CS 414 – Multimedia Systems Design

Lecture 18 – Multimedia Transport Subsystem (Part 4)

Klara Nahrstedt

Spring 2009

Administrative

- MP2 demonstrations, Monday 5-7pm in 0216 SC
- HW1 posted today, Friday, February 27, deadline for HW1 is Friday, March 6 midnight
  - Email your solutions in **pdf** format to klara@cs.uiuc.edu or
  - Slide your HW1 solution under the door of 3104 SC
    - HW1 solutions must by typed except figures/drawings if you slide a paper copy of your solutions under the 3104 SC door
- Midterm, March 9, 11-11:50 in class

Outline

- Establishment Phase
  - Negotiation, Translation
  - Admission, Reservation
- Transmission Phase
  - Traffic Shaping
    - Isochronous Traffic Shaping
    - Shaping Bursty Traffic
  - Rate Control
  - Error Control
  - Adaptation

Performance Guarantees

- Every traffic management needs QUEUE MANAGEMENT (QM)
- Statistical versus Deterministic Guarantees
- Conservation of Work
  - QM schemes differentiate if they are work conserving or not
  - **Work conserving system** – sends packet once the server has completed service (examples – FIFO, LIFO)
  - **Non-work conserving scheme** – server waits random amount of time before serving the next packet in queue, even if packets are waiting in the queue

Rate Control

- Multimedia networks use **rate-based mechanisms** (conventional networks use window-based flow control and FIFO)

Work-conserving schemes	**Non-work-conserving schemes**
Fair Queuing	Jitter Earliest-Due-Date
Virtual Clock	Stop-and-Go
Delay Earliest-Due-Data	Hierarchical Round-Robin

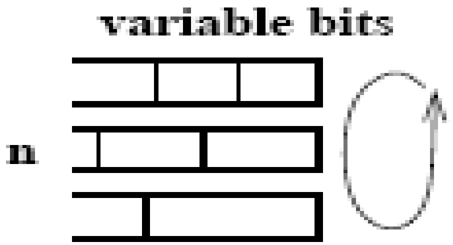
Earliest Due Date (EDD) [Ferrari]

- Based on Earliest Deadline First Scheduling Policy
- EDD works for periodic message models
  - Packet has end-to-end deadline D_i
- EDD partitions end-to-end deadline D_i into local deadlines $D_{i,k}$ during connection establishment procedure

Delay EDD

- Upon arrival of Packet *j* of connection *i*:
  - Determine effective arrival time:
    - $a^e_{i,j} = max(a^e_{i,j-1} + p_i, a_{i,j})$
  - Stamp packet with local deadline:
    - $d_{i,j} = a^e_{i,j} + D_{i,k}$
  - Process packets in EDF order
- Delay EDD is greedy
- Problem with EDD: jitter

Weighted Fair Queuing

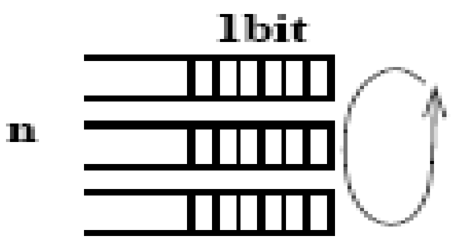


Nagle's solution 1987

isolation for misbehaved customers

Limitations: 1. ignored packet length

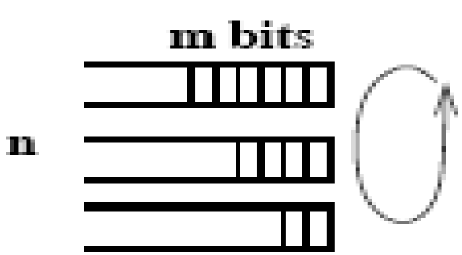
2. sensitive to pattern of packet arrival



Demers, Keshav, Shenker 1990

Fair queueing wait (n-1) bits times before sending

Limitations: give every host the same fraction of bandwidth



Weighted Fair queueing   m > n

(m-bits cycles)

Parekh's Thesis 1992 - proof of strong performance guarantees

WFQ vs FQ

- Both in WFQ and FQ, each data flow has a separate FIFO queue.
- In **FQ,** with a link data rate of *R*, at any given time the *N* active data flows (the ones with non-empty queues) are serviced simultaneously, each at an average data rate of

  ***R* / *N*.**
  - Since each data flow has its own queue, an ill-behaved flow (who has sent larger packets or more packets per second than the others since it became active) will only punish itself and not other sessions.
- **WFQ** allows different sessions to have different service shares. If *N* data flows currently are active, with weights $w_1, w_2 \ldots w_N$, data flow number *i* will achieve an average data rate of $\boldsymbol{R * w_i/(w_1+w_2+\ldots+w_n)}$

Comparison between WFQ and Jitter Control

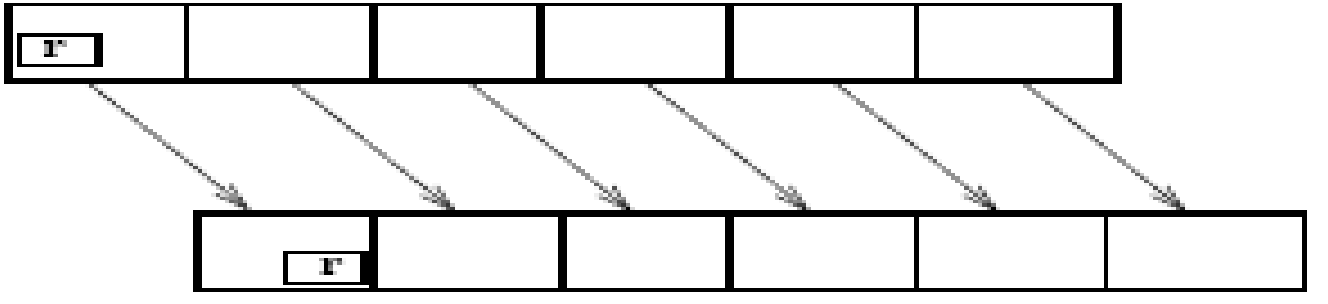
- WFQ guarantees packet delay less than a given value *D*, but as long as delay is within bound it does not guarantee what the delay will be
- Example: send packet at time *t0* over a path whose minimum delay is *d*
  - WFQ guarantees that packet arrives no later than t*0*+*d*, but packets can arrive any time *t0*+ *x* between *[t0+d, t0+D] .*
  - x is jitter

Jitter Control Non-Work-Conserving Schemes

STOP-AND-GO QUEUEING (Golestani 1990)

used with (r,T)-smooth traffic shaping

T-bits



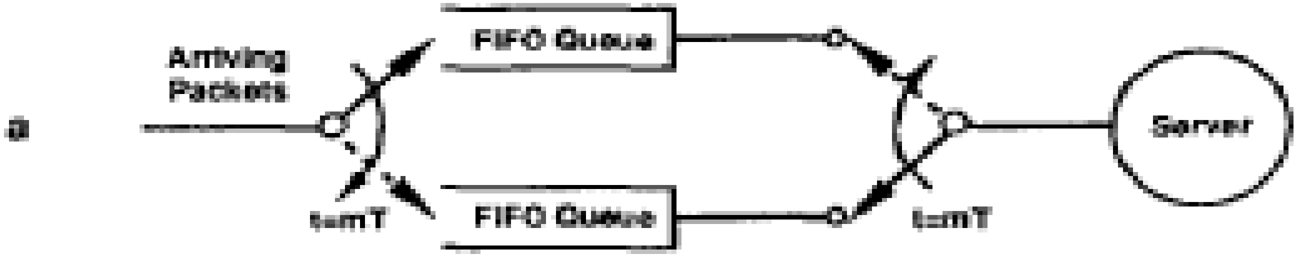
JITTER-EARLIEST DUE-DATE (Verma, 1991)

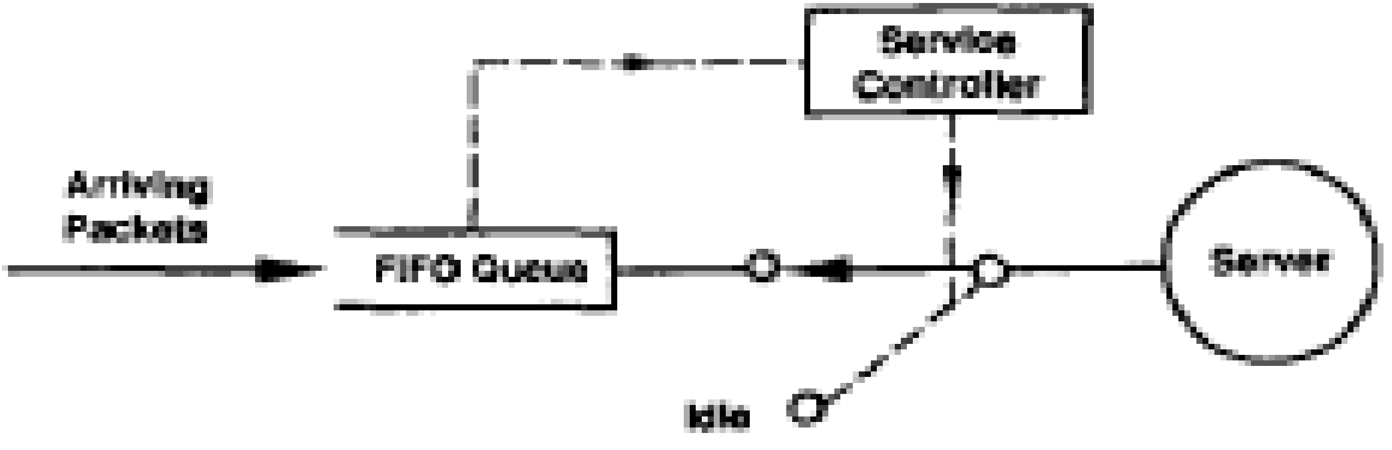
Jitter EDD gives each hop in the flow's path queueing budget b.

Each hop computes its jitter bound j.

At each hop a packet is eligible to be transmitted b-j time units after packet arrives.

Implementation of Stop-and-Go





Jitter-EDD

- Delay-EDD: does not control jitter. This has effect on buffer requirements.
- Jitter-EDD is non-greedy.
- Jitter-EDD maintains **Ahead Time** $ah_{i,j}$, *which is the difference* between local relative deadline *Di,k-1 and actual delay at switch k-1.*
- Ahead time is stored in packet header
- Upon receiving *j-th packet of connection i with* $ah_{i,j}$ *at time* $a_{i,j}$:
  - Calculate ready time at switch *k:*
    - $a^e_{i,j}=max(a^e_{i,j-1} + p_i , a_{i,j})$
    - $r_{i,j} = max(a^e_{i,j} , a_{i,j} + ah_{i,j})$
  - Stamp packet with deadline $d_{i,j}=r_{i,j}+D_{i,k}$ *and process according to EDF* starting from ready time $r_{i,j}$.

Error Control

- Error Detection
  - Ability to detect the presence of errors caused by noise or other impairments during transmission from sender to receiver
  - Traditional mechanisms: check-summing, PDU sequencing
    - Checksum of a message is an arithmetic sum of message code words of a certain word length (e.g., byte)
    - CRC – Cyclic Redundancy Check – function that takes as input a data stream of any length and produces as output a value (commonly a 32-bit integer) – can be used as a checksum to detect accidental alteration of data during transmission or storage
  - Multimedia mechanisms: byte error detection at application PDU, time detection

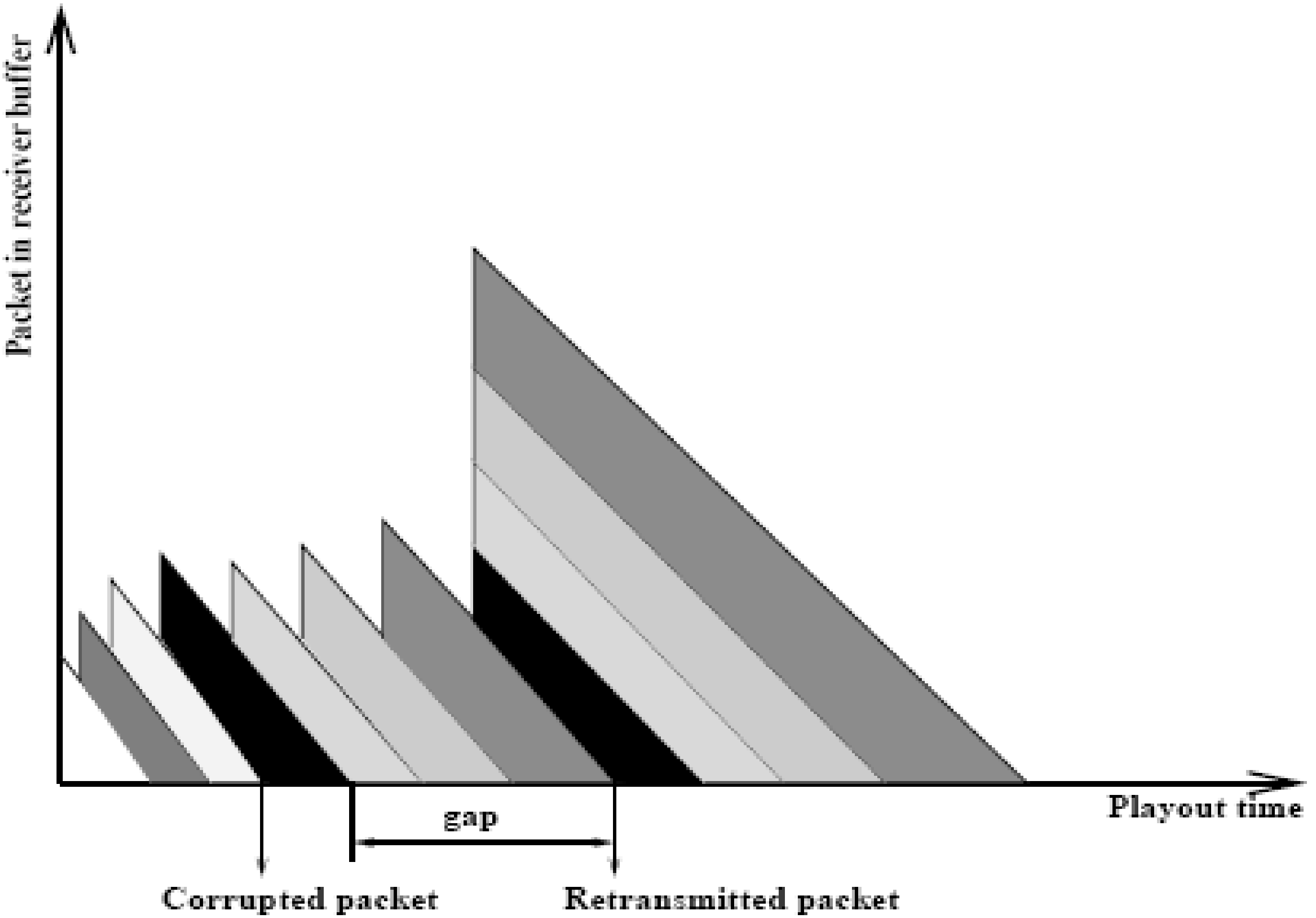
Design of Error Correction Codes

- Automatic repeat-request (ARQ)
  - Transmitter sends the data and also an error detection code, which the receiver uses to check for errors, and requests retransmission for erroneous data
  - The receiver sends ACK (acknowledgement of correctly received data)
- Forward Error Correction (FEC)
  - Transmitted encodes the data with an error-correcting code (ECC) and sends the coded msg. No ACK exists.

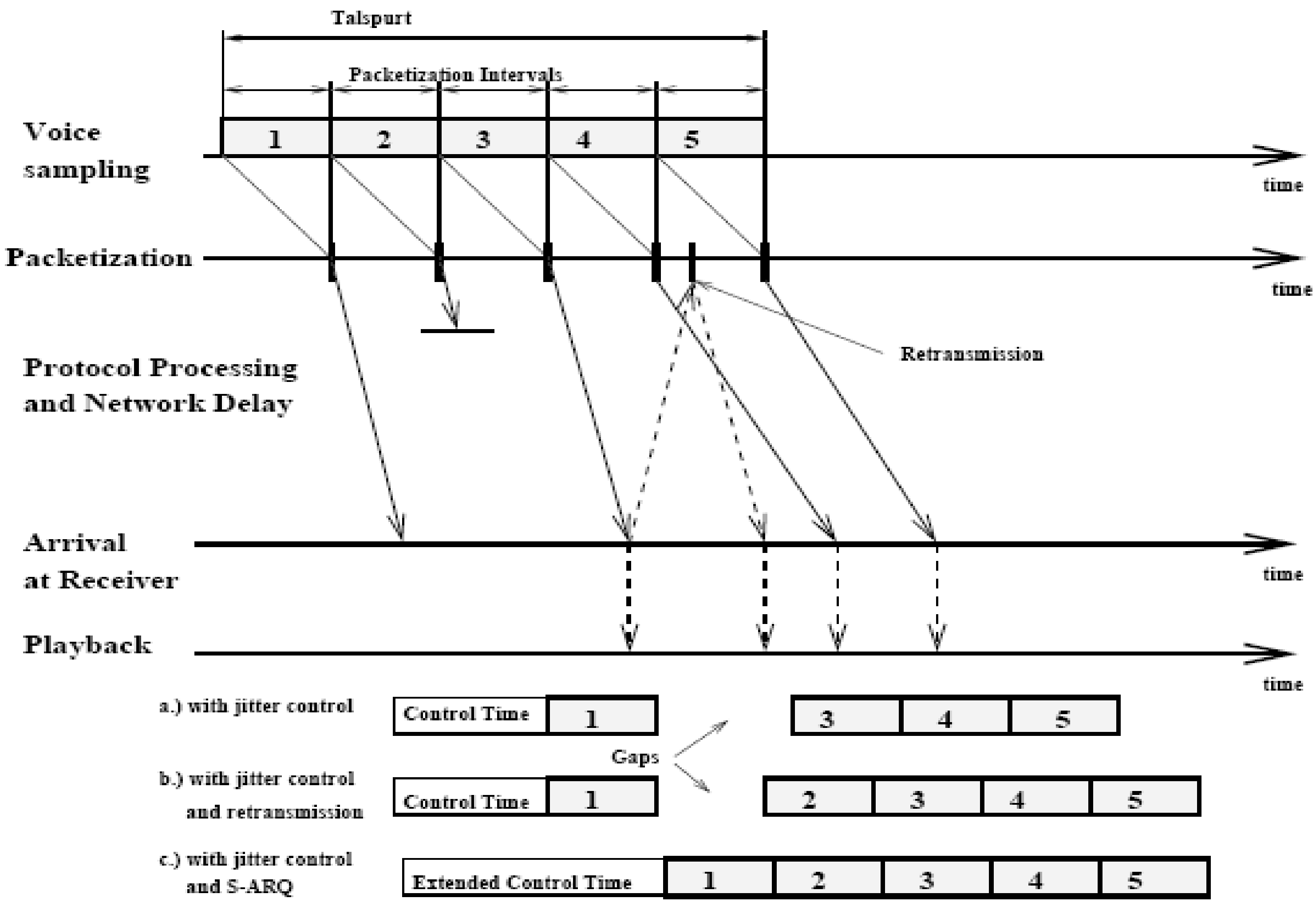
Error Control

- Error Correction
  - Traditional mechanisms: retransmission using acknowledgement schemes, window-based flow control
  - Multimedia mechanisms:
    - Go-back-N Retransmission
    - Selective retransmission
    - Partially reliable streams
    - Forward error correction
    - Priority channel coding
    - Slack Automatic Repeat Request

Go-back-N Retransmission



Jitter Control in Slack Automatic Repeat Request Scheme



Adaptation

- Transmission Phase needs traffic management with rate control and error control
- It also needs monitoring and adaptation
  - Network adaptation
  - Source adaptation
    - Feedback from network to source or feedback from out source
    - Adaptive rate control
    - Traffic shaping

Conclusion

- Establishment Phase
  - Negotiation, Translation
  - Admission, Reservation
- Transmission Phase
  - Traffic Shaping
    - Isochronous Traffic Shaping
    - Shaping Bursty Traffic
  - Rate Control
  - Error Control
  - Adaptation
- Case Studies of Multimedia Protocols