et al, Stilidias and Varma, Cobb et al

Generalized Processor Sharing

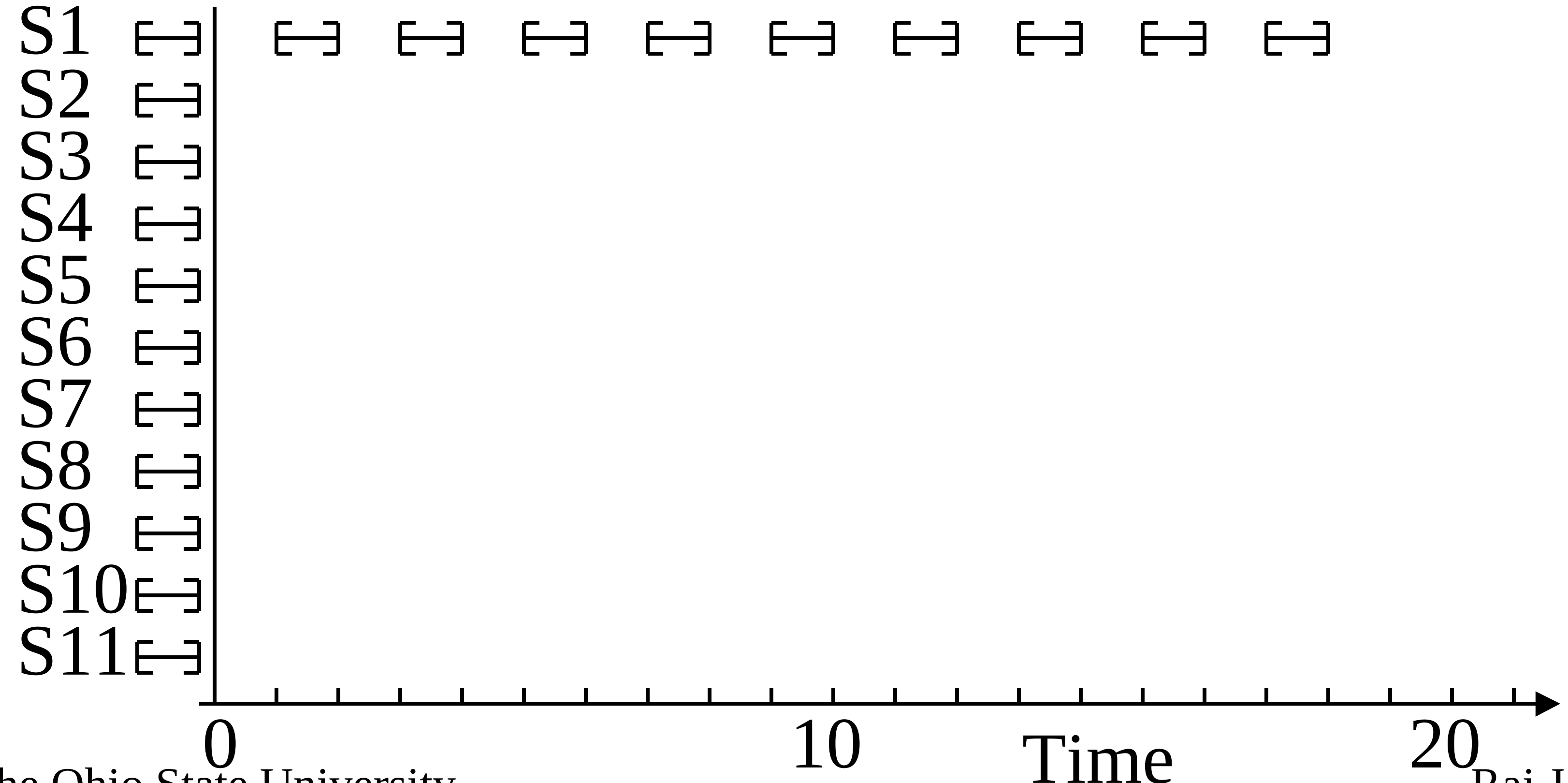
- Idealized policy to split bandwidth
- Each user has a fraction si of the bandwidth
- All unused bandwidth is allocated in proportion to the fraction ϕ_i
- At time t, ith active user gets a fraction ri

$$r_i = \phi_i / \Sigma_{\text{active } j}\, \phi_j$$

- Weighted round-robin with infinitely small service quantum

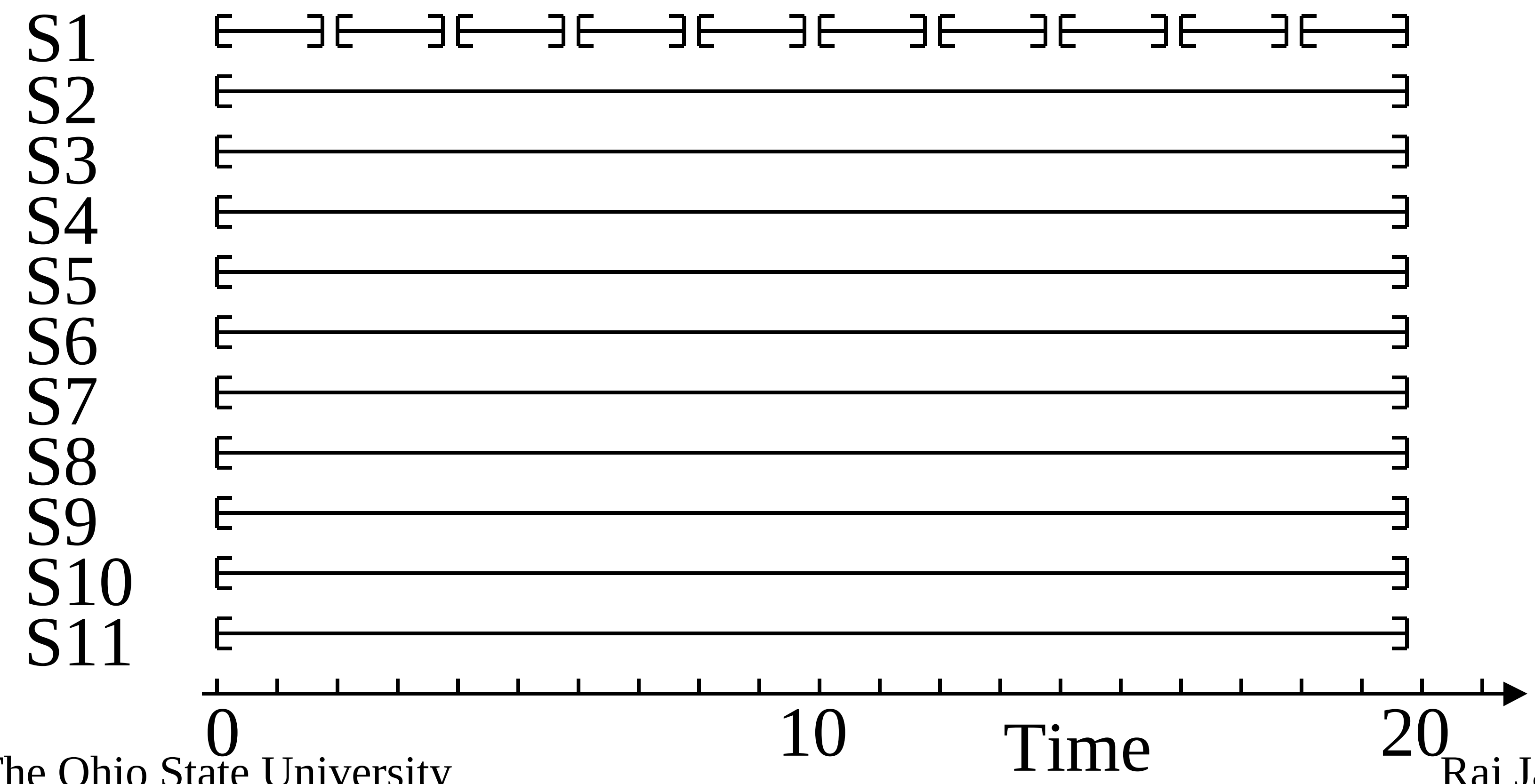
GPS Example: Arrivals

- Eleven Sources. First source gets 0.5. Other 10 sources get 0.05 each. First source sends 10 cells. 2-11 send one each at t=0.



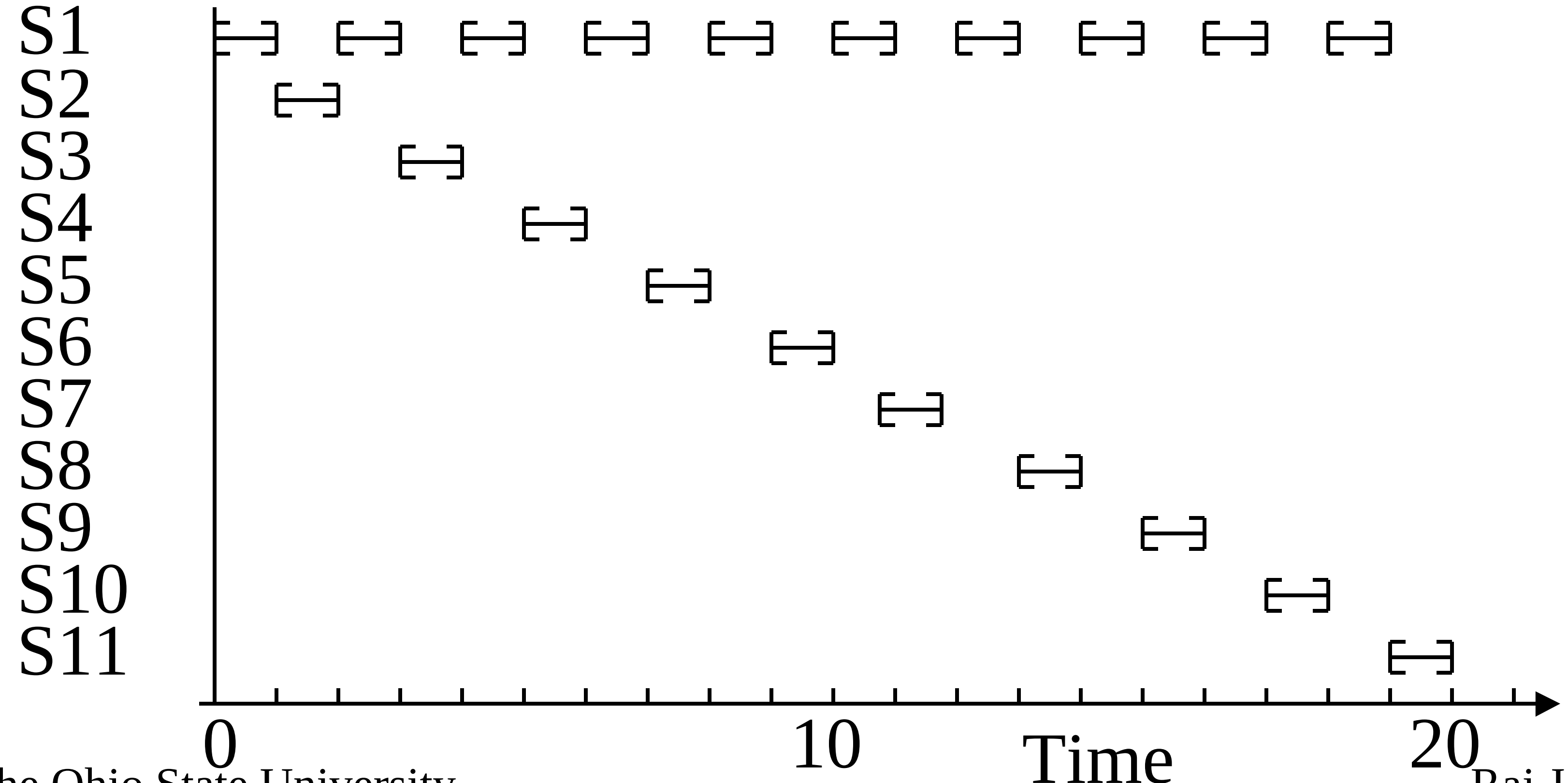
GPS Example: Service

- Each cell of the first source takes 2 units of time. Sources 2-11 take 20 units each.



Weighted Fair Queueing

- Approximates bit-by-bit round robin. Compute GPS finish time and schedule the packet with the smallest finish time.



Weighted Fair Queueing (WFQ)

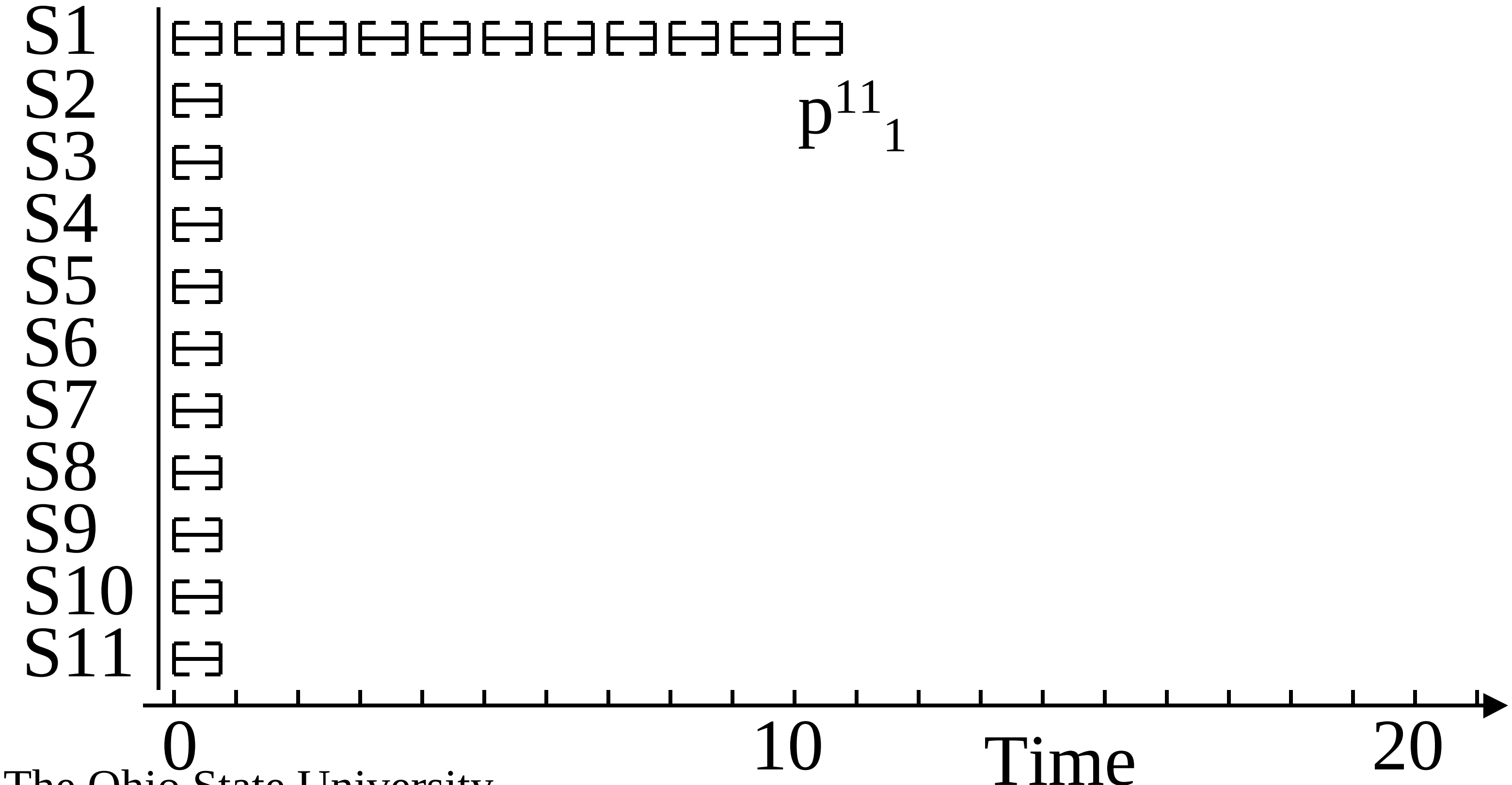
- Basis of IETF's integrated services
- Naive implementation requires O(log(m)), m=# of packets
- Keshav's implementation requires O(log(V)), V=# of flows
- **Known Properties**: CAC and End-to-end delay bounds have been derived for leaky-bucket shaped sources
- Parekh and Gallagher showed that leaky bucket +FQ $\Rightarrow$ delay guarantees

WFQ (Cont)

- **Known Problems**:
  - Need large bandwidth reservation to get small delay bound.
  - Complex to implement.
  - Packets can be serviced much earlier than GPS. Can introduce significant unfairness over GPS.
- **Refs**: Demers et al, Keshav, Srinivasan et al

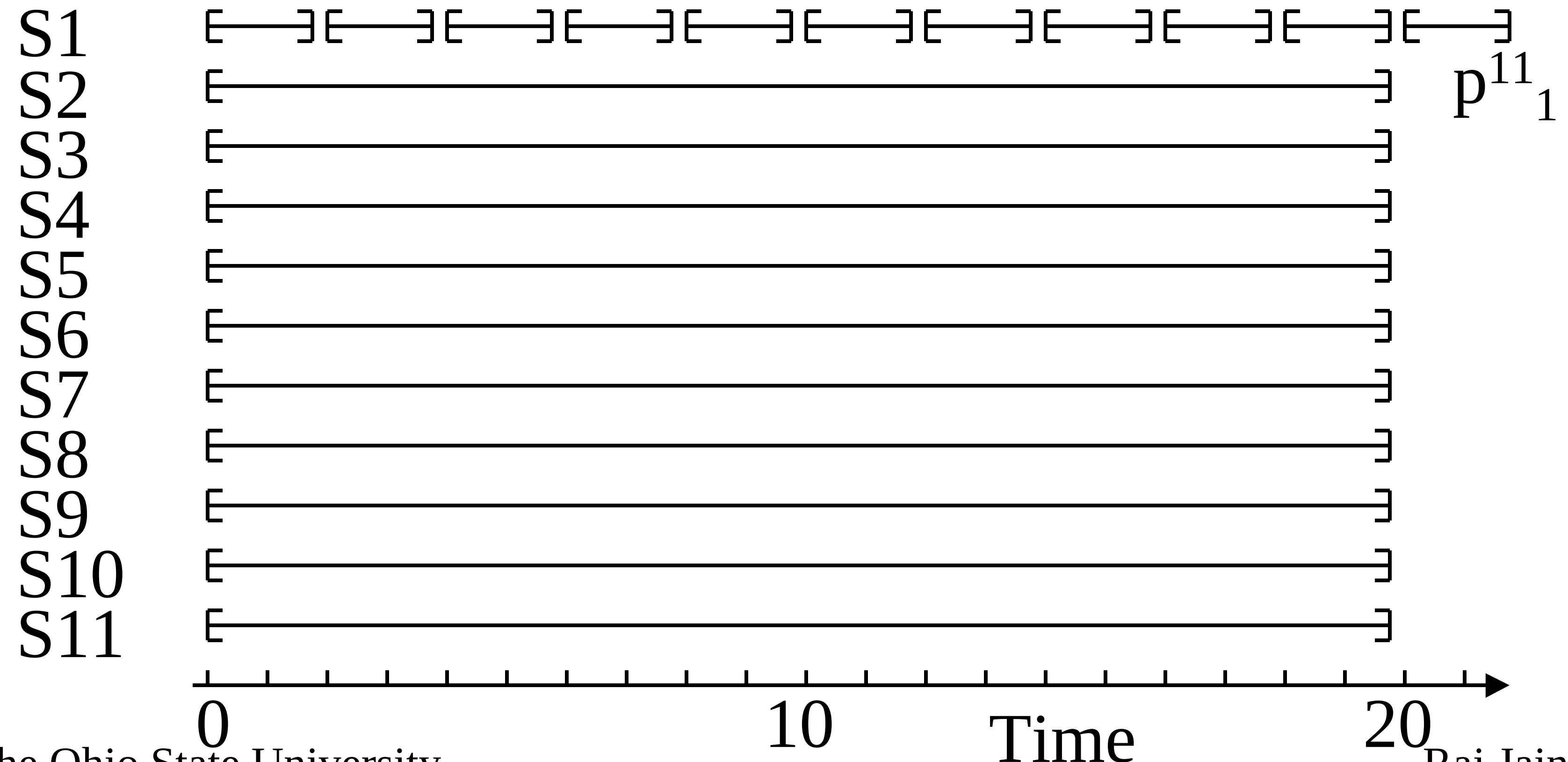
GPS Example 2: Arrivals

- Eleven Sources. First source gets 0.5. Other 10 sources get 0.05 each. First source sends 11 cells. 2-11 send one each at t=0.



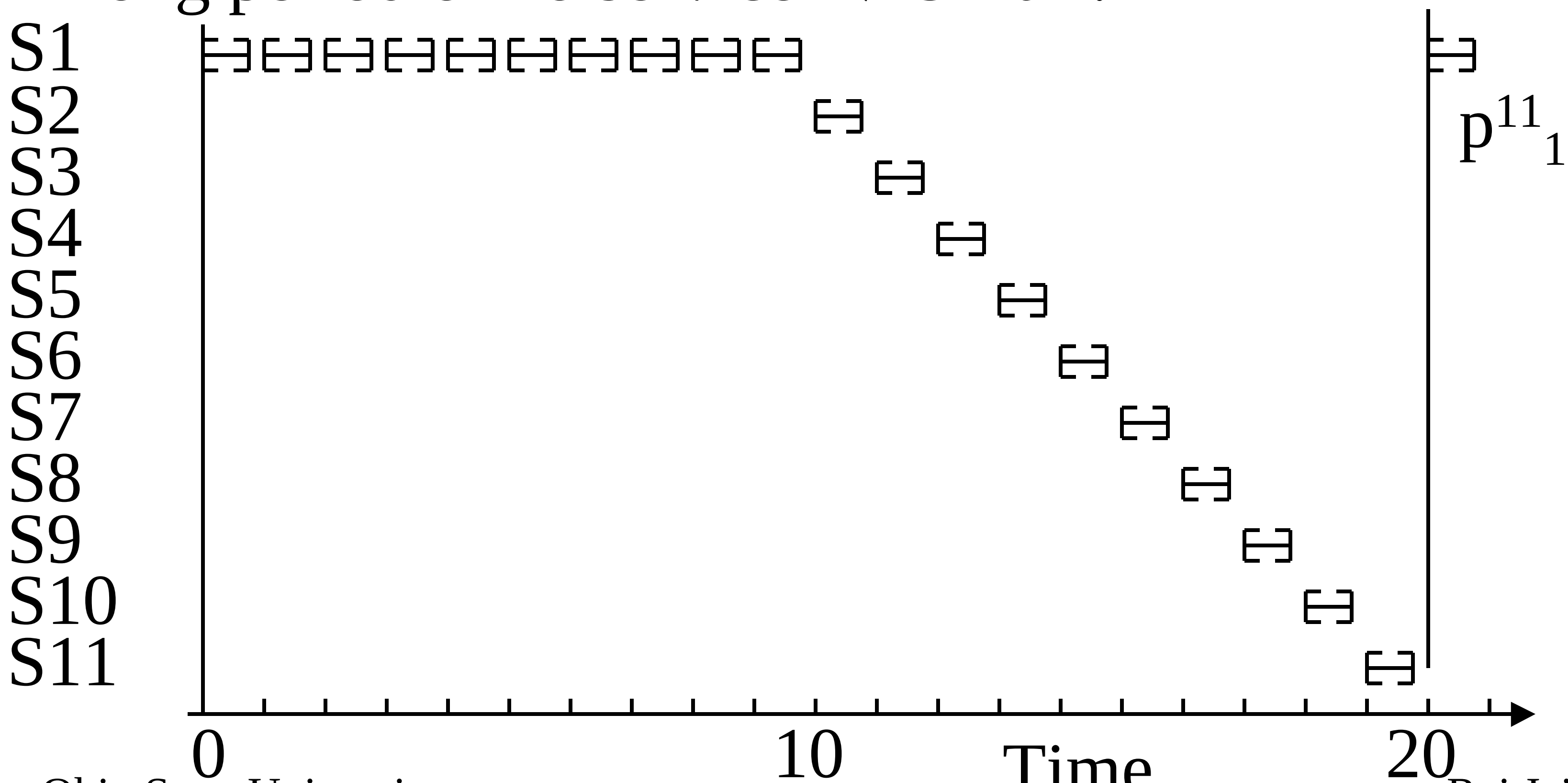
GPS Example 2: Service

- Each cell of the first source takes 2 units of time. Sources 2-11 take 20 units each.



WFQ: Service

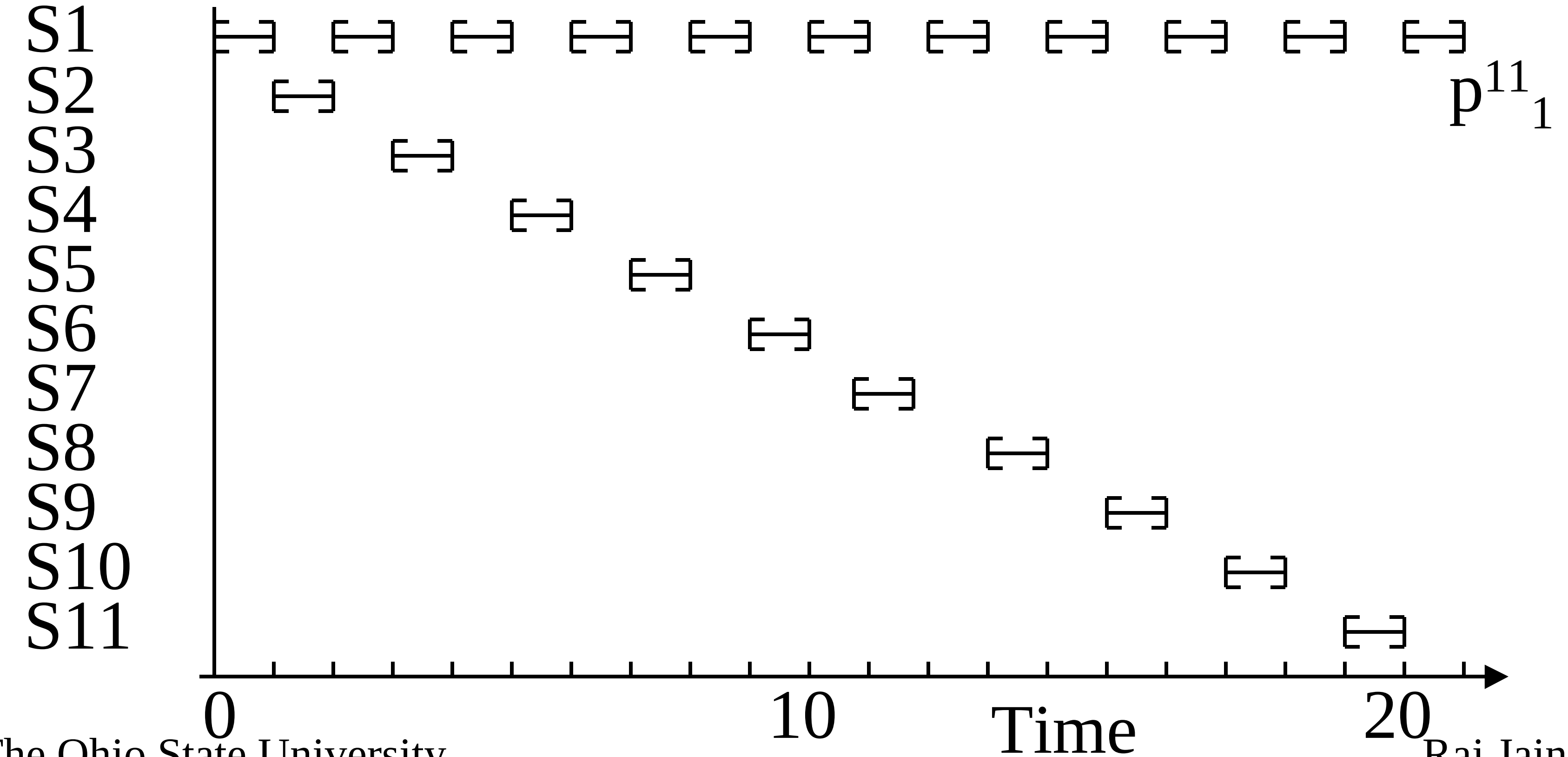
- Packets finish at the same time or earlier than GPS. Some packets finish much earlier. Long period of no service ⇒ Unfair.



Worst Case Fair Weighted Fair Queuing (WF2Q)

- WF2Q fixes the unfairness problem in WFQ.
  - WFQ: Among packets waiting in the system, pick one that will finish service first under GPS.
  - WF2Q: Among packets waiting in the system that have started service under GPS, select one that will finish first under GPS.
- WF2Q provides service close to GPS (difference in packet service time bounded by max. packet size).
- WF2Q+ is an simpler implementation of WF2Q
- Refs: Jon Bennett, Hui Zhang.

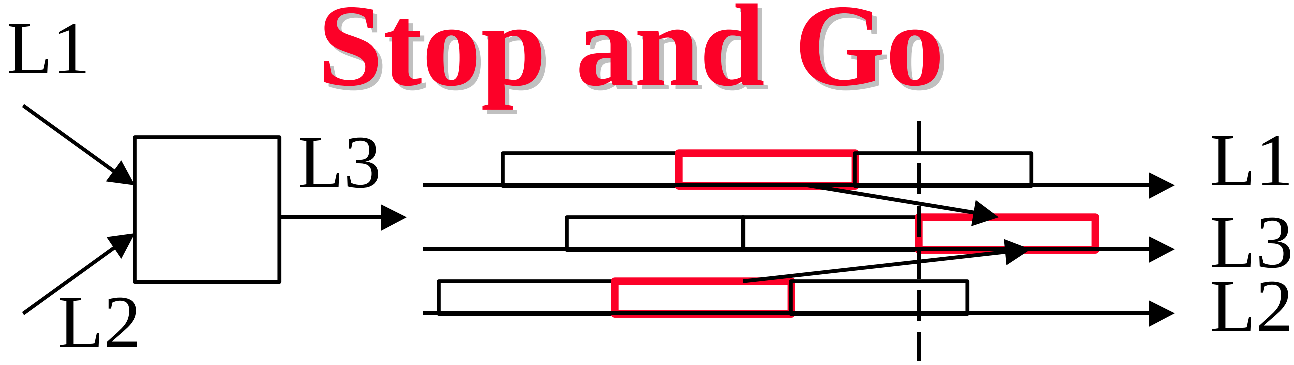
WF2Q: Service



Self-Clocked Fair Queueing (SCFQ)

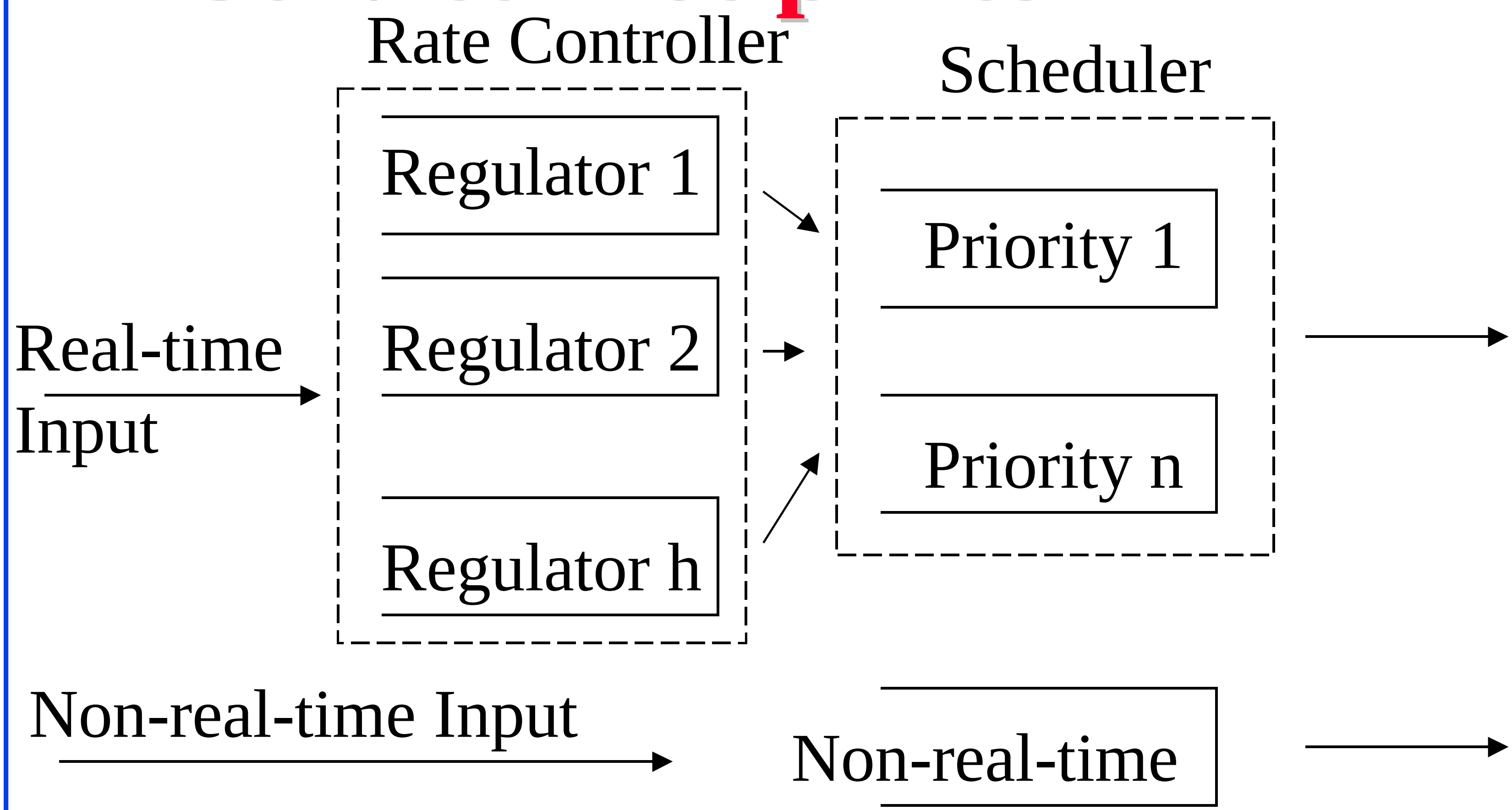
- Computational complexity of computing virtual finishing time in WFQ depends upon the number of times flows change from idle to busy and vice versa. SCFQ reduces it to O(1). Dequeue and enqueue is still O(log(V))
- Uses system clock instead of wall clock (as in Virtual Clock)
- A packet's tag = Length/rate + Max{tag of previous packet in that flow, tag of packet in service at the time of arrival}
- Ref: Golestani

Stop and Go



- Time is divided into constant size frames
- An arriving packet leaves on the next departing frame
- Server cannot idle if there are eligible packets
- Known Properties: Shaping maintained throughout
  $\Rightarrow$ Delay bound possible $\Rightarrow$ Tight delay jitter
- Non-Work conserving. Link idle if no packets for current frame but there are the next frame
- Known Problems: Allocation = PCR $\Rightarrow$ Inefficient

Rate Controlled Service Disciplines



- Rate-Controller (shaper) + Packet Scheduler
  Decouples bandwidth allocation and delay bounds

RCSD (Cont)

- Rate Controller:
  - Each packet is assigned an eligibility time on arrival
  - Packet is held in rate controller and released to scheduler at its eligibility time
- Many different schedulers and rate controllers can be combined to produce different algorithms.

Rate-Jitter	$\times$	Earliest due date
		Static priority
Delay-Jitter		FCFS

Multi-class Scheduling For ATM

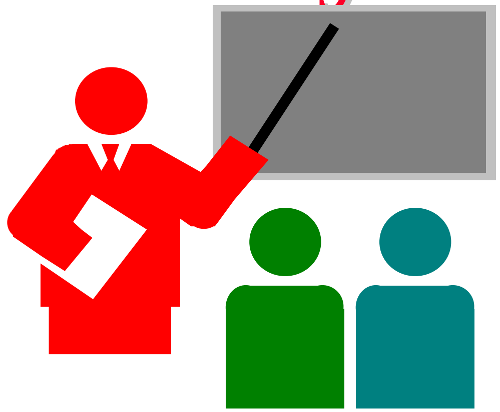
CBR		
rt-VBR		
nrt-VBR	ABR	UBR

- Each class has an allocation = Guaranteed under overload
- Some classes need minimum delay ⇒have priority.
- Some classes are greedy: They will send more than allocated and will want to use all left-over. No left-over capacity.
- Left-over capacity must be fairly allocated.

Scheduling Methods: Comparison

Algorithm	Unfairness	Complexity
Round Robin	∞	O(1)
Fair Queueing	Max	O(log(V))
SCFQ	2Max	O(log(V))
DRR	3 Max	O(1)
Virtual Clock	∞	O(log(V))
WRR	Max	O(1)

Summary



- Schedulers can provide bandwidth, delay, loss guarantees
- Large # of VCs ⇒ Need O(1) complexity
- Frequent rate changes ⇒ Allocation to a VC should depend upon that VC's demands and not on others
- Non-work conserving schedulers provide end-to-end delay guarantees.

References

- For detailed list of references, see http://www.cis.ohio-state.edu/~jain/refs/ref_schd.htm

Thank You!