Real-Time Communication

- Integrated Services: Integration of variety of services with different requirements (real-time and non-real-time)
- Traffic (workload) characterization
- Scheduling mechanisms
- Admission control / Access control (policing)
- Deterministic vs. stochastic analysis
  - Traffic characterization
  - Performance guarantees

Providing Real-Time Guarantees

sender application ----> network service ----> receiver application

traffic specification
- packet sizes
- packet inter-arrival times
- general traffic descriptors

performance requirements
- delay
- jitter
- bandwidth
- packet loss

As long as the traffic generated by the sender does not exceed the specified bounds, the network service will guarantee the required performance.
Real-Time Guarantees: Mechanisms

- sender application
- receiver application
- network service
- connection-oriented service
- deterministic packet scheduling in switches and routers
- rigorous (and robust) delay computation
- real-time-connection establishment

Enforcement:
- policing
- traffic shaping

Traffic Description: Traffic Bounding Functions

- Arrival as stochastic process
  \( A = \{ A(t), \ t \geq 0 \} \)

- \( A \) and \( a \) are poor traffic descriptors:
  - time dependent

- Deterministic traffic arrival descriptors (time-independent)
  - Maximum Traffic Function
    \( b(I) \geq \max_{t \geq 0} \{ A(t+I) - A(t) \} \)
  - Maximum Rate Function
    \( b(I)/I \geq \max_{t \geq 0} \{ A(t+I) - A(t) \}/I \)
Maximum Traffic Functions

\[ b(I)/I = \max_{t \geq 0} (A(t + I) - A(t))/I \]

Traffic Bounding Functions

\[ b(I)/I \]

Traffic Bounding Functions

\[ b(I)/I \]

Traffic Bounding Functions

\[ b(I)/I \]

Traffic Bounding Functions

\[ b(I)/I \]
Maximum Rate vs. Maximum Traffic Functions

Maximum Rate Representation

\[ \frac{b(I)}{I} \]

Maximum Traffic Representation

\[ b(I) \]

Traffic Models

**Deterministic:**
1. Periodic model: \((e, p)\)
2. Deferred Server, Sporadic Server model: \((e_S, p_S)\)
3. \((\sigma, p)\) model [Cruz]
4. Leaky bucket model [Turner, ...]: \((\beta, p)\)
5. \((x_{min}, x_{ave}, I, s_{max})\) model [Ferrari & Verma]
6. D-BIND model (Deterministic Bounding Interval Length Dependent) [Knightly & Zhang]
7. \(\Gamma\)-functions [Zhao]

**Probabilistic:**
1. S-BIND model (Stochastic Bounding Interval) [Knightly]
2. Markov-Modulated Poisson Processes
Traffic Bounding Function \( b(.) \)

- Let \( b(.) \) be a monotonically increasing function.
- \( b(.) \) is a deterministic traffic constraint function of a connection if during any interval of length \( I \), the number of bits arriving during the interval is no greater than \( b(I) \).
- Let \( A[t_1, t_2] \) be the number of packets arriving during interval \([t_1, t_2]\). Then, \( b(.) \) is a traffic constraint function if

\[
\forall s, I > 0, \quad A[s, s + I] \leq b(I)
\]

- Each model defines inherently a traffic constraint function.
- The accuracy of models can be compared by comparing their constraint functions.

Cruz' \((\sigma, \rho)\) Model

- If the traffic is fed to a server that works at rate \( \rho \) while there is work to be done, the size of the backlog will never be larger than \( \sigma \).
- IOW: The number of jobs/cells released during any interval \( I \) does not exceed \( \rho I + \sigma \).
- Graphical representation:

\[
b^{(\sigma, \rho)}(I) = \sigma + \rho \times I
\]
The Leaky Bucket Model

- Implementation:
  - Maintain counter for each traffic stream.
  - Increment counter at rate $\rho$, to maximum of $\beta$.
  - Each time a packet is offered, the counter is checked to be > 0.
  - If so, decrement counter and forward packet; otherwise drop packet.

Concatenating Leaky Buckets

- What about limiting the maximum cell rate?

$\beta_1 = 1$
(\(x_{\text{min}}, x_{\text{ave}}, I_{\text{ave}}, s_{\text{max}}\)) model [Ferrari & Verma]

- \(x_{\text{min}}\): minimum packet interarrival time
- \(x_{\text{ave}}\): average packet interarrival time
- \(I_{\text{ave}}\): averaging interval length
- \(s_{\text{max}}\): maximum packet length

\[
b(x_{\text{min}}, x_{\text{ave}}, I_{\text{ave}}, s_{\text{max}})(I) = \left( \min \left( \frac{t \mod I_{\text{ave}}}{x_{\text{ave}}} I_{\text{ave}}, \left| I_{\text{ave}} \right| \right) + \frac{I_{\text{ave}}}{I_{\text{ave}}} \right)s_{\text{max}}
\]

D-BIND [Knightly & Zhang]

- Other models do not accurately describe burstiness.
- Rate-interval representation:

Model traffic by multiple rate-interval pairs: \((R_k, I_k)\), where rate \(R_k\) is the worst-case rate over every interval of length \(I_k\).
D-BIND (2)

- Constraint function for D-BIND model with $P$ rate-interval pairs:
  \[ b(t) = \frac{R_s I_s - R_{s-1} I_{s-1}}{I_s - I_{s-1}} (t - I_{s-1}) + R_s I_s, \quad I_{s-1} \leq t \leq I_s \]
  \[ b(0) = 0 \]
  \[ b(t) = b(t - \lceil t / I_s \rceil) \text{ for } t > I_s \]

- Comparison:

Policing for the D-BIND Model

- Lemma: If $b(t)$ is piece-wise linear concave, then $R_k$ is strictly decreasing with increasing $I_k$.

- Lemma: If a piece-wise linear constraint function $b(t)$ with $P$ linear segments is concave, then the source may be fully policed with a cascade of $P$ leaky buckets.
Delay Computation: Overview

- Delay computation for FIFO server with deterministically constraint input traffic:

\[ d_{\text{FIFO}} = \max_{I \geq 0} \left\{ \sum_{j} b(I) - RI \right\} / R \]

End-to-End Analysis

- Traffic regulation: reshape traffic to adhere to traffic function.
- Alternative: re-characterize by accounting for burstiness added by queueing delays

\[ F_y(I) = F_x(I + d_y) \]

- where \( d_y \) is delay on Server \( Y \).
- Deterministic Case:

\[ d_{\text{2}} = \sum_{X \in R} d_X \]
Switch Scheduling

- **Work-conserving** (greedy) vs. **non-work-conserving** (non-greedy) mechanisms.
- **Rate-allocating** disciplines: Allow packets to be served at higher rates than the guaranteed rate.
- **Rate-controlled** disciplines: Ensures each connection the guaranteed rate, but does not allow packets to be served above guaranteed rate.

- **Priority-based** scheduling:
  - fair queuing
  - virtual clock
  - earliest due date (EDD)
  - rate-controlled static priority (RCSP)

- **Weighted Round-Robin** scheduling:
  - WRR

Bit-by-Bit Weighted Round-Robin

- bit-by-bit round robin
- each connection is given a weight
- each queue served in FIFO order
Fair Queueing [Demers, Keshav, Shenker]

- Emulate Bit-by-Bit Round Robin by prioritizing packets.
- Prioritize packets on basis of their finish time \( f_j \):
  - \( a_j \): arrival time of \( j \)-th packet
  - \( e_j \): length of packet
  - \( f_j \): finish time
  - \( BW \): allocated fraction of link bandwidth

\[
f_j = \max(f_{j-1}, a_j) + \frac{e_j}{BW}
\]

- Example:

![Diagram](image)

Complications: 4 1.5 1

What if connections dynamically change?

Virtual Clock Algorithm [L. Zhang]

- Emulate time-division multiplex (TDM) mechanism
- However:
  - TDM: when some connections idle, the slots assigned are idle
  - VC: idle slots are deleted from TDM frames

- auxiliary virtual clock (auxVC): finish time of \( j \)-th packet.
- virtual tick (Vtick): time to complete transmission of ready \( j \)-th packet.

\[
Vtick_j = \frac{e_j}{BW}
\]

- Replace \( f_j \) by \( Vtick_j \): VC becomes identical to WFQ algorithm!
- Will analyze delay analysis later.
Rate-Controlled Static Priority (RCSP) [Zhang & Ferrari]
Traffic Regulation in RCSP

• Hold packets in regulator to guarantee minimum inter-packet arrival time.
  \[ r_{ij} = \max(a_{ij}, r_{ij-1} + p_r) \]
• Implementation: buffer and timers in traffic regulator.
• Buffer requirements:
  \[ B_p = \left( \left\lfloor \frac{d_p}{p_r} \right\rfloor + \left\lfloor \frac{d_{p-1}}{p_r} \right\rfloor \right) e_p \]

Is it Necessary to Regulate?

• [Liebeherr, Wrege, Ferrari, Transactions on Networking, 1995]
• Generalization of schedulability for arbitrary traffic constraint functions \( b(I) \):

**Theorem**: A set \( N \) of connections that is given by \( \{b_j, d_j\} \) is schedulable according to a static-priority algorithm if and only if for all priorities \( p_r \), and for all \( I \geq 0 \) there is a \( t \) with \( t \leq d_r - s_p^{\min} \) such that:

\[
\forall I, \exists t \leq d_r - s_p^{\min} : I + t \geq \sum_{j \in C_p} b_j(I) - s_p^{\min} + \sum_{q \in C_t} b_q((I + t) + \max_b b^{\max}_r \}
\]
Earliest Due Date (EDD) [Ferrari]

- based on EDF
- delay-EDD vs. jitter-EDD
- works for periodic message models (single packet in period): \((p_i, I, D_i)\)
- partition end-to-end deadline \(D_i\) into local deadlines \(D_{ik}\) during connection establishment procedure.
- 2-Phase establishment procedure:

\[a_{e_{ij},j} = \max(a_{e_{ij-1},j} + p_i, a_{i,j})\]
\[d_{i,j} = a_{e_{ij},j} + D_{ik}\]

Phase 1: tentative establishment

Phase 2: relaxation

Delay EDD

- Upon arrival of Packet \(j\) of Connection \(i\):
  - Determine effective arrival time: \(a_{e_{ij},j} = \max(a_{e_{ij-1},j} + p_i, a_{ij})\)
  - Stamp packet with local deadline: \(d_{ij} = a_{e_{ij},j} + D_{ik}\)
  - Process packets in EDF order.

- Delay EDD is greedy.

- Can be mapped into special case of Sporadic Server.

- Acceptance test (\(\Delta = \) total density): \(\Delta + 1/p < 1 - 1/p_{\text{min}}\)
- Offered local deadline: \(LD_i = \min(p_r, 1/(1-\Delta-1/p_{\text{min}}))\)

- Problem with EDD: jitter
  - max end-to-end delay over \(k\) switches: \(\sum_k D_{ik}\)
  - min end-to-end delay over \(k\) switches: \(k\)
### Jitter EDD

- Problem with Delay-EDD: does not control jitter. This has effect on buffer requirements.
- Jitter-EDD maintains **Ahead Time** $ah_{i,j}$, which is the difference between local relative deadline $D_{i,k-1}$ and actual delay at Switch $k-1$.
- Ahead time is stored in packet header (alternatively, we use global time synchronization)
- Upon receiving the $j$-th packet of Connection $i$ with $ah_{i,j}$ at time $a_i,j$:
  - Calculate ready time as Switch $k$:
    \[
    a_i^{r} = \max(a_{i,j-1} + p_{i,j}, a_i,j) \\
    r_{i,j} = \max(a_{i,j} + ah_{i,j})
    \]
  - Stamp packet with deadline $d_{i,j} = r_{i,j} + D_{i,k}$ and process according to EDF starting from ready time $r_{i,j}$.
- Result: Regenerate traffic at each switch.

### Rate Control vs. Jitter Control

- **Rate Control**

- **Jitter Control**
Simple EDF with Arbitrary Arrival Functions
[Liebeherr, Wrege, Ferrari: Transactions on Networking, 1995]

**Theorem:** A set $\Pi$ of connections that is given by $\{b_i, d_i\} \in \Pi$ and $d_i \leq d_j$ whenever $i < j$ is EDF schedulable if and only if for all $I \geq d_i$:

$$I \geq \sum_{j=1}^{m} b_j (I - d_j) + \max_{\{s_k^{\max}\} \text{ where } k, d_k \leq I} \max_{k} s_k^{\max}$$

Informal “proof”: A deadline violation occurs at time $I$ if the maximum traffic arrivals with deadline before or at time $I$, i.e. $I < \sum_{j=1}^{m} b_j (I - d_j)$ exceeds $I$.

---

EDF Test for Special Cases: Example $(\sigma, \rho)$

- For some traffic models, closed-form expressions for the schedulability test exist.
- For $(\sigma, \rho)$ traffic:

$$I \geq \sum_{j=1}^{m} \sigma_j + \rho_j (I - d_j) + \max_{k, d_k \leq I} \max_{s_k^{\max}} \text{ for } d_j \leq I < d_{j+1}, 1 \leq j < |\Pi|$$

$$I \geq \sum_{j=1}^{m} \sigma_j + \rho_j (I - d_j) \text{ for } I \geq d_{|\Pi|}$$

- A closed form for the delay can be given as follows:

$$d_j = \frac{\sigma_j + \sum_{i=1}^{j-1} (\sigma_i - \rho_i d_i) + \max_{k, d_k \leq I} \max_{s_k^{\max}}}{1 - \sum_{i=1}^{j-1} \rho_i}$$
Weighted Round Robin (WRR)

- Each connection $i$ is assigned a weight $w_i$, i.e., it is allocated $w_i$ slots during each round.
- Slot: time to transmit maximum-sized packet.

- **Traffic model:**
  - periodic $(p_i, e_i, D_i)$
  - variable bit rate models possible

- **Realizations:**
  - greedy WRR
  - Stop-and-Go (SG)
  - Hierarchical Round Robin (HRR)

Throughput and Delay Guarantees

- Each connection $i$ is guaranteed $w_i$ slots in each round.
- Round length $RL$: upper bound on sum of weights (design parameter)

$$\sum w_i \leq RL$$

- **Constraints:**
  - Delays:
    1. $RL \leq p_{\min}$
    2. $w_i \geq \left\lceil \frac{e_i}{p_i/RL} \right\rceil$

- at first switch: $\left\lceil \frac{e_i}{w_i} \right\rceil RL$
- downstream: once packet passes first switch, it is immediately eligible on switches downstream -> has to wait at most $RL$

=> end-to-end delay through $N$ switches:

$$w_i \leq \left\lceil \frac{e_i}{w_i} \right\rceil RL + N - 1 \leq p_i + (N - 1)RL$$
Problems with Greedy WRR

- Greedy WRR does not control jitter:
  
  \[
  \begin{align*}
  &\text{First Switch} \\
  &\text{min end-to-end delay:} \quad e_i + (N-1) \\
  &\text{max end-to-end delay:} \quad p_i + (N-1)RL \\
  &\text{jitter:} \quad p_i - e_i + (N-1)(RL - 1)
  \end{align*}
  \]

- Buffer needed at \( k \)-th switch for Connection \( i \):
  
  \[
  (1 + \left\lceil \frac{(k - 1)(RL - 1)}{p_i} \right\rceil) e_i
  \]

- Need traffic shaping at each switch.

Non-Greedy WRR

- Actual length of rounds in greedy WRR varies with amount of traffic at switch.

- Non-greedy WRR schemes fix round length into fixed-length frames.

- Stop-and-Go [Golestani]

- Hierarchical Round Robin [Kalmanek, K., K.]
Stop & Go [Golestani, 1990]

- Frame-based: divide time in frames of length $RL$.
- Packet arriving during frame at input link is eligible for transmission during next frame on output link.

- Stop-and-Go is not work-conserving.
- Traffic model [(r, RL) smooth traffic]: during each frame of length $RL$, the total number of bits transmitted by source does not exceed $rRL$ bits.

**Proposition:** If the connection satisfies (r,RL) smoothness at the input of the first server, and each server ensures that packets will always go out on the next departing frame, the connection will satisfy (r,RL) smoothness at each server throughout the network.

Stop & Go: Implementation

- Implementation of scheduler is not defined by Stop-and-Go frameworks.

- Implementation 1: FIFO scheduler with double-queue structure

- Implementation 2:
Multi-Frame Stop-and-Go

[For example, Zhang&Knightly: "Comparison of RCSP and SG", ACM Multimedia, 4(6) 1996]

- Problem with Stop-and-Go (or any other frame-based approach): delay-bandwidth coupling
  - Delay of packet is bounded by a multiple of frame time. This is a problem, for example for low-bandwidth, low-delay connections.
  (Why?)
- Solution: Use multi-level framing. Example:

<table>
<thead>
<tr>
<th>RL_2</th>
</tr>
</thead>
<tbody>
<tr>
<td>RL_1</td>
</tr>
</tbody>
</table>

- Hierarchical framing with n levels with frame sizes $RL_1, ..., RL_n$, where $RL_{m+1} = K_m RL_m$ for $m = 1, ..., n-1$.
- Stop-and-Go rule for packets of level-$p$ connection: Packets that arrived during a $RL_p$ frame will not become eligible until the start of the next $RL_p$ frame.
- Packets with smaller frame size have higher priority (non-preemptively) over packets with larger frame size.

Hierarchical Round Robin

[Kalmanek, Kanadia, Keshav, 1990]

- End-to-end delay and jitter of S&G depends on $RL$ only.
- How about having multiple S&G servers, with different $RL$’s, and multiplex them on the same outgoing link?

- Server $X$ is seen as periodic stream of requests by Server $S$, with
  - $e_x = sw_x$, $p_x = RL_x$, $D_x = RL_x$
  - schedule using rate-monotonic scheduler
  - Configuration time test: check whether task set $\{(sw_x, RL_x, RL_x)\}$ is schedulable.
- Admission Control Test:
  - Bandwidth test: check sum of required $w_i$’s $\leq sw_x$
  - Delay test: $p_i + N RL_x$
  - Jitter test: $2 RL_x$, with buffer requirement $2 w_i$