QoS in Frame relay networks

**Frame Relay: characteristics**

- Packet switching with virtual circuit service
  - Label named DLCI: Data Link Connection Identifier
  - Virtual circuits are bi-directional
- "Connection" is associated with the virtual circuit
- No error control (DL-control is not used even at edge)
- No flow control
- LAP-F protocol
- Packet size:
  - variable up to 4096 byte
- Mainly thought for data traffic

**Frame Relay: user-network interface**

- Negotiable parameters, a-priori, on a contract basis:
  - CIR (Committed Information Rate) [bit/s]
  - CBS (Committed Burst Size) [bit]
  - EBS (Excess Burst Size) [bit]
- CIR: guaranteed bit rate (throughput)
- CBS: amount of data the network is willing to accept over a measurement period T
- EBS: amount of excess data the network may transfer over T. Packets are marked with the DE bit set to 1
- Data exceeding CBS+EBS are directly discarded at network access

**Frame Relay: measurement interval T definition**

<table>
<thead>
<tr>
<th>CIR</th>
<th>CBS</th>
<th>EBS</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt; 0</td>
<td>&gt; 0</td>
<td>&gt; 0</td>
<td>CBS/CIR</td>
</tr>
<tr>
<td>&gt; 0</td>
<td>&gt; 0</td>
<td>= 0</td>
<td>CBS/CIR</td>
</tr>
<tr>
<td>= 0</td>
<td>= 0</td>
<td>&gt; 0</td>
<td>EBS/Access Rate</td>
</tr>
</tbody>
</table>

**LAPF packet**

- **ADDRESS field**
  - DLCI: Data Link Connection Identifier
  - FECON/BECN: forward/backward explicit congestion notification
  - DE: discard eligibility
  - C/R: command/response
  - D/C: DLCI or DL-CORE
  - EA: extension bit

**ADDRESS field**

- Default format (2 byte)
- 3 byte format
- 4 byte format
Frame Relay: resource allocation

\[ \sum_{A,j} CIR_{A,j} \leq ACCESS\_RATE_A \]

- where \( A,j \) refers to the VC from \( A \) to \( j \)

\[ \sum_{A,j} CIR_{A,j} \leq ACCESS\_RATE_A \]

- where \( A,j \) refers to the VC from \( A \) to \( j \)

\[ \sum_{A,j} CIR_{A,j} \leq ACCESS\_RATE_A \]

- where \( A,j \) refers to the VC from \( A \) to \( j \)

\[ \sum_{A,j} CIR_{A,j} \leq ACCESS\_RATE_A \]

- where \( A,j \) refers to the VC from \( A \) to \( j \)

\[ \sum_{i,j} CIR_{i,j} \leq ACCESS\_RATE_A \]

- where \( i,j \) refers to the VC from \( i \) to \( j \)

\[ \sum_{i,j} CIR_{i,j} \leq LINK\_SPEED \quad \forall \text{ links} \]

- where \( i,j \) refers to the VC from \( i \) to \( j \)
QoS in Frame relay networks

Frame Relay: resource allocation

\[ \sum_{i,j} \text{CIR}_{i,j} \leq \text{LINK\_SPEED} \quad \forall \text{ links} \]

where \( i,j \) refers to the VC from \( i \) to \( j \)

Frame Relay: algorithms

- Policing, or conformance verification
  - Leaky Bucket
  - Token Bucket
- Congestion control
  - backward
  - forward

Conformance verification

- Basic idea
  - If a packet reaches network access and is conformant to the CBS constraint over \( T \), it is transmitted at high priority with \( \text{DE} = 0 \)
  - If a packet reaches network access and is not conformant to CBS over \( T \) but it is conformant to CBS+EBS over \( T \), it is transmitted at low priority with \( \text{DE} = 1 \)
  - If a packet reaches network access and is not conformant to CBS+EBS over \( T \), it is discarded
- Same algorithms can be used to do shaping
  - Traffic adaptation to make it conformant
  - Delay instead of marking/dropping

Leaky Bucket

- As a traffic regulator
  - User traffic entering the buffer is transmitted at a maximum CBR rate equal to \( \rho \)
  - User traffic exceeding the buffer size \( B \) is dropped
  - Any source becomes a CBR source at rate \( \rho \)
    - if packet size is fixed
  - When using to do conformance verification, if packets arrive earlier than they should be, with respect to \( \rho \), drop it
QoS in Frame relay networks

Leaky Bucket

- Buffer size B is not a critical parameter: we can assume infinite buffer size
- Amount of data sent over a period T is \( \leq T\rho \)
- Dimensioning of parameter \( \rho \) for VBR traffic with peak \( B_p \) and average \( B_M \)
  - \( \rho \geq B_p \): too much traffic could be discarded
  - \( \rho \geq B_M \): waste of link bit rate, largely underutilized
- Traffic regulator which does not admit any burstiness

Token Bucket

- Tokens are generated at a fixed rate \( \rho \)
- A maximum number of \( \beta \) tokens can be stored in the token buffer
  - Permits some burstiness
- User data are sent over the network only if there is a token available in the token buffer
- Maximum amount of data send over a period T is \( \leq T\rho + \beta \)
- The source becomes a VBR source with
  - \( B_M = \rho \)
  - \( B_p = \) access rate
  - Burst duration = \( \beta \)
- Access to the network can be further regulated with a cascading leaky bucket to limit \( B_p \)

Token Bucket + Leaky Bucket

- Regulates average rate \( \rho_2 \), peak rate \( \rho_1 \), burst duration \( \beta \)

All packets conformant to CBS

One packet at low priority
QoS in Frame relay networks

Measurements problems

- Better solution is a temporal sliding window of W seconds
  - When a packet arrives, the rate is measured accounting for the amount of byte received during the last W seconds
  - Difficult to implement (necessary to remember packet arrival times)
- It is possible to exploit a fluid approximation
  - New rate R estimate at each packet arrival
  - At packet arrival, we assume that the flow has transmitted R*W byte in the last W seconds and the estimate of R is updated

Congestion

- Informally
  - too many sources sending too much data too fast for network to handle
- More formally
  - Average input rate larger than average output rate
- Congestion signal
  - long delays (queueing in router buffers)
  - lost packets (buffer overflow at routers)
- Effect
  - Retransmissions (sometimes un-needed)
  - Reduced throughput

Effect of congestion

- Multihop paths
- As red traffic increases, all arriving blue pkts at upper queue are dropped
- The blue throughput goes to 0
- When packet dropped, any upstream transmission capacity used for that packet was wasted!

Solutions?

- Increase the buffer size
  - Buffers helps in solving
    - Contentions
    - Short term congestion only
  - For permanent overload, the excess traffic is lost, regardless of the buffer size
- Increase the link speed
  - We may even create worse congestion
- Increase processing speed
  - We will transfer more packets, possibly exacerbating the congestion
- Congestion is created by an excess of traffic
  - To solve it, we need to reduce the input traffic
Approaches to congestion control

- Drop based
  - Network only drop packets when needed ad relies on end-to-end (transport) protocols to solve the congestion
- Credit based
  - Network nodes provide credits to upstream nodes
    - Backpressure
- Signalling based
  - Network nodes detect congestion and signal to users
    - Via a single or few bit (forward/backward)
    - Via explicit rate computation
- In all cases, rely on cooperation

Features of the approaches

- Drop
  - Easy
  - No need to be flow aware
- Credit
  - Very complex
  - Need to be flow aware to avoid blocking a link
- Signalling
  - Can trade complexity vs effectiveness
  - Can be either flow unaware or flow aware

Frame Relay: congestion

- Summation of CIR over all virtual circuits on each link may exceed the available bit rate over a link (overbooking)
  - Creates congestion, potentially a long-term congestion
- Traffic burstiness may create congestion (typically short term congestion)
- Need to control congestion?
  - X.25 (ISDN) may exploit link-by-link (hop-by-hop) flow control (and internal switch backpressure) to control (un-fairly) congestion
  - In Internet the congestion control is delegated to hosts running TCP, the network simply drops packets
- Frame relay, which does not implement flow control, uses explicit signaling from network nodes to signal congestion to users through FECN and BECN bits

Congestion control in Frame Relay

- Flow control is not supported in Frame Relay
- The network is unprotected against congestion
  - Only protection mechanism is packet discarding
- Congestion should not occur if sources are sending at CIR!
  - When a switch (network node) establishes that congestion has occurred, to signal congestion it sets one among two bits:
    - FECN (Forward technique)
    - BECN (Backward technique)

Congestion control: goals

- Avoid packet loss
- Constraints?
  - Maximum network utilization
  - Fairness
  - Often in contrast
- Simple case: all flows are alike
  - Fairness means to provide the same set of resources to all flows
  - Over a single bottleneck the problem is trivial
  - Network wide problem

Congestion control: an example

- Maximize the bit rate received by each flow
  - Flow 1: 5 Mb/s
  - Flow 2: 5 Mb/s
  - Flow 3: 40 Mb/s
- Maximize the overall network utilization
  - Flow 1: 10 Mb/s
  - Flow 2: 0 Mb/s
  - Flow 3: 45 Mb/s
Max-min fairness

- One possible definition of fairness
- A bandwidth allocation is defined as max-min if
  - It maximizes the bandwidth allocation to flows who receive the minimum allocation
- Property:
  - A max-min allocation is such that, to increase the bandwidth allocated to another flow which is already a smaller or equal bandwidth
  - In other words, no bandwidth increase can be obtained without penalizing flows already receiving a smaller allocation
- A max-min allocation cannot be obtained with local assignments
  - A global network view is needed

Max-min fairness: algorithm

- Given: topology, link capacity, flows and flow routing
  1) The algorithm starts with a 0 allocation to all flows, each flow is marked as unsatisfied
  2) The allocation of all unsatisfied flows is increased by the same, small, quantity, until a bottleneck link is saturated
  3) All bottlenecked flows are saturated, thus, cannot receive a larger allocation
  - Bottlenecked flows are marked as satisfied
  4) Goto 2, until all flows are bottlenecked and satisfied
- Must re-run for any topology or flow modification

Max-min fairness: example

- Problem: find a fair bandwidth allocation to flows, according to the max-min paradigm
- Order in which links are saturated
  - L1, L4, L3, L2
- Solution: fair max-min allocation
  - S1: 45.25 Mbps
  - S2: 20.75 Mbps
  - S3: 17 Mbps
  - S4: 17 Mbps
  - S5: 37.75 Mbps
  - S6: 20.75 Mbps
  - S7: 20.75 Mbps
  - S8: 20.75 Mbps

Forward congestion

- When a switch detects congestion, it sets the FECN bit to 1 on all arriving packets sharing the congested buffer
- Congestion signaled to all congested VCs
- When the congestion indication reaches the receiver, it is redirected to the transmitter on a data flow traveling in the opposite direction
- The transmitter reduces the transmission speed according to a standardized algorithm
- Properties:
  - Relatively slow
  - Simple to implement
  - No additional traffic is created, if there is a data flow from receiver to transmitter (normally at least Acks are sent)

Backward congestion

- When a switch detects congestion, it sets the BECN bit to 1 on all packets belonging to congested VCs
  - These packets are not stored in the congested buffer!
  - Ad-hoc signaling packets may be generated by the switch if no data traffic is flowing in the opposite direction
- The transmitter reduces the transmission speed according to a standardized algorithm when it detects packets with BECN=1
- Properties:
  - Relatively fast
  - Complex: need to store a list of congested DLCI on the forward path and to wait (or create after a timeout) packets with the proper DLCI on the backward path
QoS in Frame relay networks

Source behaviour: FECN

- The FECN technique is based on the idea that congestion phenomena are relatively slow
- Transmitter
  - Start transmitting at a speed equal to CIR
  - Computes the percentage of LAP-F frames received with a FECN bit set to 1 (FECN<sub>1</sub>) over a pre-determined time interval
    - If FECN<sub>1</sub> is >50%, the emission rate is reduced
    - If FECN<sub>1</sub> is <50%, the emission rate is incremented
  - Rate based transmitter:
    - \( R_{\text{initial}} = \text{CIR} \)
    - \( R_{\text{initial}} = 1/8 R_0 \)
    - If a single frame with FECN=1 is received:
      - \( R_{\text{new}} = 1/8 R_0 \)
    - If a single frame with FECN=0 is received:
      - Increase rate

Source behaviour: BECN

- The BECN technique is based on the idea that congestion phenomena are fast
  - Instantaneous reaction (not based on \( \delta = 2 \text{RTT} \))
  - \( R_{\text{initial}} = \text{CIR} \)
  - If a single frame with BECN=1 is received:
    - \( R_{\text{new}} = 1/8 R_0 \)
  - If a single frame with BECN=0 is received:
    - Increase rate

Congestion control: issues

- How to detect congestion?
  - Measure ingress flow speed in each buffer
    - Over which time interval?
    - Would complexity given the binary feedback available?
    - Always balance complexity, performance, signaling capability
    - Operates on a flow basis or on traffic aggregate?
    - Threshold on buffer occupancy
      - Instantaneous buffer occupancy
      - "Fast", but unreliable
      - Typically exists some hysteresis to avoid switching between congested/non-congested states
      - Average occupancy over a time-slicing measurement window
      - How the window size should be determined?
      - More stable, but slower in reaction
    - Occupancy derivative
      - More precise than occupancy alone
      - Typically, more than 100 packets should be treated differently if the "peakwise" buffer occupancy was 20 or 50 packets
      - Need to define time interval over which the derivative
    - Thresholded value?
      - Close to zero occupancy to exploit most of the buffer size
      - Enough space below threshold to allow for synchronous arrivals and avoid unneeded completion signals

- Buffer sizing?
  - Buffer above threshold should increase proportionally to
    - Number of connections involved in congestion
    - Connection RTTs
    - Need to buffer in-flight packets
    - Connection rate
  - Always pay attention to
    - The scenario in which algorithms are compared
    - Network topology (often single bottleneck node examined)
    - Number of flows
    - Flow behavior
    - Difficulty in properly setting parameters
    - If choosing wrong values what happens?
    - How difficult to set up proper values?
    - Algorithm robustness to parameter setting
    - Algorithm complexity w.r.t. performance gain
  - All parameters (threshold, measurement window, buffer size) could be set off-line or modified at run time
  - Run time modification is worth the effort?

- Source behaviour: FECN

- Measuring interval \( \delta \approx 2 \text{RTT} \)
- Rate based transmitter:
  - \( R_{\text{initial}} = \text{CIR} \)
  - \( \text{if } \text{FECN} > \text{FECN}_0 \)
  - \( R_{\text{new}} = 0.875 W_{\text{max}} \)
  - \( \text{if } \text{FECN} < \text{FECN}_0 \)
  - \( R_{\text{new}} = 0.0625 W_{\text{max}} \)
    - If not transmitting for \( T \), restart from \( R_{\text{initial}} \)
  - Window based transmitter:
    - \( W_{\text{max}} = 1 \)
    - \( \text{if } \text{FECN} > \text{FECN}_0 \)
    - \( W_{\text{max}} = 0.875 W_{\text{max}} \)
    - \( \text{if } \text{FECN} < \text{FECN}_0 \)
    - \( W_{\text{max}} = W_{\text{max}} \)
  - \( \text{TODO: } \text{Algorithm complexity} > 2 \text{RTT} \)

- Source behaviour: BECN

- Fairness among flows
  - More active flows receive more congestion signals
  - May be ok, since are creating more congestion, but is it max-min fair?
  - Temporarily inactive flows?

- Signalling frequency
  - Any reaction to congestion signals is constrained by the flow RTT
  - \( \text{RTT} \) would make sense to adapt the (congestion) signaling frequency to flow RTT
  - Practically impossible to know flow RTT in network nodes (may be done at the interface)
  - Connections with shorter RTTs react faster
  - Both when increasing and decreasing rate

- When congestion is detected (set up congestion bit in the header)
  - Operate on packet reaching the buffer or leaving the buffer?