Small cycle impact in pulsed queues

Yizhou Li (Huawei)
Guanhua Zhuang (Huawei)
Li Dong (Shenyang Institute of Automation)
Wenbin Dai (Shanghai Jiao Tong University)
Introduction

• Pulsed Queues was presented in Nendica as a continuation of early discussions towards the PAR


• Suggested to use syntonization (frequency synchronization) instead of synchronization for CQF and its variance (CQF+).

• A measurement of phase offset is required to correctly match the cycles of two neighbor nodes.

• Goal of the slides
  • Potential impact in syntonized CQF/pulsed queues when the cycle time is small
Large receiving window time at the transmitter port may cause cycle identification ambiguity

- Assumptions:
  - per-hop latency variance between: Lmin & Lmax
  - Scheduled traffic takes full cycle time T
- Max receiving window time at the transmitter:
  \[(T + L_{max}) - (0 + L_{min})\]
  \[= T + (L_{max} - L_{min})\]
- # of receiving buffers to accommodate the max receiving window time:
  \[\text{floor}\left(\frac{T + (L_{max} - L_{min})}{T}\right) + 2\]
- # of sending buffer = 1
- Total # of buffers = \[\text{floor}\left(\frac{L_{max} - L_{min}}{T}\right) + 4\]
- Special case: depending on B’s phase offset relative to A, one buffer less may still work
- Example in left case:
  - \((L_{max} - L_{min})\) is 1.6T
  - Total # of buffers: 5 (or 4 in special case)
  - If \((L_{max} - L_{min})\) is < T, total # of buffers = 4 (or 3 in special case)
When such ambiguity has significant negative impact

• Cycle time $T$ is small, and
• Processing latency variance ($L_{\text{max}} - L_{\text{min}}$) at node is large so that it is comparable to $T$

• Recall:
  • When ($L_{\text{max}} - L_{\text{min}}$) $\ll T$, to eliminate the cycle ambiguity:
    • Make potential receiving window always fall in a single cycle:
      • increase the dead time (not allowing sending traffic) in a cycle to absorb the processing latency variance
  • Not full utilization of 100% of cycle time $T$ currently due to dead time
    • Dead time contributor 1: Guard band at the beginning of $T$ for the interruption from lower priority traffic
    • Dead time contributor 2: Dead time at the end of $T$ to absorb the latency variance and cycle shifting between two neighbor nodes
  • When $T$ is small, the dead time can eat $T$ up. This is not desired.
Cycle time $T$ can be small in the order of 10x $\mu$s

- **Rough factors to determine cycle time $T$**
  - allow at least one 1500B/max size packet to be sent within $T$
  - preferably multiple packets can be sent within $T$
  - $T < $ e2e bounded latency requirement / # of hops (roughly, not the exact number)

- **Existing CQF usage scenario: $T$ is no less than 100x $\mu$s**
  - Sufficiently good for streaming traffic as e2e bounded latency requirement is in the magnitude of few ms

- **Some observations:**
  - With increasing of link speed, the same amount of data can be transmitted within a smaller cycle time
  - Smaller cycle makes CQF+ applicable to more strict e2e bounded latency requirement usage scenarios.
    - Application period requirement in industry automation [60802-d1-2]: 100 $\mu$s to 2 ms (isochronous), 500 $\mu$s to 1 ms (Cyclic-synchronous), 2 to 20 ms (Cyclic-Asynchronous), 100 ms to 1s as latency (Alarms and Events), 50 ms to 1 s (network control traffic), latency < 2ms (video), latency < 100ms (Audio/Voice)

- **Cycle time in the order of magnitude of ~10x $\mu$s would be desired**
Processing latency variance at node is relatively large

• Store and forward time variance per packet

<table>
<thead>
<tr>
<th>Bit_Rate</th>
<th>Store and forward latency (us)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Min frame size (64B)</td>
<td>Max frame size (1518B)</td>
</tr>
<tr>
<td>100Mbps</td>
<td>64 *8 / 100Mbps ≈ 5 us</td>
<td>1518 *8 / 100Mbps ≈ 121 us</td>
</tr>
<tr>
<td>1Gbps</td>
<td>0.5</td>
<td>12.1</td>
</tr>
<tr>
<td>10Gbps</td>
<td>0.05</td>
<td>1.21</td>
</tr>
</tbody>
</table>

• Switch Fabric Latency in incast case
  • For n port switch, latency variance = (n-1) * frame_size / bit_rate

<table>
<thead>
<tr>
<th>Bit_Rate</th>
<th>Incast Latency variance (us)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>16-port switch</td>
<td>24-port switch</td>
</tr>
<tr>
<td>100Mbps</td>
<td>116 us *16 ≈ 1856 us</td>
<td>2784 us</td>
</tr>
<tr>
<td>1Gbps</td>
<td>185.6 us</td>
<td>278.4 us</td>
</tr>
<tr>
<td>10Gbps</td>
<td>18.6 us</td>
<td>27.8 us</td>
</tr>
</tbody>
</table>

• Processing latency variance is not negligible (or even larger) when cycle time T is ~10x us
Some Thoughts

• Cycle identification ambiguity is an issue when cycle T has to be small
• Consider the explicit cycle labeling to remove the ambiguity, especially when # of buffer >3
• Require the measurement of latency variance to estimate the potential receiving window time for a single cycle
  • # of buffers required
    • May need a guess at the very beginning for dry run
    • Adjust based on measurement to proper value
    • Monitor to check the violation of the assumed latency variance
  • Determine the cycle/buffer mapping relationship between neighbor nodes