DetServ: Network Models for Real-Time QoS Provisioning in SDN-based Industrial Environments

Amaury Van Bemten
amaury.van-bemten@tum.de

February, 28 2018 - WoNeCa 2018 – Erlangen

Based on:

©2018 Technical University of Munich
Wind power: what really happened when the Ayrshire turbine caught fire?

Dramatic picture was seized upon by opponents of wind energy but the same gales caused a much bigger nuclear power outage.

As Scots surveyed the damage after the fierce winds of 8 December last year, many newspapers carried a striking picture of a wind turbine in flames.

The drama at Ardrossan windfarm in North Ayrshire - which was caught on film - became a defining image of hurricane-force winds that peaked at 165mph, bringing down power lines and leaving about 60,000 people without electricity.

For some, the fire symbolised all that is wrong with wind turbines. Sir Bernard Ingham, secretary of the Supporters of Nuclear Energy group, said: “They are no good when the wind doesn’t blow and they are no good when the wind does blow.”
£2.000.000 turbine destroyed
Turbine stopped when wind too strong

60.000 people without electricity
Turbine disconnected from the grid when it fails
£2.000.000 turbine destroyed

Turbine stopped when wind too strong

60.000 people without electricity

Turbine disconnected from the grid when it fails

sensor – controller – actuator communication
Turbine stopped *when* wind too strong
Turbine disconnected from the grid \textit{when} it fails
£2,000,000 turbine destroyed

60,000 people without electricity

In order to avoid such losses and damages, sensor – controller – actuator communication must be deterministically delay bounded.
deterministically delay bounded

packet delay

delay bound

packets
Real-Time Quality of Service (QoS)

or industrial-grade QoS
SoA proprietary technologies are typically **costly** and not **interoperable**

- Fieldbus systems: CAN, PROFINET, PROFIBUS
- Industrial Ethernet: Modbus, EtherCAT, POWERLINK
- Prioritized Industrial Ethernet
- TDMA & Ring Systems
- Use SDN to provide real-time QoS with commodity hardware
- SDN

(cost: money, configuration time) vs (performance: resource efficiency)
Embed $f$ such that

i. its delay $t_f$ is guaranteed

ii. the guarantees provided to previously embedded flows are still valid
Embed $f$ such that

i. its **delay $t_f$ is guaranteed**

ii. the guarantees provided to previously embedded flows are **still valid**
Embed $f$ such that

i. its delay $t_f$ is guaranteed

ii. the guarantees provided to previously embedded flows are still valid.
Embed $f$ such that

i. its **delay** $t_f$ is guaranteed

ii. the guarantees provided to previously embedded flows are **still valid**
Embed $f$ such that
i. its delay $t_f$ is guaranteed
ii. the guarantees provided to previously embedded flows are still valid
Embed $f$ such that

i. its **delay $t_f$ is guaranteed**

ii. the guarantees provided to previously embedded flows are **still valid**
The delay of a route depends on

# the physical links
# how the flow is scheduled at each node

embed new flow $f$ with delay $t_f$
The delay of a route depends on

# the physical *links*

# how the flow is *scheduled* at each node
The delay of a route depends on

- the physical **links**
- how the flow is **scheduled** at each node
- the **queue** at which the flow is schedule at each node
The delay of a route depends on

# the physical links
# how the flow is scheduled at each node
# the queue at which the flow is scheduled at each node

assuming priority scheduling  
(cheap and ubiquitous)
The delay of a route depends on
- the physical **links**
- how the flow is **scheduled** at each node
- the **queue** at which the flow is scheduled at each node

assuming priority scheduling
(cheap and ubiquitous)
The delay of a route depends on

# the physical **links**

# how the flow is **scheduled** at each node

# the **queue** at which the flow is schedule at each node

assuming priority scheduling

(cheap and ubiquitous)

with, e.g., 3 priority queues
at each output port
The delay of a route depends on

- the physical **links**
- how the flow is scheduled at each node
- the **queue** at which the flow is scheduled at each node

assuming priority scheduling
(cheap and ubiquitous)

with, e.g., 3 priority queues at each output port

**QUEUE LINK TOPOLOGY**

Performing route selection on this topology defines both
The delay of a route depends on
# the physical links
# how the flow is scheduled at each node
# the queue at which the flow is scheduled at each node

assuming priority scheduling
(cheap and ubiquitous)

with, e.g., 3 priority queues at each output port

Queue Link Topology
Performing route selection on this topology defines both

Has to be done per queue-link
e.g., get delay of queue 2 of link AB
As we need deterministic delay guarantees,

deterministic network calculus

is a perfect candidate modeling tool!
assuming priority scheduling
(cheap and ubiquitous)

\[ R_{u,v} \]  \hspace{1cm} \text{capacity of link } (u, v) \\
\[ p \in [1, Q_{u,v}] \]  \hspace{1cm} \text{queue priorities (1 being highest)} \\
\[ (u, v, p) \]  \hspace{1cm} \text{queue with priority } p \text{ at link } (u, v) \\
\[ U_R[u, v, p] \]  \hspace{1cm} \text{rate of flow through queue } (u, v, p) \\
\[ U_B[u, v, p] \]  \hspace{1cm} \text{burst of flow through queue } (u, v, p) \\
\[ l_{u,v,p}^{\max} \]  \hspace{1cm} \text{maximum packet size through queue } (u, v, p) \\

The service curve for priority queue \((u, v, p)\) is given by

\[
\left( R_{u,v} t - t \sum_{j=1}^{p-1} U_R[u, v, j] - \sum_{j=1}^{p-1} U_B[u, v, j] - \max_{p+1 \leq j \leq Q_{u,v}} \{ l_{u,v,j}^{\max} \} - l_{u,v,p}^{\max} \right) +
\]
assuming priority scheduling
(cheap and ubiquitous)

\[ R_{u,v} \]

capacity of link \((u, v)\)

\[ p \in [1, Q_{u,v}] \]

queue priorities (1 being highest)

\[(u, v, p)\]

queue with priority \(p\) at link \((u, v)\)

\[ U_R[u, v, p] \]

rate of flow through queue \((u, v, p)\)

\[ U_B[u, v, p] \]

burst of flow through queue \((u, v, p)\)

\[ l_{u,v,p}^{\text{max}} \]

maximum packet size through queue \((u, v, p)\)

The service curve for priority queue \((u, v, p)\) is given by

\[
R_{u,v} - t \sum_{j=1}^{p-1} U_R[u, v, j] - \sum_{j=1}^{p-1} U_B[u, v, j] - \max_{p+1 \leq j \leq Q_{u,v}} \{ l_{u,v,j}^{\text{max}} \} - l_{u,v,p}^{\text{max}} \]

whole link
service

service used by
higher priority queues

one packet from a
lower priority queue
(non-preemptive scheduling)

per packet delay
The service curve for priority queue $(u, v, p)$ is given by

$$
\beta_{u,v,p} = \left( R_{u,v}t - t \sum_{j=1}^{p-1} U_R[u,v,j] - \sum_{j=1}^{p-1} U_B[u,v,j] - \max_{p+1 \leq j \leq Q_{u,v}} \{ l_{u,v,j}^{\max} \} - l_{u,v,p}^{\max} \right)^+ 
$$
In order to respect the QoS requirements of the flows,

The path $P_f$ of a flow $f$ must be chosen such that if fulfills the delay requirement $t_f$ of the flow,

$$\sum_{(u,v,p) \in P_f} T[u, v, p] \leq t_f$$

and we must ensure that no buffer overflow occurs

$$B_{max}(u, v, p) \leq A_B[u, v, p] \quad \forall (u, v, p)$$

Buffer capacity of a queue
In order to respect the **QoS requirements** of the flows,

The path $P_f$ of a flow $f$ must be chosen such that if fulfills the delay requirement $t_f$ of the flow,

\[ \sum_{(u,v,p) \in P_f} T[u, v, p] \leq t_f \]

and we must ensure that **no buffer overflow** occurs

\[ B_{\text{max}}(u, v, p) \leq A_B[u, v, p] \quad \forall (u, v, p) \]

Buffer capacity of a queue

**Must be fulfilled at all times,**
for all the flows
In order to respect the QoS requirements of the flows, the path $P_f$ of a flow $f$ must be chosen such that if fulfills the delay requirement $t_f$ of the flow,

$$\sum_{(u,v,p) \in P_f} T[u,v,p] \leq t_f$$

and we must ensure that no buffer overflow occurs

$$B_{\text{max}}(u,v,p) \leq A_B[u,v,p] \quad \forall (u,v,p)$$

We have...

$$T[u,v,p] = \frac{\sum_{j=1}^{p} U_B[u,v,j] + \max_{p+1 \leq j \leq Q_{u,v}} \{l_{u,v,j}^{\text{max}}\} + l_{u,v,p}^{\text{max}}}{R_{u,v} - \sum_{j=1}^{p-1} U_R[u,v,j]}$$

$$B_{\text{max}}(u,v,p) = U_B[u,v,p] + U_R[u,v,p] \frac{\sum_{j=1}^{p-1} U_B[u,v,j] + \max_{p+1 \leq j \leq Q_{u,v}} \{l_{u,v,j}^{\text{max}}\} + l_{u,v,p}^{\text{max}}}{R_{u,v} - \sum_{j=1}^{p-1} U_R[u,v,j]}$$
In order to respect the QoS requirements of the flows, the path $P_f$ of a flow $f$ must be chosen such that if fulfills the delay requirement $t_f$ of the flow,

$$\sum_{(u,v,p)\in P_f} T[u,v,p] \leq t_f$$

and we must ensure that no buffer overflow occurs

$$B_{max}(u,v,p) \leq A_{B}[u,v,p] \quad \forall (u,v,p)$$

We have...

Must be fulfilled at all times, for all the flows

Dependence on other flows embedded at the same link
Should we re-check all the previously embedded flows?

No, it does not scale!
Should we re-check all the previously embedded flows?

**No**, it does not scale!

→ Define **upper bounds** which are **independent of the state of the network**!
Should we re-check all the previously embedded flows?  

**No**, it does not scale!

→ Define **upper bounds** which are independent of the state of the network!

\[
\sum_{(u,v,p) \in P_f} T[u, v, p] \leq t_f \quad \forall f
\]

\[
B_{\text{max}}(u, v, p) \leq A_B[u, v, p] \quad \forall (u, v, p)
\]
Let’s find an upper bound independent of the network state...

\[ T[u, v, p] = \frac{\sum_{j=1}^{p} U_B[u, v, j] + \max_{p+1 \leq j \leq Q_{u,v}} \{l_{u,v,j}^{\max}\} + l_{u,v,p}^{\max}}{R_{u,v} - \sum_{j=1}^{p-1} U_R[u, v, j]} \]

\[ B_{max}(u, v, p) = U_B[u, v, p] + U_R[u, v, p] \frac{\sum_{j=1}^{p-1} U_B[u, v, j] + \max_{p+1 \leq j \leq Q_{u,v}} \{l_{u,v,j}^{\max}\} + l_{u,v,p}^{\max}}{R_{u,v} - \sum_{j=1}^{p-1} U_R[u, v, j]} \]
Let's find an upper bound independent of the network state...

Packets cannot be bigger than the biggest Ethernet frame size

\[
T[u, v, p] = \frac{\sum_{j=1}^{p} U_B[u, v, j] + 1542 + 1542}{R_{u,v} - \sum_{j=1}^{p-1} U_R[u, v, j]}
\]

\[
B_{\text{max}}(u, v, p) = U_B[u, v, p] + U_R[u, v, p] \frac{\sum_{j=1}^{p-1} U_B[u, v, j] + 1542 + 1542}{R_{u,v} - \sum_{j=1}^{p-1} U_R[u, v, j]}
\]
Let’s find an upper bound independent of the network state...

\( A_R[u, v, p] \) defined as the maximum rate that can be accepted at a queue

\[
T[u, v, p] = \frac{\sum_{j=1}^{p} U_B[u, v, j] + 1542 + 1542}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u, v, p]}
\]

\[
B_{max}(u, v, p) = U_B[u, v, p] + A_R[u, v, p] \frac{\sum_{j=1}^{p-1} U_B[u, v, j] + 1542 + 1542}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u, v, p]}
\]
Let's find an upper bound independent of the network state...

\[
T[u, v, p] = \frac{\sum_{j=1}^{p} U_B[u, v, j] + 1542 + 1542}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u, v, p]} 
\]

\[
B_{max}(u, v, p) = U_B[u, v, p] + \frac{\sum_{j=1}^{p-1} U_B[u, v, j] + 1542 + 1542}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u, v, p]} 
\]
Let's find an upper bound independent of the network state...

Limit bursts such that no buffer overflow occurs

\[ T[u, v, p] = \frac{\sum_{j=1}^{p} U_B[u, v, j] + 1542 + 1542}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u, v, p]} \]

\[ B_{max}(u, v, p) = U_B[u, v, p] + A_R[u, v, p] \frac{\sum_{j=1}^{p-1} U_B[u, v, j] + 1542 + 1542}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u, v, p]} \]

\[ \leq A_B[u, v, p] \]
Let's find an upper bound independent of the network state...

Limit bursts such that no buffer overflow occurs

\[
T[u, v, p] = \frac{\sum_{j=1}^{p} M_B[u, v, p] + 1542 + 1542}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u, v, p]}
\]

\[
B_{\text{max}}(u, v, p) = \frac{\sum_{j=1}^{p-1} M_B[u, v, p] + 1542 + 1542}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u, v, p]} \leq A_B[u, v, p]
\]
Let's find an upper bound independent of the network state...

Limit bursts such that no buffer overflow occurs

\[ T[u, v, p] = \frac{\sum_{j=1}^{p} M_B[u, v, p] + 1542 + 1542}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u, v, p]} \]

\[ B_{max}(u, v, p) = M_B[u, v, p] + A_R[u, v, p] \frac{\sum_{j=1}^{p-1} M_B[u, v, p] + 1542 + 1542}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u, v, p]} \]

\[ \leq A_B[u, v, p] \]

The maximum bursts \( M_B[u, v, p] \) can be computed recursively

\[ M_B[u, v, 1] + A_R[u, v, 1] \frac{3084}{R_{u,v}} = A_B[u, v, 1] \]
Let's find an upper bound independent of the network state...

Limit bursts such that no buffer overflow occurs

\[
T[u, v, p] = \frac{\sum_{j=1}^{p} M_B[u, v, p] + 1542 + 1542}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u, v, p]}
\]

\[
B_{\max}(u, v, p) = M_B[u, v, p] + A_R[u, v, p] \frac{\sum_{j=1}^{p-1} M_B[u, v, p] + 1542 + 1542}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u, v, p]}
\]

\[
\leq A_B[u, v, p]
\]

The maximum bursts \( M_B[u, v, p] \) can be computed recursively

\[
M_B[u, v, 2] + A_R[u, v, 2] \frac{M_B[u, v, 1] + 3084}{R_{u,v} - A_R[u, v, 1]} = A_B[u, v, 1]
\]

etc.
We have the following expression, independent of the state of the network,

\[ T[u, v, p] \leq \frac{\sum_{j=1}^{p} M_B[u, v, j]}{R_{u, v}} - \sum_{j=1}^{p-1} A_R[u, v, j] + 3084 \triangleq T^{MHM}[u, v, p] \]

defined per queue by a resource allocation algorithm computed recursively.
We have the following expression, independent of the state of the network,

\[ T[u, v, p] \leq \frac{\sum_{j=1}^{p} M_B[u, v, j]}{R_{u,v}} - \sum_{j=1}^{p-1} A_R[u, v, j] + 3084 \triangleq T^{MHM}[u, v, p] \]

computed recursively

defined per queue by a resource allocation algorithm

The opposite can actually also be done...
\[ T[u, v, p] \leq \frac{\sum_{j=1}^{p} M_B[u, v, j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u, v, j]} \triangleq T^{MHM}[u, v, p] \]

**The Multi-Hop Model (MHM)**

```
return \( T^{MHM}[u, v, p] \)
```

```
update \( U_R[u, v, p], U_B[u, v, p] \)
```

check \( M_B[u, v, p], A_R[u, v, p] \) are not exceeded

```
getDelay()
```

```
registerPath()
```

```
hasAccess()
```
Let’s see how this looks like graphically... at a link with 3 priority queues

\[
M_B[u, v, p] + A_R[u, v, p]\frac{\sum_{j=1}^{p-1} M_B[u, v, j] + 3084}{R_{u, v} - \sum_{j=1}^{p-1} A_R[u, v, j]} = A_B[u, v, p]
\]

\[
\beta_{u,v,p} = \left( R_{u,v}t - t \sum_{j=1}^{p-1} A_R[u, v, j] - \sum_{j=1}^{p-1} M_B[u, v, j] - 3084 \right)^+
\]

\[
T^{MHM}[u, v, p] = \frac{\sum_{j=1}^{p} M_B[u, v, j] + 3084}{R_{u, v} - \sum_{j=1}^{p-1} A_R[u, v, j]}
\]
Let's see how this looks like graphically... at a link with 3 priority queues

\[ \nabla = R_{u,v,1} = R_{u,v} \]

\[ \beta_{u,v,1} \]

\[ \begin{align*} 
\mathbf{M}_B[u,v,p] + \mathbf{A}_R[u,v,p] & \frac{\sum_{j=1}^{p-1} \mathbf{M}_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1} \mathbf{A}_R[u,v,j]} = \mathbf{A}_B[u,v,p] \\
\beta_{u,v,p} = \left( R_{u,v}t - t \sum_{j=1}^{p-1} \mathbf{A}_R[u,v,j] - \sum_{j=1}^{p-1} \mathbf{M}_B[u,v,j] - 3084 \right)^+ \\
\mathbf{T}^{MHM}[u,v,p] = \frac{\sum_{j=1}^{p} \mathbf{M}_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1} \mathbf{A}_R[u,v,j]} 
\end{align*} \]
Let's see how this looks like graphically... at a link with 3 priority queues

\[ \nabla = R_{u,v,1} = R_{u,v} \]

\[ \beta_{u,v,1} \]

\[ A_B[u,v,1] \]

\[ T_{u,v,1} \]

\[ \text{time} \]

\[ \text{high priority queue} \]

\[ M_B[u,v,p] + A_R[u,v,p] \frac{\sum_{j=1}^{p-1} M_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u,v,j]} = A_B[u,v,p] \]

\[ \beta_{u,v,p} = \left( R_{u,v} t - t \sum_{j=1}^{p-1} A_R[u,v,j] - \sum_{j=1}^{p-1} M_B[u,v,j] - 3084 \right)^+ \]

\[ T^{MHM}[u,v,p] = \frac{\sum_{j=1}^{p} M_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u,v,j]} \]
Let's see how this looks like graphically... at a link with 3 priority queues

\[ \nabla = R_{u,v,1} = R_{u,v} \]

\[ \beta_{u,v,1} \]

\[ \nabla = A_R[u,v,1] \]

\[
M_B[u,v,p] + A_R[u,v,p] \frac{\sum_{j=1}^{p-1} M_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u,v,j]} = A_B[u,v,p]
\]

\[
\beta_{u,v,p} = \left( R_{u,v} t - t \sum_{j=1}^{p-1} A_R[u,v,j] - \sum_{j=1}^{p-1} M_B[u,v,j] - 3084 \right)^+
\]

\[
T^{MHW}[u,v,p] = \frac{\sum_{j=1}^{p} M_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u,v,j]}
\]
Let’s see how this looks like graphically... at a link with 3 priority queues

\[ \nabla = R_{u,v,1} = R_{u,v} \]
\[ \beta_{u,v,1} \]
\[ \nabla = A_R[u, v, 1] \]

\[ M_B[u, v, 1] \]
\[ A_B[u, v, 1] \]

\[ T^{MHM}[u, v, 1] \]

\[ T_{u,v,1} \]

\[ M_B[u, v, p] + A_R[u, v, p] \frac{\sum_{j=1}^{p-1} M_B[u, v, j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u, v, j]} = A_B[u, v, p] \]

\[ \beta_{u,v,p} = \left( R_{u,v} t - t \sum_{j=1}^{p-1} A_R[u, v, j] - \sum_{j=1}^{p-1} M_B[u, v, j] - 3084 \right)^+ \]

\[ T^{MHM}[u, v, p] = \frac{\sum_{j=1}^{p} M_B[u, v, j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u, v, j]} \]
Let’s see how this looks like graphically... at a link with 3 priority queues

\[ \nabla = R_{u,v,1} = R_{u,v} \]

\[ \beta_{u,v,1} \]

\[ \nabla = A_R[u,v,1] \]

\[ M_B[u,v,1] \]

\[ A_B[u,v,1] \]

\[ T^{MM}[u,v,1] \]

\[ M_B[u,v,p] + A_R[u,v,p] \frac{\sum_{j=1}^{p-1} M_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u,v,j]} = A_B[u,v,p] \]

\[ \beta_{u,v,p} = \left( R_{u,v} t - t \sum_{j=1}^{p-1} A_R[u,v,j] - \sum_{j=1}^{p-1} M_B[u,v,j] - 3084 \right)^+ \]

\[ T^{MM}[u,v,p] = \frac{\sum_{j=1}^{p} M_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u,v,j]} \]
Let’s see how this looks like graphically... at a link with 3 priority queues

\[ \nabla = R_{u,v,1} = R_{u,v} \]

\[ \beta_{u,v,1} \]

\[ \nabla = A_R[u, v, 1] \]

\[ M_B[u, v, 1] \]

\[ A_B[u, v, 1] \]

\[ M_B[u, v, p] + A_R[u, v, p] \frac{\sum_{j=1}^{p-1} M_B[u, v, j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u, v, j]} = A_B[u, v, p] \]

\[ \beta_{u,v,p} = \left( R_{u,v} t - t \sum_{j=1}^{p-1} A_R[u, v, j] - \sum_{j=1}^{p-1} M_B[u, v, j] - 3084 \right)^+ \]

\[ T^{MHM}[u, v, p] = \frac{\sum_{j=1}^{p} M_B[u, v, j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u, v, j]} \]
Let’s see how this looks like graphically... at a link with 3 priority queues

\[ \nabla = R_{u,v,1} = R_{u,v} \]
\[ \beta_{u,v,1} \]
\[ \nabla = A_R[u, v, 1] \]

\[ M_B[u, v, 1] \]
\[ A_B[u, v, 1] \]

\[ T^{MHM}[u, v, 1] \]

\[ M_B[u, v, p] + A_R[u, v, p] \frac{\sum_{j=1}^{p-1} M_B[u, v, j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u, v, j]} = A_B[u, v, p] \]

\[ \beta_{u,v,p} = \left( R_{u,v}t - t \sum_{j=1}^{p-1} A_R[u, v, j] - \sum_{j=1}^{p-1} M_B[u, v, j] - 3084 \right)^+ \]

\[ T^{MHM}[u, v, p] = \frac{\sum_{j=1}^{p} M_B[u, v, j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u, v, j]} \]
Let’s see how this looks like graphically... at a link with 3 priority queues

\[ \nabla = R_{u,v,1} = R_{u,v} \]

\[ \beta_{u,v,1} \]

\[ \nabla = A_R[u,v,1] \]

\[ M_B[u,v,1] \]

\[ A_B[u,v,1] \]

\[ T_{u,v,1} \]

\[ T_{u,v,2} \]

\[ \text{time} \]

\[ \text{high priority queue} \]

\[ \text{data} \]

\[ M_B[u,v,p] + A_R[u,v,p] \frac{\sum_{j=1}^{p-1} M_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u,v,j]} = A_B[u,v,p] \]

\[ \beta_{u,v,p} = \left( R_{u,v} t - t \sum_{j=1}^{p-1} A_R[u,v,j] - \sum_{j=1}^{p-1} M_B[u,v,j] - 3084 \right)^+ \]

\[ T^{MHM}[u,v,p] = \frac{\sum_{j=1}^{p} M_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u,v,j]} \]
Let’s see how this looks like graphically... at a link with 3 priority queues

\[ \nabla = R_{u,v,1} = R_{u,v} \]

\[ \beta_{u,v,1} \]

\[ \nabla = A_R[u,v,1] \]

\[ M_B[u,v,1] \]

\[ A_B[u,v,1] \]

\[ T_{u,v,1} \]

\[ T_{u,v,2} \]

\[ \text{time} \]

\[ \text{data} \]

\[ \text{high priority queue} \]

\[ \nabla = A_R[u,v,2] \]

\[ \beta_{u,v,2} \]

\[ M_B[u,v,2] \]

\[ A_B[u,v,2] \]

\[ T_{u,v,2} \]

\[ \text{time} \]

\[ \text{medium priority queue} \]

\[ M_B[u,v,p] + A_R[u,v,p] \frac{\sum_{j=1}^{p-1} M_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u,v,j]} = A_B[u,v,p] \]

\[ \beta_{u,v,p} = \left( R_{u,v} t - t \sum_{j=1}^{p-1} A_R[u,v,j] - \sum_{j=1}^{p-1} M_B[u,v,j] - 3084 \right)^+ \]

\[ T^{MHM}[u,v,p] = \frac{\sum_{j=1}^{p} M_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u,v,j]} \]
Let’s see how this looks like graphically... at a link with 3 priority queues

\[ \nabla = R_{u,v,1} = R_{u,v} \]
\[ \beta_{u,v,1} \]
\[ \nabla = A_R[u, v, 1] \]

\[ M_B[u, v, 1] \]
\[ A_B[u, v, 1] \]

\[ T_{u,v,1} \]
\[ T_{u,v,2} \]

\[ \text{time} \]
\[ \text{high priority queue} \]

\[ T^{HMM}[u, v, 1] \]

\[ M_B[u, v, p] + A_R[u, v, p] \frac{\sum_{j=1}^{p-1} M_B[u, v, j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u, v, j]} = A_B[u, v, p] \]

\[ \beta_{u,v,p} = \left( R_{u,v} t - t \sum_{j=1}^{p-1} A_R[u, v, j] - \sum_{j=1}^{p-1} M_B[u, v, j] - 3084 \right)^+ \]

\[ T^{HMM}[u, v, p] = \frac{\sum_{j=1}^{p} M_B[u, v, j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u, v, j]} \]
Let's see how this looks like graphically... at a link with 3 priority queues

\[ \nabla = R_{u,v,1} = R_{u,v} \]

\[ \beta_{u,v,1} \]

\[ \nabla = A_R[u,v,1] \]

\[ M_B[u,v,1] \]

\[ A_B[u,v,1] \]

\[ T_{u,v,1} \]

\[ T_{u,v,2} \]

\[ \text{time} \]

\[ \text{high priority queue} \]

\[ M_B[u,v,p] + A_R[u,v,p] \]

\[ \frac{\sum_{j=1}^{p-1} M_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u,v,j]} = A_B[u,v,p] \]

\[ \beta_{u,v,p} = \left( R_{u,v}t - t \sum_{j=1}^{p-1} A_R[u,v,j] - \sum_{j=1}^{p-1} M_B[u,v,j] - 3084 \right) + \]

\[ T^{MHM}[u,v,p] = \frac{\sum_{j=1}^{p} M_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1} A_R[u,v,j]} \]

\[ \nabla = A_R[u,v,2] \]

\[ R_{u,v,2} = A_R[u,v,1] - A_R[u,v,1] \]

\[ A_B[u,v,2] \]

\[ M_B[u,v,2] \]

\[ T^{MHM}[u,v,2] \]

\[ T_{u,v,2} \]

\[ \text{medium priority queue} \]

\[ \nabla = A_R[u,v,3] \]

\[ R_{u,v,3} = R_{u,v} - A_R[u,v,1] - A_R[u,v,2] \]

\[ A_B[u,v,3] \]

\[ M_B[u,v,3] \]

\[ T^{MHM}[u,v,3] \]

\[ T_{u,v,3} \]

\[ \beta_{u,v,3} \]

\[ \text{low priority queue} \]
Let’s see how this looks like graphically... at a link with 3 priority queues

**The Multi-Hop Model (MHM)**

- return $T^{MHM}_{u,v,p}$
- getDelay()
- update $U_R[u,v,p], U_B[u,v,p]
- registerPath()
- check $M_B[u,v,p], A_R[u,v,p]
- hasAccess()

Network Model

\[ \nabla = R_{u,v,1} = R_{u,v} \]

\[ \beta_{u,v,1} \]

\[ \nabla = A_R[u,v,1] \]

\[ M_B[u,v,1] \]

\[ A_B[u,v,1] \]

\[ T_{u,v,1} \]

\[ T_{u,v,2} \]

\[ T^{MHM}_{u,v,1} \]

\[ \beta_{u,v,2} \]

\[ \nabla = A_R[u,v,2] \]

\[ M_B[u,v,2] \]

\[ A_B[u,v,2] \]

\[ T^{MHM}_{u,v,2} \]

\[ \beta_{u,v,3} \]

\[ \nabla = A_R[u,v,3] \]

\[ M_B[u,v,3] \]

\[ T^{MHM}_{u,v,3} \]

\[ \nabla = R_{u,v,3} = R_{u,v} - A_R[u,v,1] - A_R[u,v,2] \]

\[ \nabla = A_R[u,v,3] \]
In such a situation, the MHM leads to a waste of resources. The buffer budget will never be used! because the rate blocks acceptance of other flows.
A solution is to **artificially reduce the buffer budget**!

This also reduces the delay of the queue, and hence the lower priority queues can have:
- a **lower delay**, or
- a **higher burst** budget, or
- a **higher data rate** budget
The resource allocation hence has to allocate

- $A_R[u, v, p]$ : a data rate
- $M_B[u, v, p]$ : a buffer capacity

to each queue in the network.

The resource allocation algorithm is responsible for adjusting a priori, the trade-off between resources.

The quality of this choice depends on the type of flows

→ bursty traffic? rate demanding traffic? low delay?

delay/burst/data rate budget trade-off
Can we do this differently?

We have to find an upper bound independent of the network state...

\[ T[u, v, p] = \frac{\sum_{j=1}^{p} U_B[u, v, j] + \max_{p+1 \leq j \leq Q_{u,v}} \{l_{u,v,j}^{\max}\} + l_{u,v,p}^{\max}}{R_{u,v} - \sum_{j=1}^{p-1} U_R[u, v, j]} \]

The MHM does this by bounding \( A_R[u, v, p] \), \( M_B[u, v, p] \) and \( l_{u,v,p}^{\max} \)

The resource allocation algorithm can rather bound the delay itself

\[ T[u, v, p] = \frac{\sum_{j=1}^{p} U_B[u, v, j] + \max_{p+1 \leq j \leq Q_{u,v}} \{l_{u,v,j}^{\max}\} + l_{u,v,p}^{\max}}{R_{u,v} - \sum_{j=1}^{p-1} U_R[u, v, j]} \leq T^{TBM}[u, v, p] \]

and let everything vary as long as

\[ T[u, v, p] \leq T^{TBM}[u, v, p] \quad \forall (u, v, p) \]

\[ B_{\max}(u, v, p) \leq A_B[u, v, p] \quad \forall (u, v, p) \]
The resource allocation algorithm can rather bound the delay itself

\[ T[u, v, p] = \frac{\sum_{j=1}^{p} U_B[u, v, j] + \max_{p+1 \leq j \leq Q_{u,v}} \{ l_{u,v,j}^{max} \} + l_{u,v,p}^{max}}{R_{u,v} - \sum_{j=1}^{p-1} U_R[u, v, j]} \leq T^{TBM}[u, v, p] \]

and let everything vary as long as

\[ T[u, v, p] \leq T^{TBM}[u, v, p] \quad \forall (u, v, p) \]

\[ B_{max}(u, v, p) = U_B[u, v, p] + U_R[u, v, p] \]

\[ \leq A_B[u, v, p] \quad \text{buffer capacity} \]
The resource allocation algorithm can rather bound the delay itself

$$T[u, v, p] = \frac{\sum_{j=1}^{p} U_B[u, v, j] + \max_{p+1 \leq j \leq Q_{u, v}} \{l_{u, v, j}^{\text{max}}\} + l_{u, v, p}^{\text{max}}}{R_{u, v} - \sum_{j=1}^{p-1} U_R[u, v, j]} \leq T^{TBM}[u, v, p]$$

and let everything vary as long as

$$T[u, v, p] \leq T^{TBM}[u, v, p] \quad \forall (u, v, p)$$

$$B_{\text{max}}(u, v, p) \leq A_B[u, v, p] \quad \forall (u, v, p)$$

$$B_{\text{max}}(u, v, p) = U_B[u, v, p] + U_R[u, v, p] - \frac{\sum_{j=1}^{p-1} U_B[u, v, j] + \max_{p+1 \leq j \leq Q_{u, v}} \{l_{u, v, j}^{\text{max}}\} + l_{u, v, p}^{\text{max}}}{R_{u, v} - \sum_{j=1}^{p-1} U_R[u, v, j]} \leq A_B[u, v, p] \quad \rightarrow \text{buffer capacity}$$

Requires to check **lower priority queues** and **higher priority queues** before the addition of a new flow
The resource allocation algorithm can rather bound the delay itself

\[ T[u, v, p] = \frac{\sum_{j=1}^{p} U_B[u, v, j] + 1542 + l_{u,v,p}^{\max}}{R_{u,v} - \sum_{j=1}^{p-1} U_R[u, v, j]} \leq T^{TBM}[u, v, p] \]

and let everything vary as long as

\[ T[u, v, p] \leq T^{TBM}[u, v, p] \quad \forall (u, v, p) \]

\[ B_{\max}(u, v, p) = U_B[u, v, p] + U_R[u, v, p] \leq A_B[u, v, p] \quad \forall (u, v, p) \]

\[ B_{\max}(u, v, p) = \frac{\sum_{j=1}^{p-1} U_B[u, v, j] + 1542 + l_{u,v,p}^{\max}}{R_{u,v} - \sum_{j=1}^{p-1} U_R[u, v, j]} \leq A_B[u, v, p] \]

Requires to check **lower priority queues and higher priority queues** before the addition of a new flow because there might be unknown best-effort traffic in the lowest priority queue.
The Threshold-based Model (TBM)

\[
\text{return } T^{TBM}[u, v, p] \\
\text{update } U_R[u, v, p], U_B[u, v, p], l_{u, v, p}^{\text{max}} \\
\text{check } T^{TBM}[u, v, q], A_B[u, v, q] \quad \forall q \geq p
\]
The Threshold-based Model (TBM)

return $T_{TBM}^{[u, v, p]}$

update $U_R[u, v, p], U_B[u, v, p], l_{u, v, p}^{max}$

check $T_{TBM}^{[u, v, q]}, A_B[u, v, q] \forall q \geq p$

Network Model

getDelay()

registerPath()

hasAccess()

Requires up to $Q_{u,v}$ times more work than the MHM

But no a priori choice on the burst/rate/delay trade-off
Let's see how this looks like graphically... at a link with 3 priority queues.
Let's see how this looks like graphically... at a link with 3 priority queues
Let's see how this looks like graphically... at a link with 3 priority queues.
Let’s see how this looks like graphically... at a link with 3 priority queues.

\[ \Delta = R_{u,v,1} \]

\[ \Delta = U_{R[u,v,1]} \]

\[ T[u,v,1] \]

\[ B_{\text{max}}(u,v,1) \]

\[ A_B[u,v,1] \]

\[ U_B[u,v,1] \]

\[ A_T[u,v,1] \]

\[ T_{u,v,1} \]

\[ A_B[u,v,2] \]

\[ A_B[u,v,3] \]

\[ A_T[u,v,2] \]

\[ A_T[u,v,3] \]

\[ B_{\text{max}}(u,v,2) \]

\[ B_{\text{max}}(u,v,3) \]

\[ \beta_{u,v,1} \]
Let's see how this looks like graphically... at a link with 3 priority queues.
Let's see how this looks like graphically... at a link with 3 priority queues
Let's see how this looks like graphically... at a link with 3 priority queues.
Let’s see how this looks like graphically... at a link with 3 priority queues.
Let’s see how this looks like graphically... at a link with 3 priority queues

Let’s try to add a new flow here
Let’s see how this looks like graphically... at a link with 3 priority queues.

Let’s try to add a new flow here.
Let’s see how this looks like graphically... at a link with 3 priority queues

Let’s try to add a new flow here

Impossible: the deadline of this queue would be violated
Let's see how this looks like graphically... at a link with 3 priority queues

Let's try to add a new flow here

Impossible: the deadline of this queue would be violated
Let's see how this looks like graphically... at a link with 3 priority queues

Let's try to add another flow here
Let’s see how this looks like graphically... at a link with 3 priority queues

\[ \Delta = R_{u,v,1} \]
\[ \beta_{u,v,1} \]
\[ \nabla = U_{R[u,v,1]} \]

Let’s try to add another flow here
Let's see how this looks like graphically... at a link with 3 priority queues.

Let's try to add another flow here.

**OKAY:** the delay of all queues is still satisfied.
The Multi-Hop Model (MHM)

return $T_{MHM}^{u,v,p}$

update $U_{R}[u,v,p], U_{B}[u,v,p]

check $M_B[u,v,p], A_{R}[u,v,p]$ are not exceeded

The Threshold-based Model (TBM)

return $T_{TBM}^{u,v,p}$

update $U_{R}[u,v,p], U_{B}[u,v,p], l_{u,v,p}^{max}$

check $T_{TBM}^{u,v,q}, A_{B}[u,v,q]$ $\forall q \geq p$
**The Multi-Hop Model (MHM)**

\[
\text{return } T^{MHM}[u, v, p]
\]

\[
\text{update } U_R[u, v, p], U_B[u, v, p]
\]

check \( M_B[u, v, p], A_R[u, v, p] \) are not exceeded

**The Threshold-based Model (TBM)**

\[
\text{return } T^{TBM}[u, v, p]
\]

\[
\text{update } U_R[u, v, p], U_B[u, v, p], t_{u,v,p}^{\text{max}}
\]

check \( T^{TBM}[u, v, q], A_B[u, v, q] \) \( \forall q \geq p \)
The Multi-Hop Model (MHM)

\[
\text{return } T^{MHM}[u, v, p]
\]

update \( U_R[u, v, p], U_B[u, v, p] \)

check \( M_B[u, v, p], A_R[u, v, p] \) are not exceeded

The Threshold-based Model (TBM)

\[
\text{return } T^{TBM}[u, v, p]
\]

update \( U_R[u, v, p], U_B[u, v, p], l_{u,v,p}^{\max} \)

check \( T^{TBM}[u, v, q], A_B[u, v, q] \) \( \forall q \geq p \)
Evaluation of the models
Evaluation of the MHM in a real wind park setup

Model running on top of OpenDaylight

Network gradually congested until the MHM rejects all the flows between SW0 and SW2

SW0-SW2-SW0 flow with a 12ms deadline
Evolution of the **packet delay** for the **SW0-SW2-SW0 flow**

Delay border never violated, no packet loss

Delay guarantee 12ms
Evolution of the **packet delay** for the **SW0-SW2-SW0 flow**

Delay border never violated, no packet loss

- Delay guarantee 12ms
- Delay increasing while increasing cross traffic
- Cross traffic stopped
Simulation of the MHM and TBM

4 queues, various topologies, various routing procedures, various delay constraints

The performance of the models is highly dependent on the other routing/resource allocation algorithms.

TBM around \( Q_{u,v} \) times slower

TBM potential to perform better but depends on how the routing and resource allocation algorithms avoid the blocking problem

The performance of the models is highly dependent on the other routing/resource allocation algorithms.
Unicast QoS Routing Algorithms for SDN:
A Comprehensive Survey and Performance Evaluation
Jochen W. Goek, Aramya Van Benten, Martin Reinsehlen, Fellow, IEEE, and Wolfgang Kellerer, Senior Member, IEEE

I. INTRODUCTION

A. Topic Area: Routing Algorithms for QoS Networking

Routing, i.e., determining a route (path) from a source node to a destination node through a sequence of intermediate switching nodes, is an elementary function of the network layer in communication networks. Given the importance of routing for communication networks, a diverse array of routing algorithms have been designed. Many routing algorithms have been specifically designed for specific network settings or applications, see Section 3.C.

Providing quality of service (QoS) is an important requirement for a wide range of communication network settings and applications. For instance, multimedia network applications require QoS from the network service, as do many network applications in industrial networks [1] and the smart grid [2] as well as network control systems [3]. The required QoS is often in the form of delay bounds (constraints) for the data packets traversing the network. Accordingly, extensive research has developed routing algorithms that satisfy given delay constraints while minimizing some cost metric, i.e., so-called delay-constrained least-cost (DCLC) routing algorithms. DCLC routing algorithms and similar routing algorithms that support QoS networking are often referred to as QoS-aware [3] (QoA) routing algorithms.

Generally, the route determination (computation) is either carried out in distributed modes, e.g., for control modules in individual distributed Internet Protocol (IP) routers, or by a centralized controller, e.g., a Software-Defined Networking (SDN) controller [4-18]. Distributed routing algorithms have been intensely researched for traditional IP routing, e.g., [9-11], and more recently for all-IP networks, see [12-16]. In the mid-1990s, the development of QoS Paradigms for the Internet, see [17]-[22], led to a renewed interest in examined routing and spurred the development of a plethora of QoS routing algorithms, which mainly targeted distributed computation. In sharp contrast, the emergence of the Software-Defined Networking (SDN) paradigm [23, [24] has shifted the research focus to centralized network control, including centralized routing computations [23]-[30]. The present work concentrates on the QoS routing for SDN.

Alphanumeric—Industrial networks require real-time guarantees for the flows they carry. That is, there are hard end-to-end delay requirements that have to be deterministically guaranteed. While proprietary extensions of Ethernet have provided solutions, these often require expensive forwarding devices. The rise of software-defined networking (SDN) opens the door to the design of centralized traffic engineering frameworks for providing such real-time guarantees. As part of such a framework, a network model is needed for the computation of worst-case delays and for traffic control. In this paper, we propose two network models based on network calculus theory for providing deterministic services (DetServ). While our first model, the multi-Ack model [21], designs a rate and a buffer budget to each queue in the network, our second model, the throughput-based model [21], simply uses a maximum delay for each queue. Via a worst-case analysis, we show that the proposed models are not violated and that no packet loss occurs. Further, we show that the Throughput model provides more flexibility than the characteristics of the flows to be embedded and that it has the potential of supporting more flows in a network. Finally, we show that the runtime cost for this increase in flexibility stays reasonable and is tractable for request processing in industry networks.

Index Terms—Access control, multi-dimensional network model, network calculus, quality of service (QoS), software-defined networking (SDN).

A. Motivation: Industrial Networking Quality of Service

Industrial communications (e.g., machine-to-machine (M2M) communications or production facilities networks) have strict Quality of Service (QoS) requirements, mainly in terms of end-to-end delay [1]. This means that flows have end-to-end delays that must not be exceeded. In this article, such flows are referred to as real-time flows. A wide range of proprietary solutions [2] and extensions of Ethernet [3] have been developed for providing this strict QoS. However, these solutions typically require changes within the network protocol stack or impose restrictions on the topology that can be deployed, which leads to expensive forwarding devices.

B. Basis: Centralized Frameworks Based on Software-Defined Networking

Software-Defined Networking (SDN) is a new networking paradigm that run control functions on a centralized controller which is then able to program the forwarding elements in the network using a standardized interface such as OpenFlow [4]. This central view offered by SDN allows to perform traffic engineering based on the global knowledge of the network. Because it only requires simple commodity forwarding elements that can be changed and updated independently [5], SDN is considered as an fine-grained solution. Therefore, as elaborated in Section II, a plethora of work has been considering the usage of SDN for the provision of QoS [6]-[18]. However, the QoS control provided by these approaches is either too inaccurate or too coarse for industrial applications [18]. As initiated by Jager et al. [19], Goek et al. [16]-[18] propose to overcome the above-mentioned shortcomings by using network calculus, a mathematical modeling framework (introduced in Section III), to maintain a deterministic model of the network state in the control plane. Fine, network calculus being a deterministic framework, accurate bounds can be computed on a per-flow basis. Second, keeping a deterministic model in the control plane allows to avoid the QoS control loop to go through the forwarding plane, thereby allowing to quickly provide new network requests [7]. As such, the two drawbacks of existing approaches are overcome.


As elaborated in Section IV, a centralized industrial QoS framework requires a network model for the computation of worst-case delays and for access control. The core contribution of this article consists of two network models that can be used as part of such QoS frameworks for providing deterministic solutions typically require changes within the network protocol stack or impose restrictions on the topology that can be deployed, which leads to expensive forwarding devices.
References
