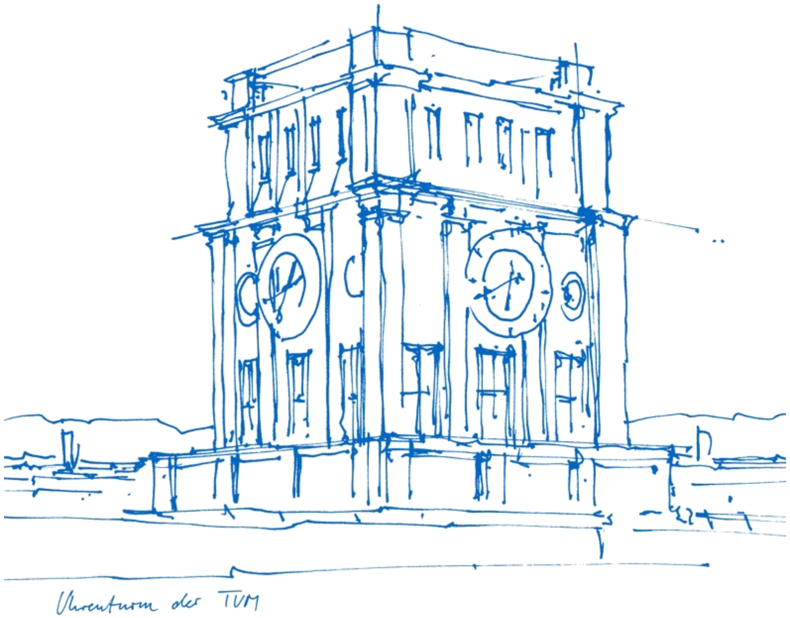
Chair of Communication Networks
Departement of Electrical and Computer Engineering
Technische University of Munich

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:
J. W. Guck, A. Van Bemten and W. Kellerer, "DetServ: Network Models for Real-Time QoS Provisioning in SDN-Based Industrial Environments," in IEEE Transactions on Network and Service Management, Dec. 2017.



Support The Guardian | Subscribe | Find a job | Sign in | Search | International edition

News | Opinion | Sport | Culture | Lifestyle | More

The Guardian

Environment ▸ Climate change | Wildlife | **Energy** | Pollution

Wind power

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

Michael MacLeod
Tue 28 Feb 2012 10.00 GMT

This article is 6 years old

18

▲ A wind turbine bursts into flames as storms tear across Scotland. Photograph: Stuart McMahon/Universal News and Sport

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 electrity

Turbine disconnected from the grid when it fails

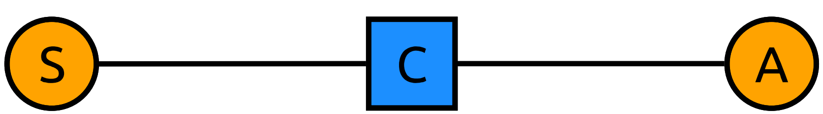
£2.000.000 turbine destroyed

Turbine stopped when wind too strong

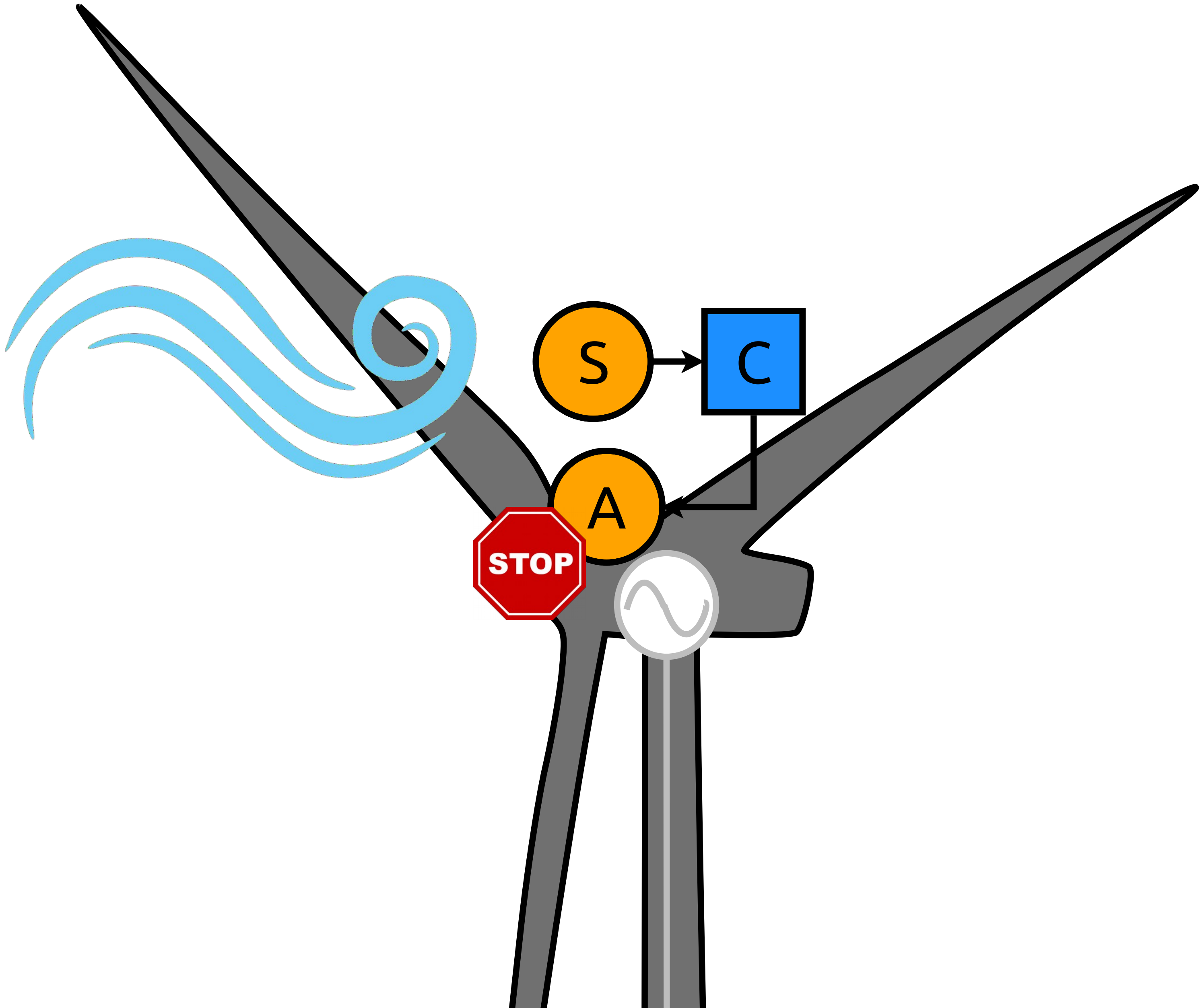
60.000 people without electrity

Turbine disconnected from the grid when it fails

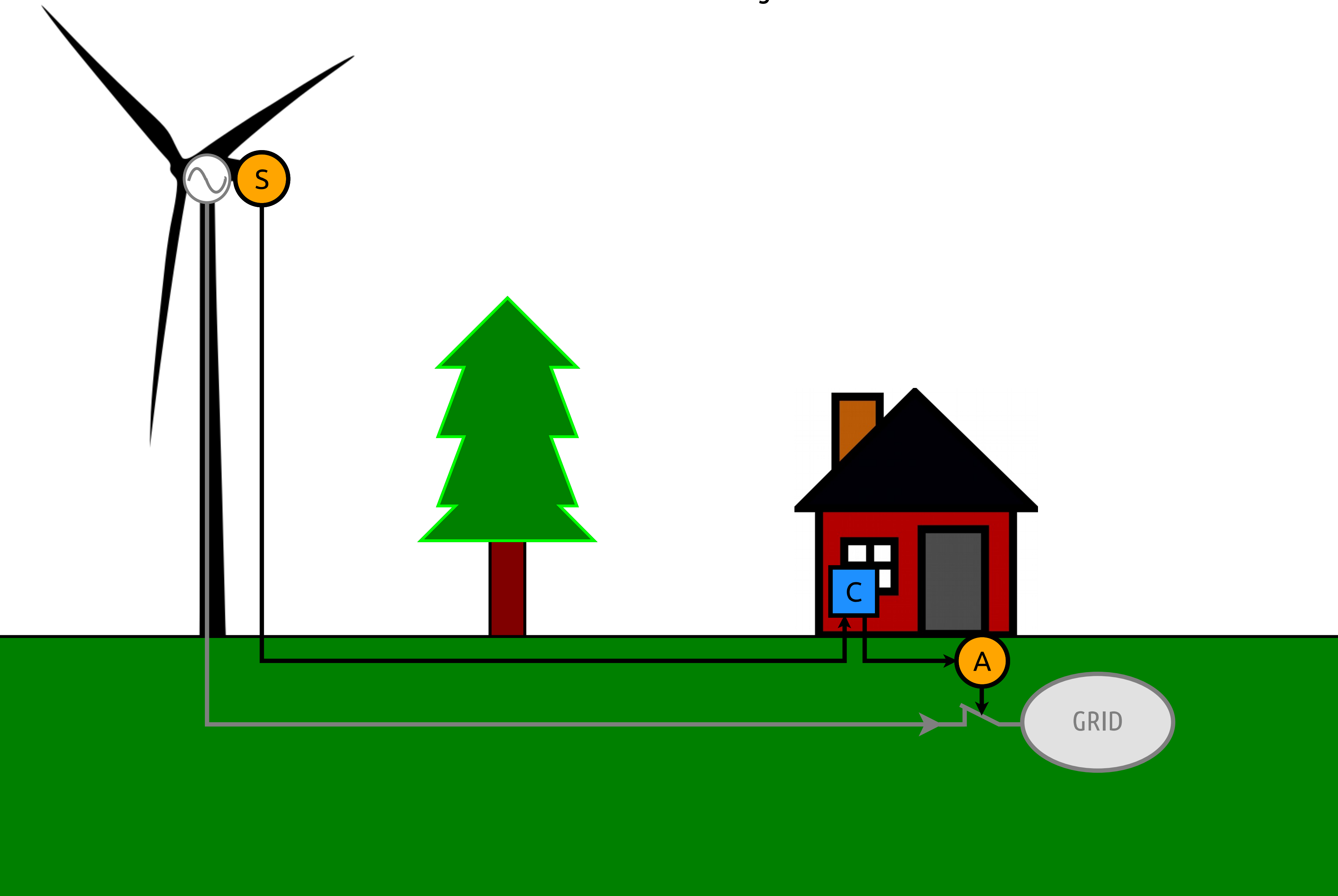
sensor – controller – actuator communication



Turbine stopped when wind too strong



Turbine disconnected from the grid when it fails



£2.000.000 turbine destroyed

60.000 people without electrity

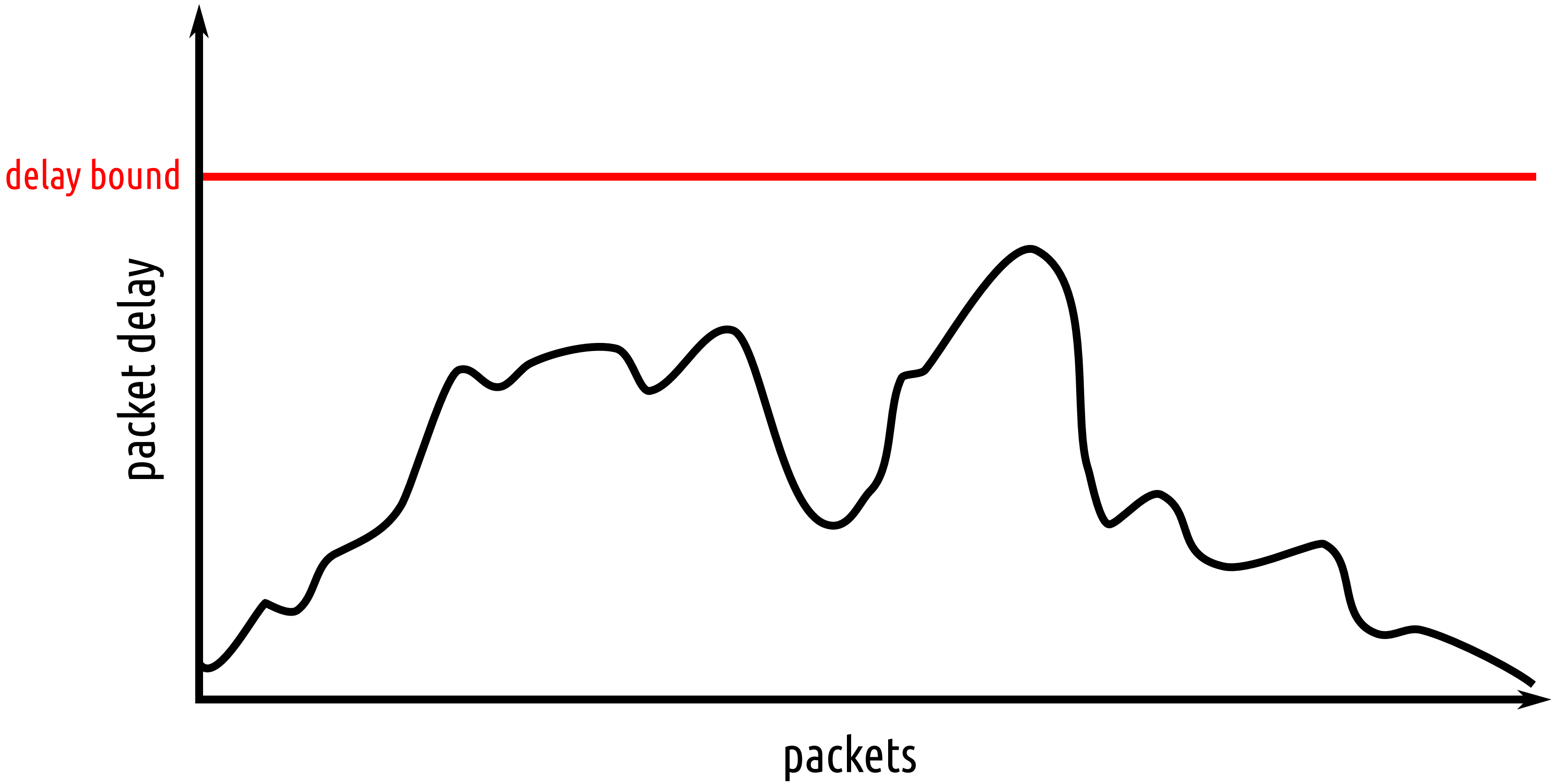
In order to avoid such losses and damages

sensor – controller – actuator communication

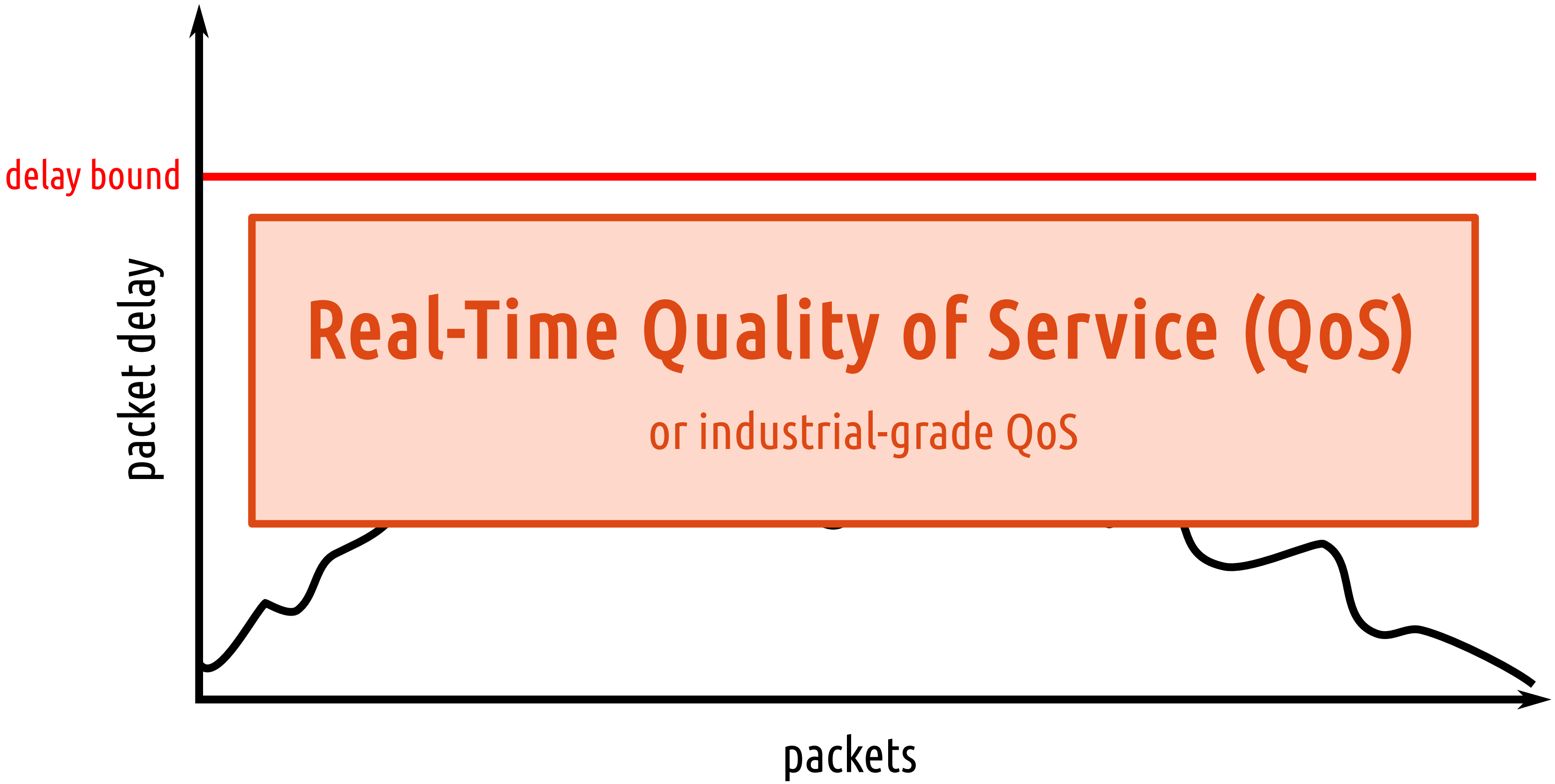
must be

deterministically delay bounded

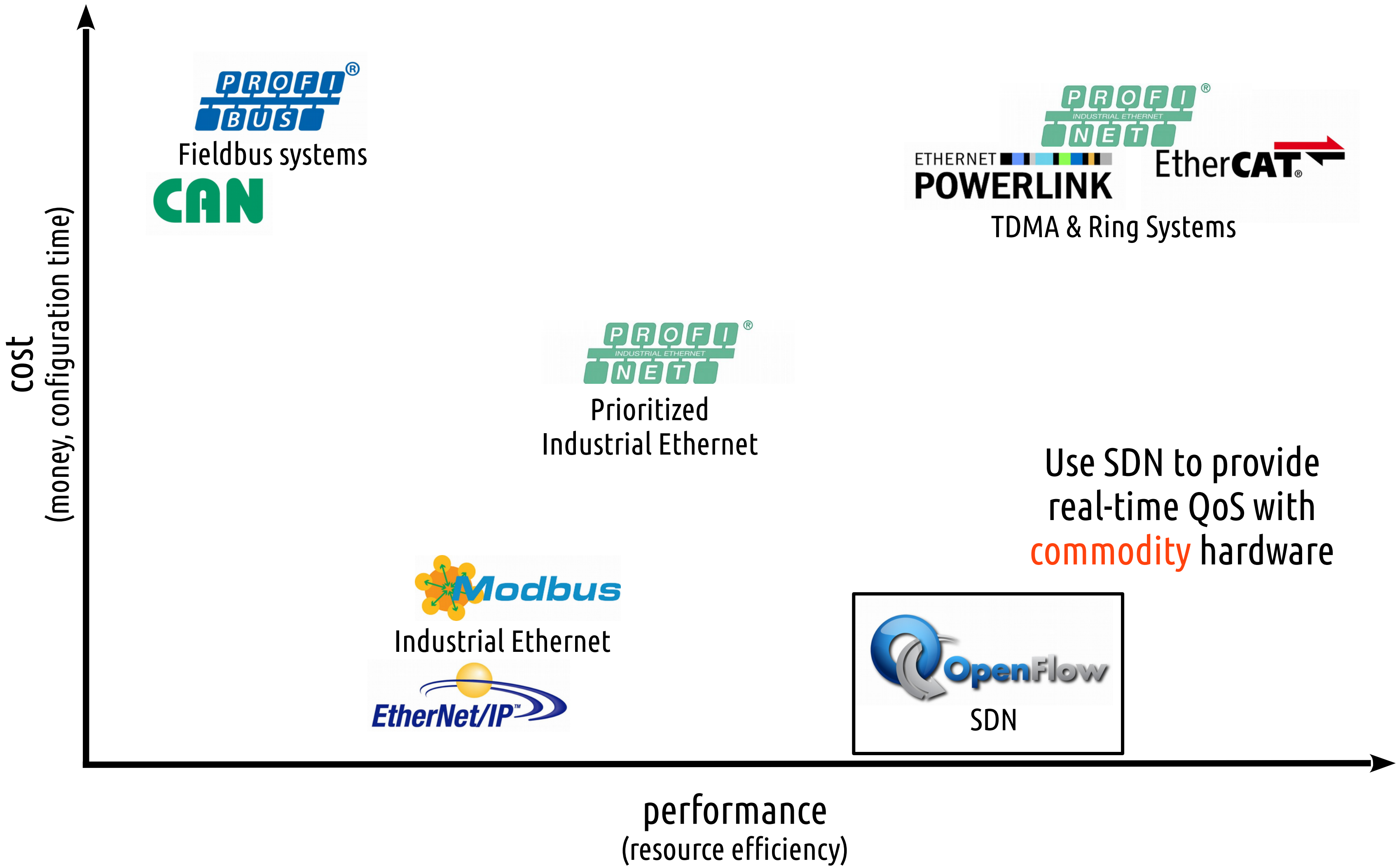
deterministically delay bounded

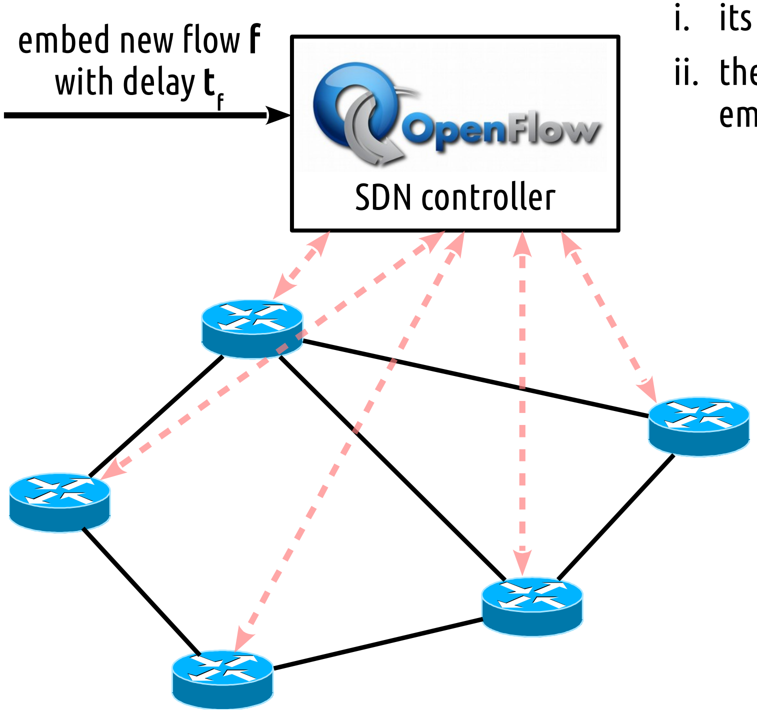


deterministically delay bounded



SoA proprietary technologies are typically costly and not interoperable





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 new flow **f** with delay $\mathbf{t}_f$

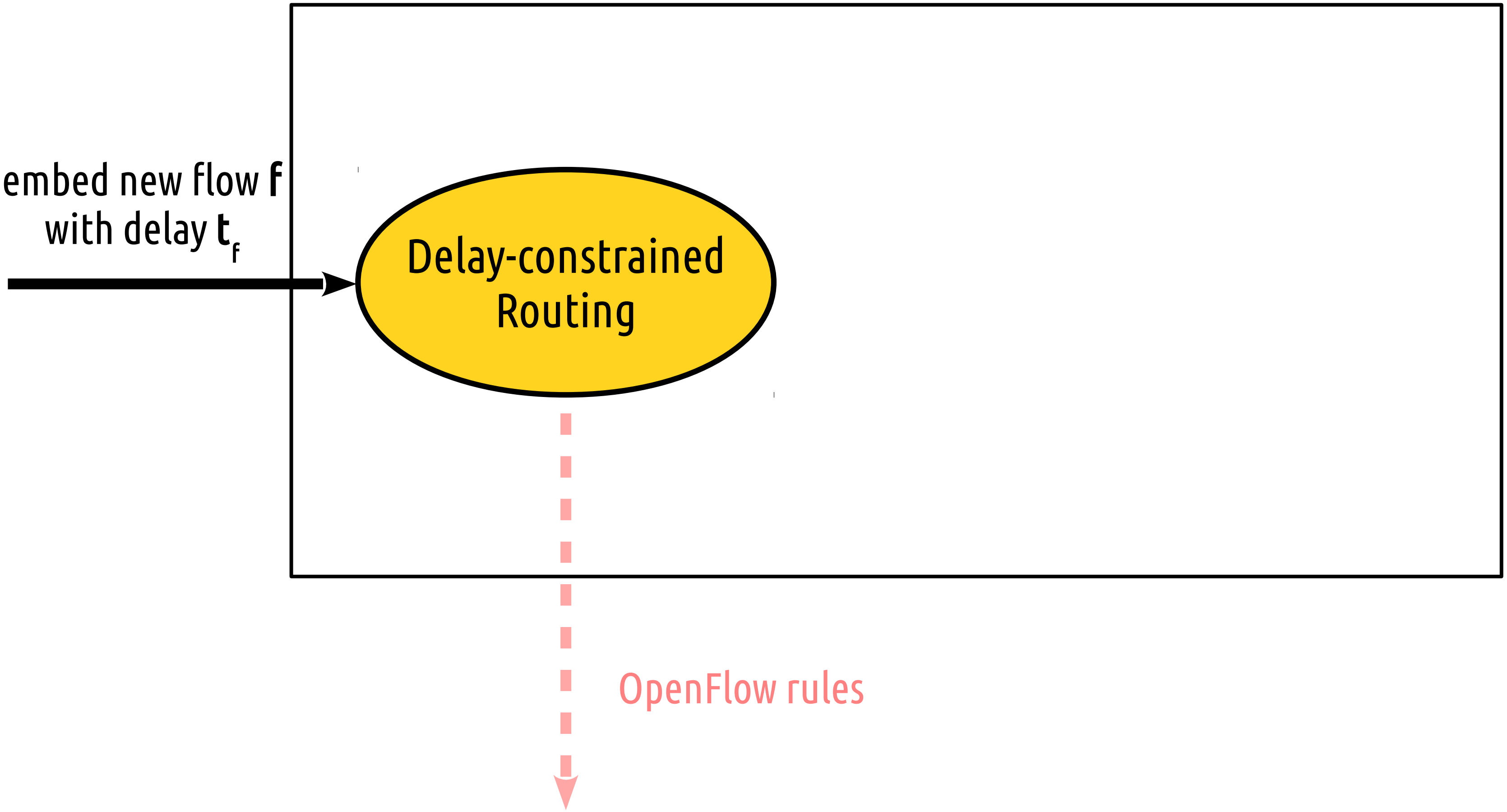
SDN controller

OpenFlow rules

Embed **f** such that

i. its **delay t_f is guaranteed**

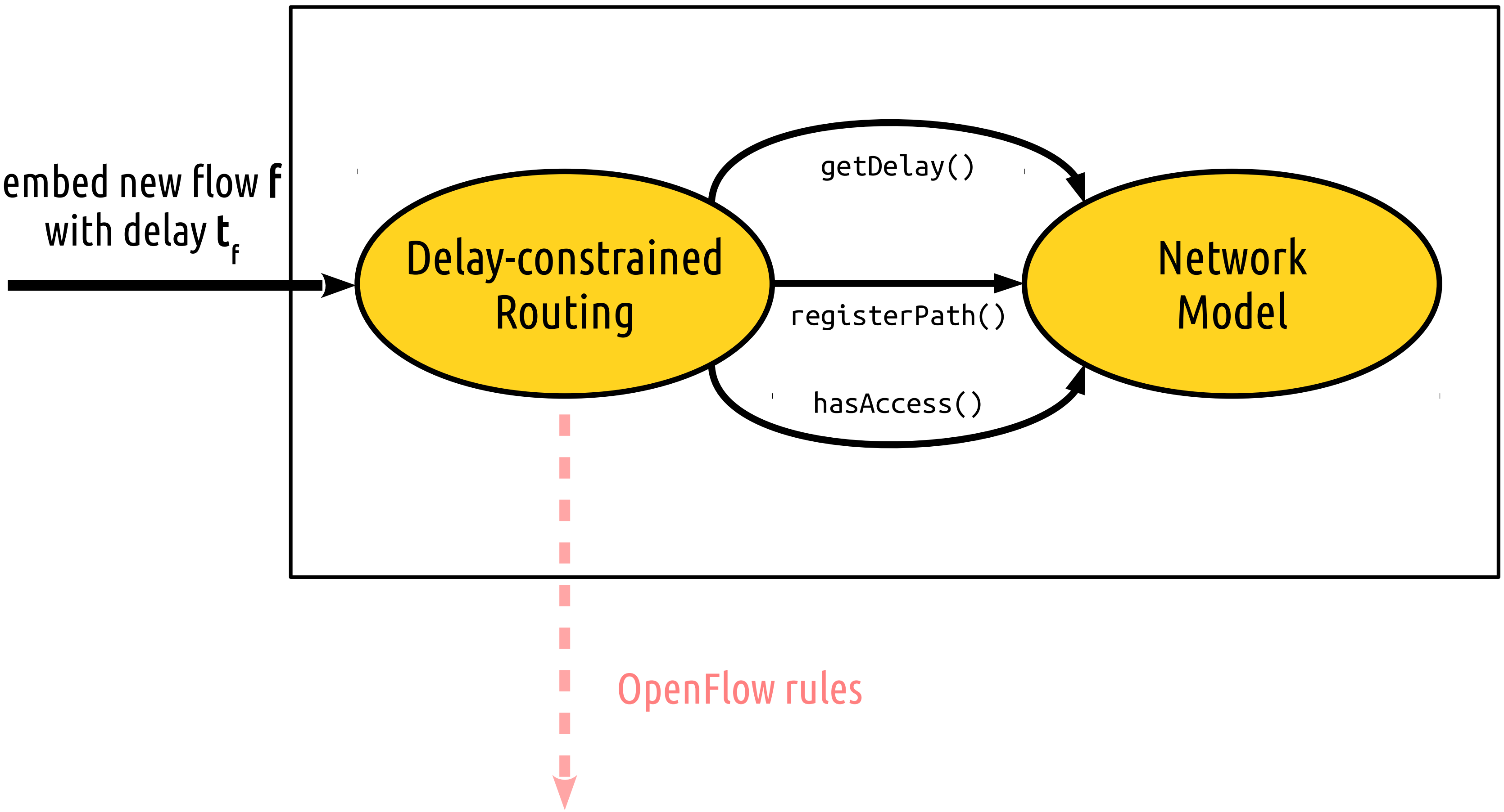
ii. the guarantees provided to previously embedded flows are **still valid**



Embed **f** such that

i. its **delay t_f is guaranteed**

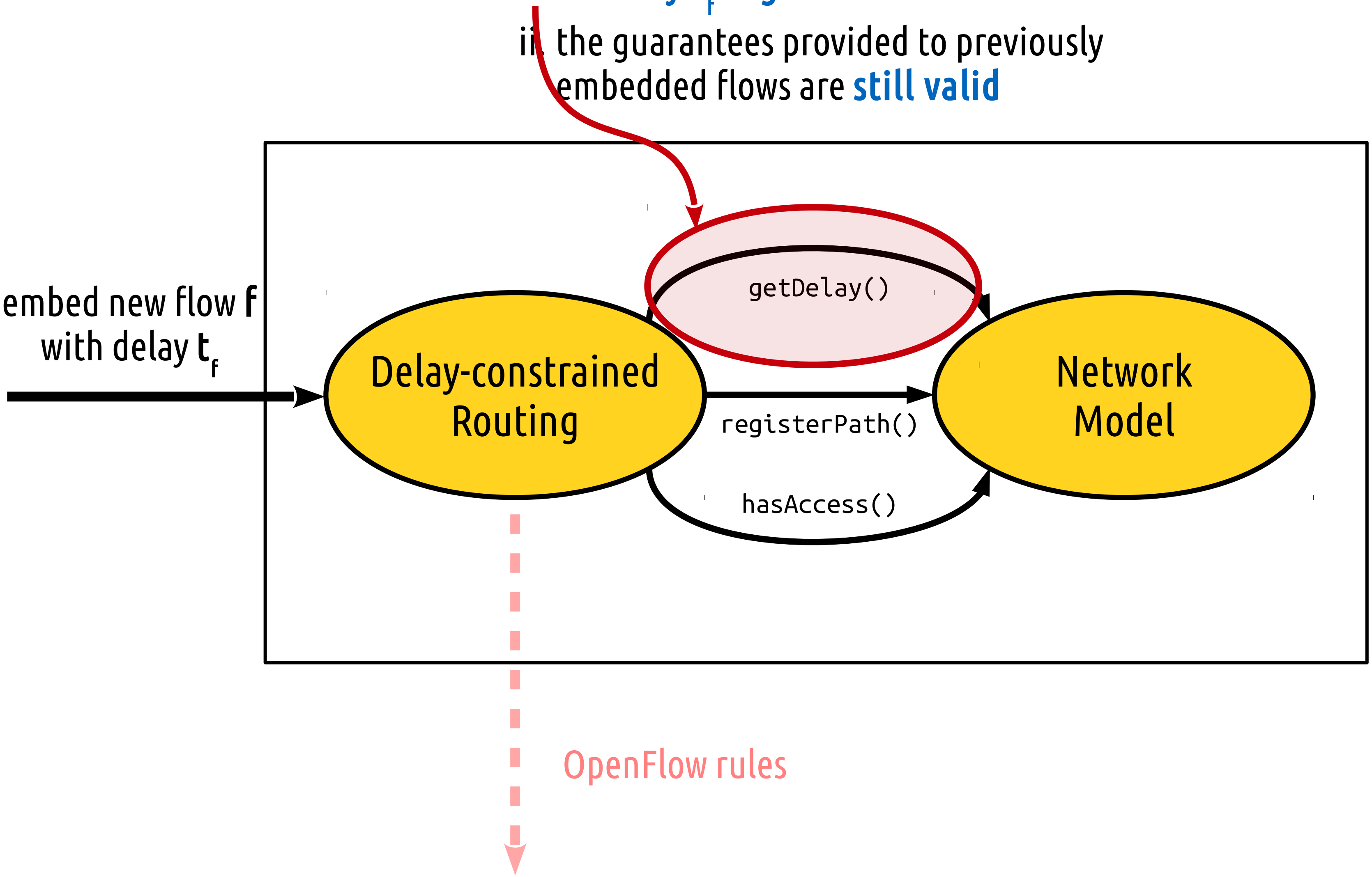
ii. the guarantees provided to previously embedded flows are **still valid**



Embed **f** such that

i. its **delay t_f is guaranteed**

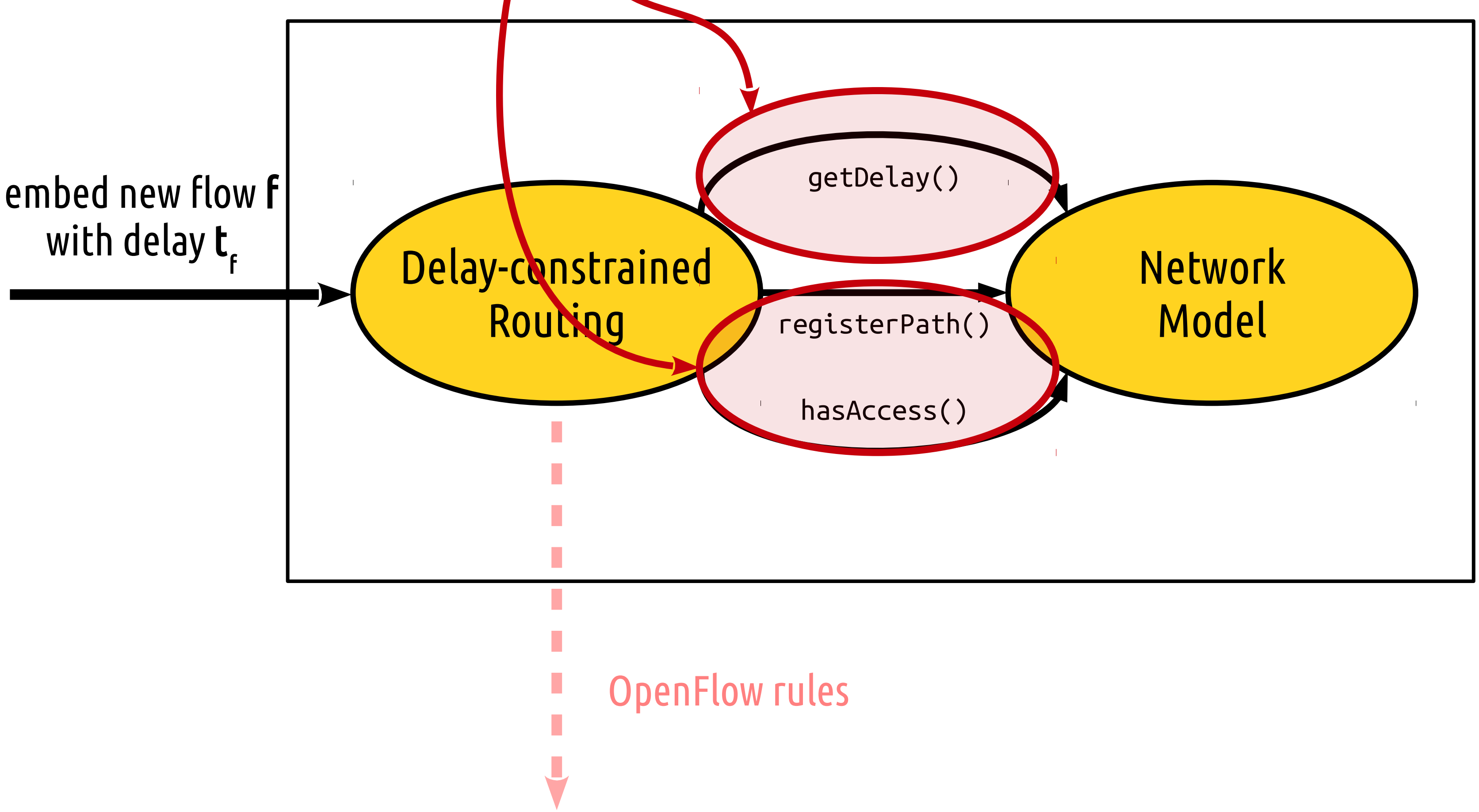
ii. the guarantees provided to previously embedded flows are **still valid**



Embed **f** such that

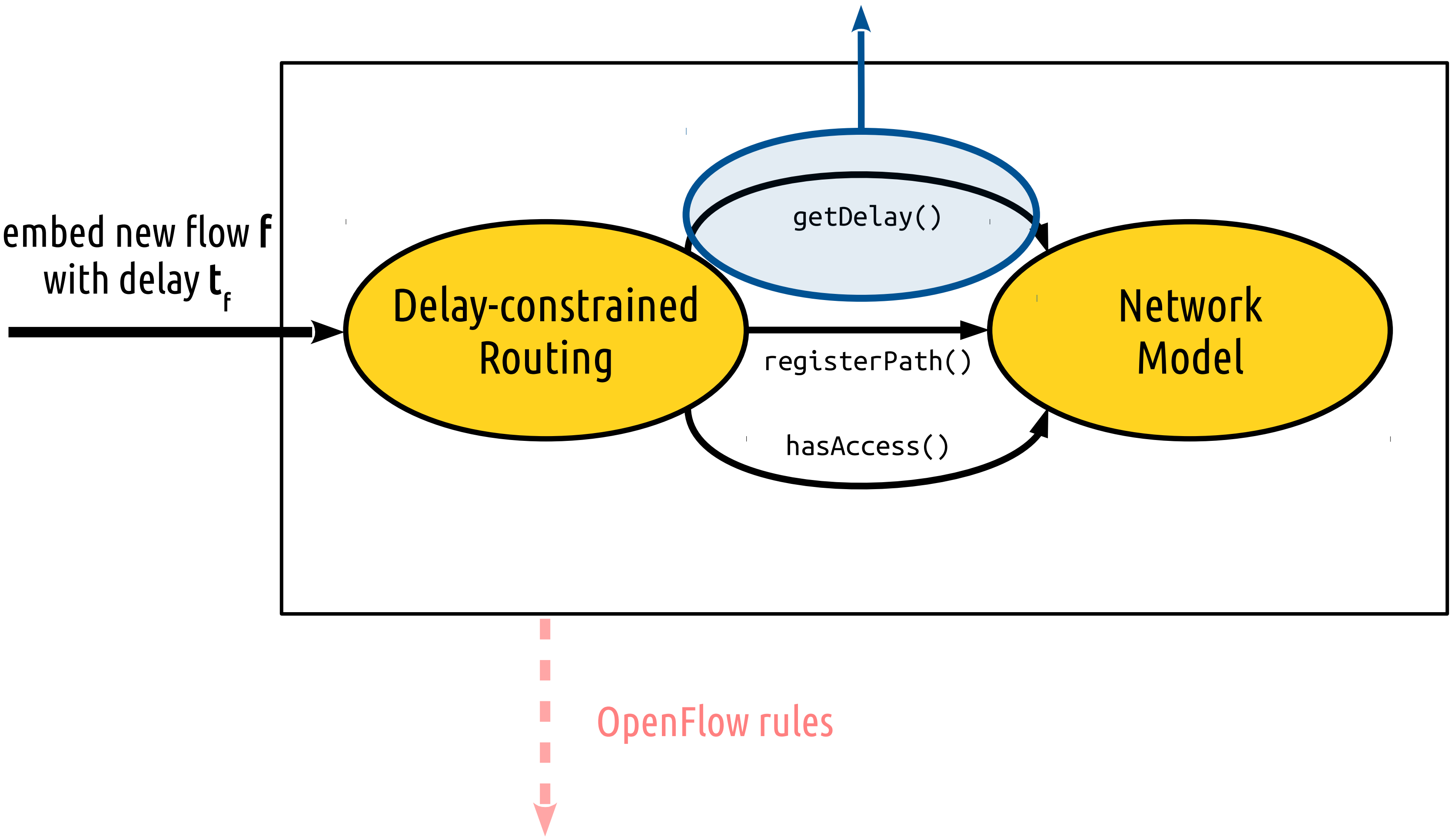
i. its **delay t_f is guaranteed**

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



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

assuming priority scheduling

(cheap and ubiquitous)

The delay of a route depends on

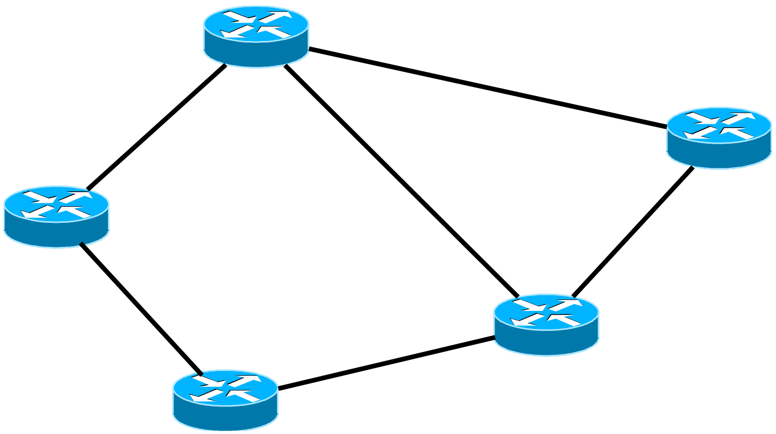
the physical **links**

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

assuming priority scheduling

(cheap and ubiquitous)

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



The delay of a route depends on

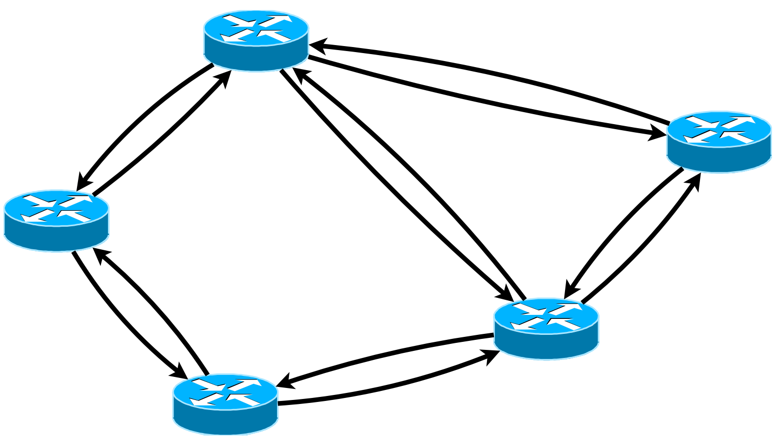
the physical **links**

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

assuming priority scheduling

(cheap and ubiquitous)

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



The delay of a route depends on

the physical **links**

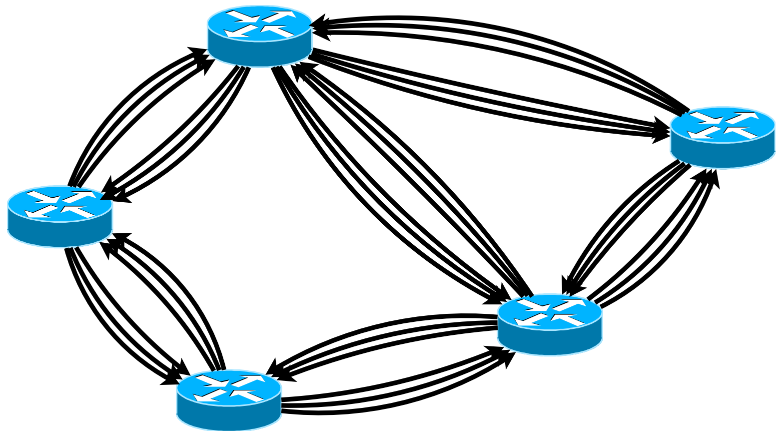
how the flow is **scheduled** at each node

the **queue** at which the flow is 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

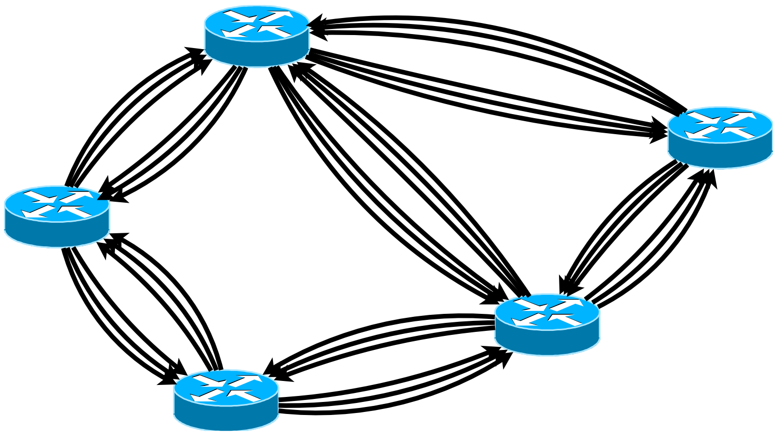
- # the physical **links**
- # how the flow is **scheduled** at each node
- # the **queue** at which the flow is schedule at each node

assuming priority scheduling
(cheap and ubiquitous)

QUEUE LINK TOPOLOGY

Performing route selection on this topology defines both

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



The delay of a route depends on

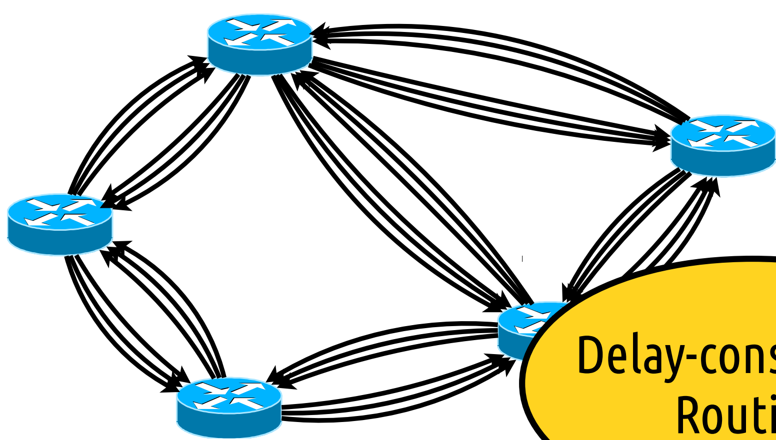
- **#** the physical **links**
- **#** how the flow is **scheduled** at each node
- **#** the **queue** at which the flow is schedule at each node

assuming priority scheduling
(cheap and ubiquitous)

QUEUE LINK TOPOLOGY

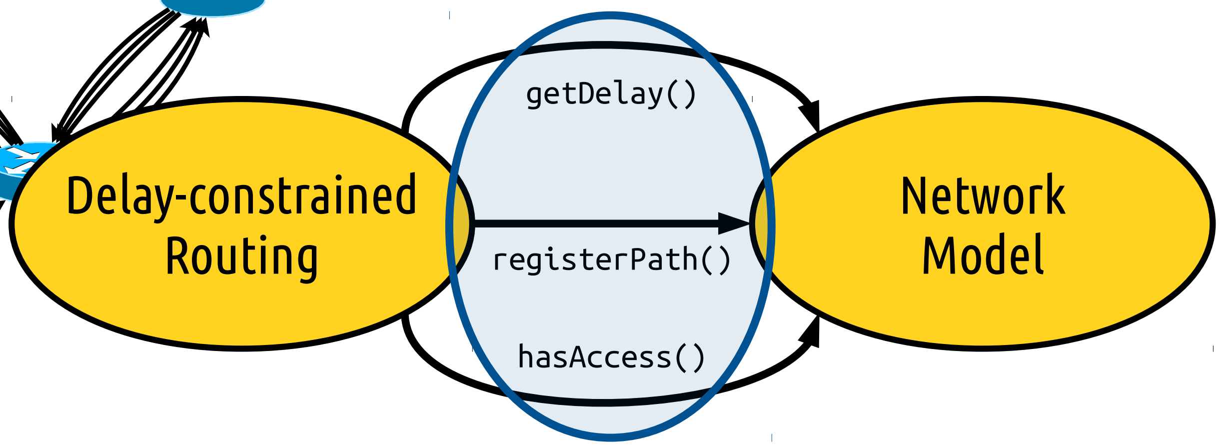
with, e.g., 3 priority queues at each output port

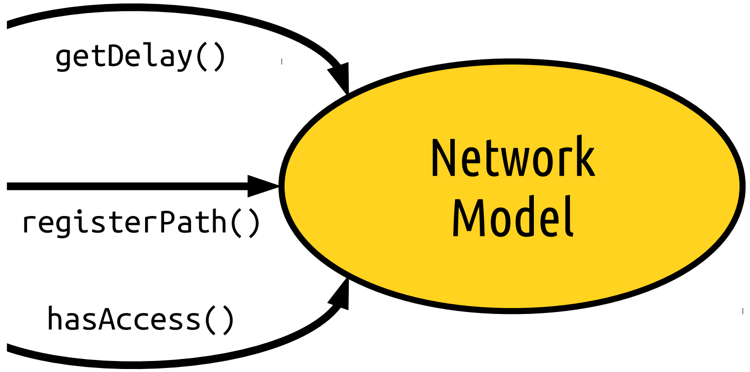
Performing route selection on this topology defines both



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)

$R_{u,v}$	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)
$\mathbf{U}_R[u,v,p]$	rate of flow through queue (u,v,p)
$\mathbf{U}_B[u,v,p]$	burst of flow through queue (u,v,p)
$l^{max}_{u,v,p}$	maximum packet size through queue (u,v,p)

We assume **token bucket** flows through each queue

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

$$\left(R_{u,v}t - t\sum_{j=1}^{p-1} \mathbf{U}_R[u,v,j] - \sum_{j=1}^{p-1} \mathbf{U}_B[u,v,j] - \max_{p+1 \le j \le Q_{u,v}} \{l^{max}_{u,v,j}\} - l^{max}_{u,v,p} \right)^+$$

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)
$\mathbf{U}_R[u, v, p]$	rate of flow through queue (u, v, p)
$\mathbf{U}_B[u, v, p]$	burst of flow through queue (u, v, p)
$l^{max}_{u,v,p}$	maximum packet size through queue (u, v, p)

We assume **token bucket** flows through each queue

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

$$\left(R_{u,v}t - t\sum_{j=1}^{p-1} \mathbf{U}_R[u,v,j] - \sum_{j=1}^{p-1} \mathbf{U}_B[u,v,j] - \max_{p+1 \le j \le Q_{u,v}} \{l^{max}_{u,v,j}\} - l^{max}_{u,v,p} \right)^+$$

whole link service

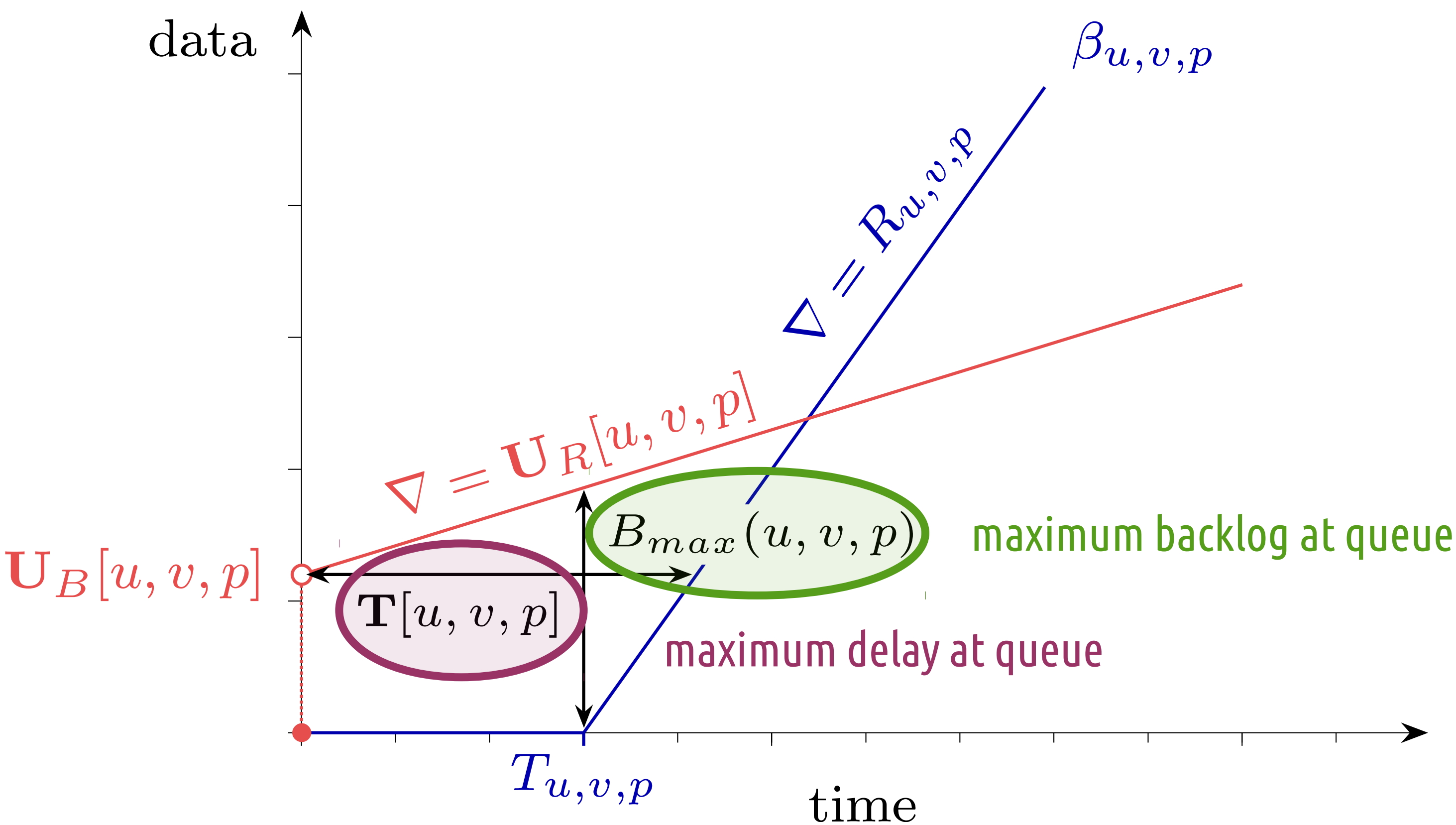
service used by higher priority queues

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

$$\beta_{u,v,p} = \left(R_{u,v}t - t\sum_{j=1}^{p-1} \mathbf{U}_R[u,v,j] - \sum_{j=1}^{p-1} \mathbf{U}_B[u,v,j] - \max_{p+1 \le j \le Q_{u,v}} \{l_{u,v,j}^{max}\} - l_{u,v,p}^{max} \right)^+$$



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,

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

and we must ensure that no buffer overflow occurs

$$B_{max}(u,v,p) \leq \mathbf{A}_B[u,v,p] \qquad \forall\ (u,v,p)$$

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,

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

and we must ensure that no buffer overflow occurs

$$B_{max}(u,v,p) \leq \mathbf{A}_B[u,v,p] \qquad \forall\ (u,v,p)$$

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,

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

and we must ensure that no buffer overflow occurs

$$B_{max}(u,v,p) \leq \mathbf{A}_B[u,v,p] \qquad \forall\ (u,v,p)$$

Buffer capacity of a queue

Must be fulfilled **at all times**, for all the flows

We have...

$$\mathbf{T}[u,v,p] = \frac{\sum_{j=1}^{p} \mathbf{U}_B[u,v,j] + \max_{p+1\leq j\leq Q_{u,v}} \{l_{u,v,j}^{max}\} + l_{u,v,p}^{max}}{R_{u,v} - \sum_{j=1}^{p-1} \mathbf{U}_R[u,v,j]}$$

$$B_{max}(u,v,p) = \mathbf{U}_B[u,v,p] + \mathbf{U}_R[u,v,p] \frac{\sum_{j=1}^{p-1} \mathbf{U}_B[u,v,j] + \max_{p+1\leq j\leq Q_{u,v}} \{l_{u,v,j}^{max}\} + l_{u,v,p}^{max}}{R_{u,v} - \sum_{j=1}^{p-1} \mathbf{U}_R[u,v,j]}$$

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,

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

and we must ensure that no buffer overflow occurs

$$B_{max}(u,v,p) \leq \mathbf{A}_B[u,v,p] \qquad \forall\ (u,v,p)$$

Buffer capacity of a queue

Must be fulfilled **at all times**, for all the flows

We have...

Dependence **on other flows** embedded at the same link

$$\mathbf{T}[u,v,p] = \frac{\sum_{j=1}^{p} \mathbf{U}_B[u,v,j] + \max_{p+1\leq j\leq Q_{u,v}} \{l_{u,v,j}^{max}\} + l_{u,v,p}^{max}}{R_{u,v} - \sum_{j=1}^{p-1} \mathbf{U}_R[u,v,j]}$$

$$B_{max}(u,v,p) = \mathbf{U}_B[u,v,p] + \mathbf{U}_R[u,v,p] \frac{\sum_{j=1}^{p-1} \mathbf{U}_B[u,v,j] + \max_{p+1\leq j\leq Q_{u,v}} \{l_{u,v,j}^{max}\} + l_{u,v,p}^{max}}{R_{u,v} - \sum_{j=1}^{p-1} \mathbf{U}_R[u,v,j]}$$

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

$$B_{max}(u,v,p) \leq \mathbf{A}_B[u,v,p] \qquad \forall\ (u,v,p)$$

Should we re-check all the previously embedded flows?

No, it does not scale!

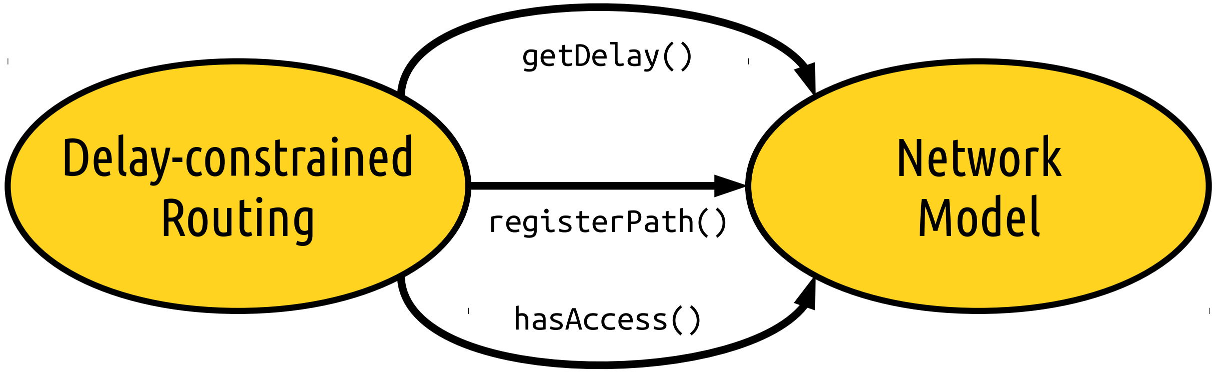
$$\sum_{(u,v,p)\in P_f} \mathbf{T}[u,v,p] \leq t_f \qquad \forall f$$

$$B_{max}(u,v,p) \leq \mathbf{A}_B[u,v,p] \qquad \forall\ (u,v,p)$$

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**!



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

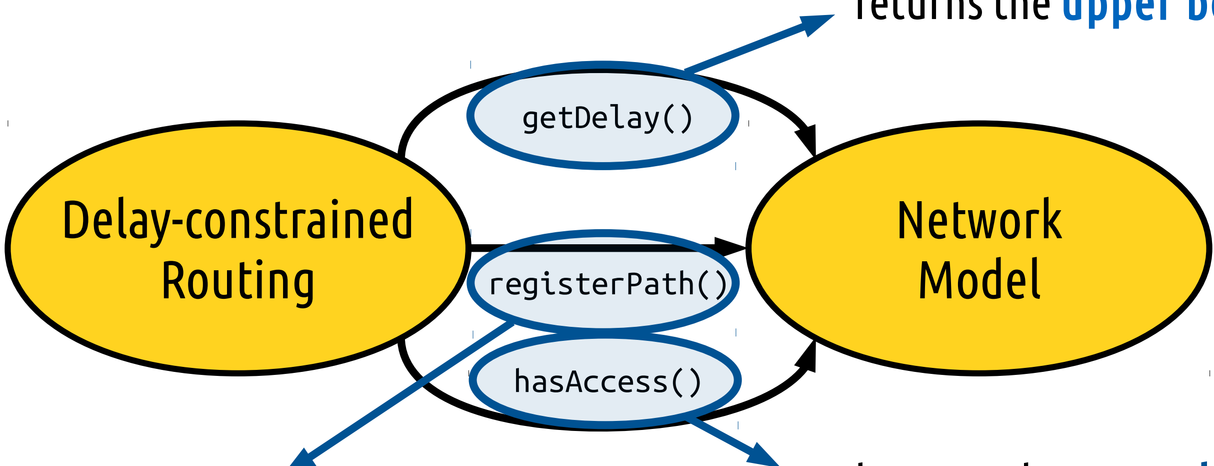
$$B_{max}(u,v,p) \leq \mathbf{A}_B[u,v,p] \qquad \forall\ (u,v,p)$$

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**!

returns the **upper bound**



adds the flow to current state

makes sure the **upper bound is not reached** if a flow is added to the current state

Let‘s find an upper bound independent of the network state…

$$\mathbf{T}[u,v,p] = \frac{\sum_{j=1}^{p} \mathbf{U}_B[u,v,j] + \max_{p+1 \le j \le Q_{u,v}} \{l_{u,v,j}^{max}\} + l_{u,v,p}^{max}}{R_{u,v} - \sum_{j=1}^{p-1} \mathbf{U}_R[u,v,j]}$$

$$B_{max}(u,v,p) = \mathbf{U}_B[u,v,p] + \mathbf{U}_R[u,v,p] \frac{\sum_{j=1}^{p-1} \mathbf{U}_B[u,v,j] + \max_{p+1 \le j \le Q_{u,v}} \{l_{u,v,j}^{max}\} + l_{u,v,p}^{max}}{R_{u,v} - \sum_{j=1}^{p-1} \mathbf{U}_R[u,v,j]}$$

Let‘s find an upper bound independent of the network state…

Packets cannot be bigger than the biggest Ethernet frame size

$$\mathbf{T}[u,v,p] = \frac{\sum_{j=1}^{p} \mathbf{U}_B[u,v,j] + \quad 1542 + 1542}{R_{u,v} - \sum_{j=1}^{p-1} \mathbf{U}_R[u,v,j]}$$

$$B_{max}(u,v,p) = \mathbf{U}_B[u,v,p] + \mathbf{U}_R[u,v,p] \frac{\sum_{j=1}^{p-1} \mathbf{U}_B[u,v,j] + \quad 1542 + 1542}{R_{u,v} - \sum_{j=1}^{p-1} \mathbf{U}_R[u,v,j]}$$

Let's find an upper bound independent of the network state...

$\mathbf{A}_R[u,v,p]$ defined as the maximum rate that can be accepted at a queue

$$\mathbf{T}[u,v,p] = \frac{\sum_{j=1}^{p} \mathbf{U}_B[u,v,j] + \qquad 1542 + 1542}{R_{u,v} - \sum_{j=1}^{p-1} \mathbf{A}_R[u,v,p]}$$

$$B_{max}(u,v,p) = \mathbf{U}_B[u,v,p] + \mathbf{A}_R[u,v,p] \frac{\sum_{j=1}^{p-1} \mathbf{U}_B[u,v,j] + \qquad 1542 + 1542}{R_{u,v} - \sum_{j=1}^{p-1} \mathbf{A}_R[u,v,p]}$$

Let's find an upper bound independent of the network state...

$$\mathbf{T}[u,v,p] = \frac{\sum_{j=1}^{p} \mathbf{U}_B[u,v,j] + 1542 + 1542}{R_{u,v} - \sum_{j=1}^{p-1} \mathbf{A}_R[u,v,p]}$$

$$B_{max}(u,v,p) = \mathbf{U}_B[u,v,p] + \mathbf{A}_R[u,v,p] \frac{\sum_{j=1}^{p-1} \mathbf{U}_B[u,v,j] + 1542 + 1542}{R_{u,v} - \sum_{j=1}^{p-1} \mathbf{A}_R[u,v,p]}$$

Let‘s find an upper bound independent of the network state…

Limit bursts such that no buffer overflow occurs

$$\mathbf{T}[u,v,p] = \frac{\sum_{j=1}^{p} \mathbf{U}_B[u,v,j] + \qquad 1542 + 1542}{R_{u,v} - \sum_{j=1}^{p-1} \mathbf{A}_R[u,v,p]}$$

$$B_{max}(u,v,p) = \mathbf{U}_B[u,v,p] + \mathbf{A}_R[u,v,p] \frac{\sum_{j=1}^{p-1} \mathbf{U}_B[u,v,j] + \qquad 1542 + 1542}{R_{u,v} - \sum_{j=1}^{p-1} \mathbf{A}_R[u,v,p]}$$

$$\leq \mathbf{A}_B[u,v,p]$$

Let‘s find an upper bound independent of the network state…

Limit bursts such that no buffer overflow occurs

$$\mathbf{T}[u,v,p] = \frac{\sum_{j=1}^{p} \mathbf{M}_B[u,v,p] + \qquad 1542 + 1542}{R_{u,v} - \sum_{j=1}^{p-1} \mathbf{A}_R[u,v,p]}$$

$$B_{max}(u,v,p) = \mathbf{M}_B[u,v,p] + \mathbf{A}_R[u,v,p] \frac{\sum_{j=1}^{p-1} \mathbf{M}_B[u,v,p] + \qquad 1542 + 1542}{R_{u,v} - \sum_{j=1}^{p-1} \mathbf{A}_R[u,v,p]}$$

$$\leq \mathbf{A}_B[u,v,p]$$

Let‘s find an upper bound independent of the network state…

Limit bursts such that no buffer overflow occurs

$$\mathbf{T}[u,v,p] = \frac{\sum_{j=1}^{p} \mathbf{M}_B[u,v,p] + \quad 1542 + 1542}{R_{u,v} - \sum_{j=1}^{p-1} \mathbf{A}_R[u,v,p]}$$

$$B_{max}(u,v,p) = \mathbf{M}_B[u,v,p] + \mathbf{A}_R[u,v,p] \frac{\sum_{j=1}^{p-1} \mathbf{M}_B[u,v,p] + \quad 1542 + 1542}{R_{u,v} - \sum_{j=1}^{p-1} \mathbf{A}_R[u,v,p]}$$

$$\leq \mathbf{A}_B[u,v,p]$$

The maximum bursts $\mathbf{M}_B[u,v,p]$ can be computed recursively

$$\mathbf{M}_B[u,v,1] + \mathbf{A}_R[u,v,1] \frac{3084}{R_{u,v}} = \mathbf{A}_B[u,v,1]$$

Let's find an upper bound independent of the network state…

Limit bursts such that no buffer overflow occurs

$$\mathbf{T}[u,v,p] = \frac{\sum_{j=1}^{p} \mathbf{M}_B[u,v,p] + 1542 + 1542}{R_{u,v} - \sum_{j=1}^{p-1} \mathbf{A}_R[u,v,p]}$$

$$B_{max}(u,v,p) = \mathbf{M}_B[u,v,p] + \mathbf{A}_R[u,v,p] \frac{\sum_{j=1}^{p-1} \mathbf{M}_B[u,v,p] + 1542 + 1542}{R_{u,v} - \sum_{j=1}^{p-1} \mathbf{A}_R[u,v,p]}$$

$$\leq \mathbf{A}_B[u,v,p]$$

The maximum bursts $\mathbf{M}_B[u,v,p]$ can be computed recursively

$$\mathbf{M}_B[u,v,2] + \mathbf{A}_R[u,v,2] \frac{\mathbf{M}_B[u,v,1] + 3084}{R_{u,v} - \mathbf{A}_R[u,v,1]} = \mathbf{A}_B[u,v,1]$$

etc.

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

$$\mathbf{T}[u,v,p] \leq \frac{\sum_{j=1}^{p} \mathbf{M}_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1} \mathbf{A}_R[u,v,j]} \triangleq \mathbf{T}^{MHM}[u,v,p]$$

computed recursively

defined per queue by a resource allocation algorithm

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

$$\mathbf{T}[u,v,p] \;\leq\; \frac{\sum_{j=1}^{p} \mathbf{M}_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1} \mathbf{A}_R[u,v,j]} \triangleq \mathbf{T}^{MHM}[u,v,p]$$

computed recursively

defined per queue by a resource allocation algorithm

The opposite can actually also be done...

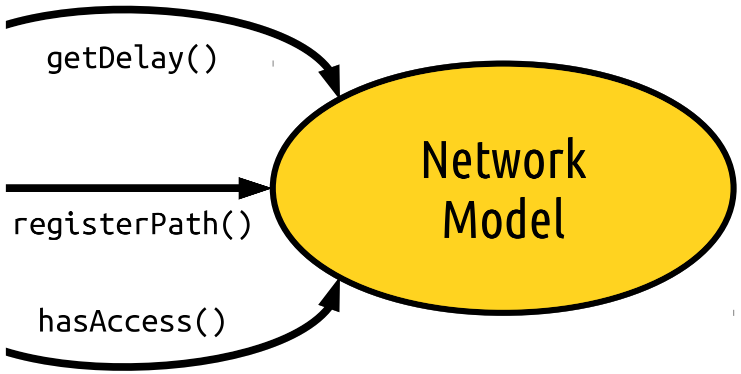
$$\mathbf{T}[u,v,p] \;\leq\; \frac{\sum_{j=1}^{p} \mathbf{M}_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1} \mathbf{A}_R[u,v,j]} \triangleq \mathbf{T}^{MHM}[u,v,p]$$

The Multi-Hop Model (MHM)

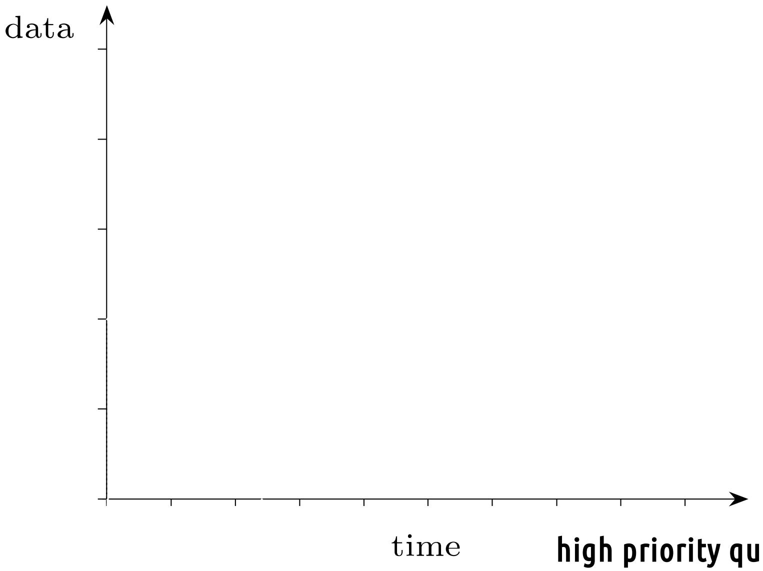
return $\mathbf{T}^{MHM}[u,v,p]$

update $\mathbf{U}_R[u,v,p], \mathbf{U}_B[u,v,p]$

check $\mathbf{M}_B[u,v,p], \mathbf{A}_R[u,v,p]$ are not exceeded



Let‘s see how this looks like graphically... at a link with 3 priority queues



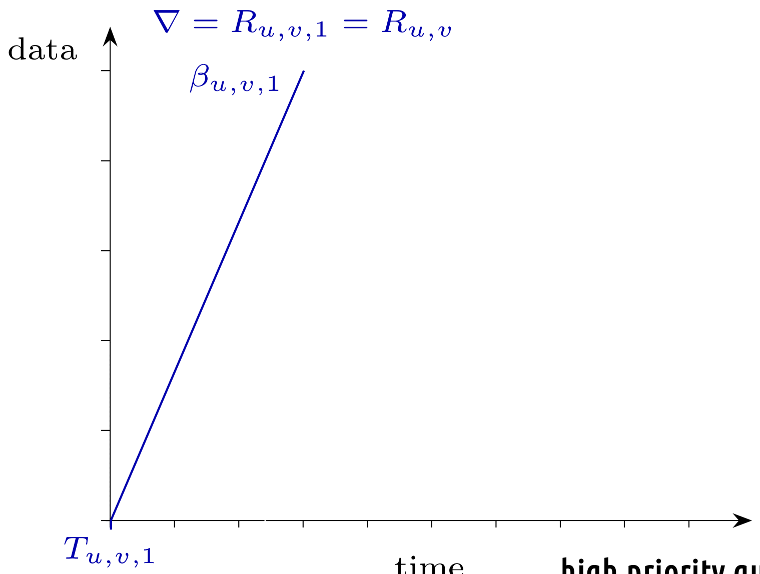
high priority queue

$$\mathbf{M}_B[u,v,p] + \mathbf{A}_R[u,v,p]\frac{\sum_{j=1}^{p-1}\mathbf{M}_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j]} = \mathbf{A}_B[u,v,p]$$

$$\beta_{u,v,p} = \left(R_{u,v}t - t\sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j] - \sum_{j=1}^{p-1}\mathbf{M}_B[u,v,j] - 3084\right)^{+}$$

$$\mathbf{T}^{MHM}[u,v,p] = \frac{\sum_{j=1}^{p}\mathbf{M}_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j]}$$

Let‘s see how this looks like graphically… at a link with 3 priority queues



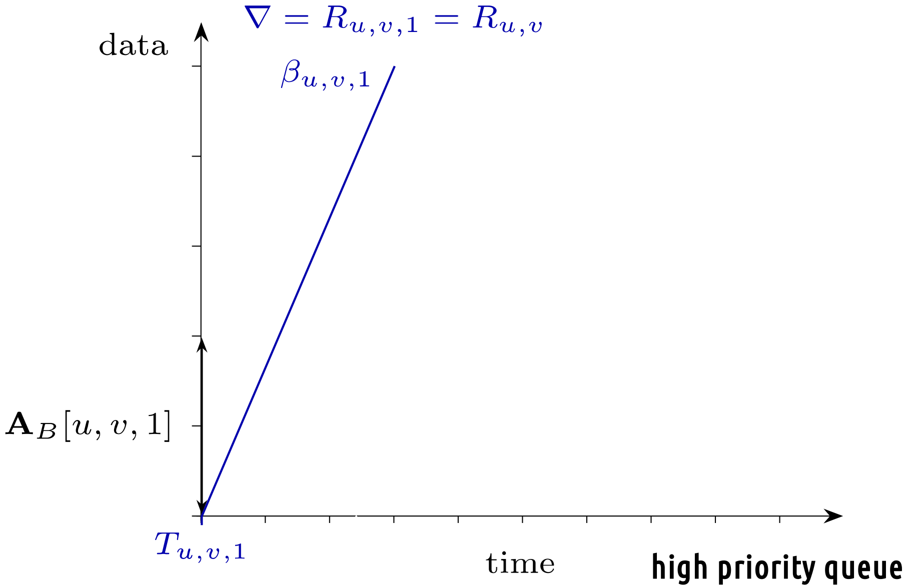
high priority queue

$$\mathbf{M}_B[u,v,p] + \mathbf{A}_R[u,v,p]\frac{\sum_{j=1}^{p-1}\mathbf{M}_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j]} = \mathbf{A}_B[u,v,p]$$

$$\beta_{u,v,p} = \left(R_{u,v}t - t\sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j] - \sum_{j=1}^{p-1}\mathbf{M}_B[u,v,j] - 3084\right)^+$$

$$\mathbf{T}^{MHM}[u,v,p] = \frac{\sum_{j=1}^{p}\mathbf{M}_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j]}$$

Let‘s see how this looks like graphically… at a link with 3 priority queues

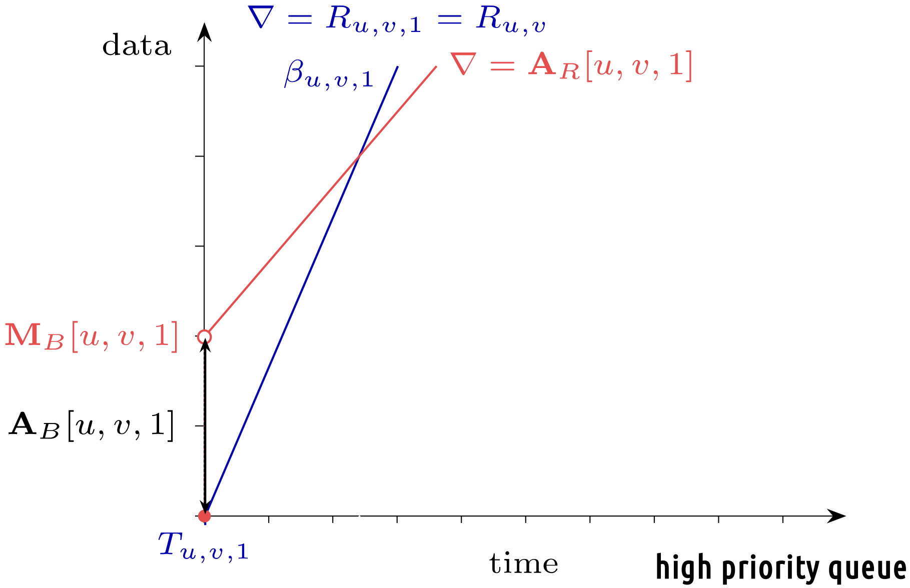


$$\mathbf{M}_B[u,v,p] + \mathbf{A}_R[u,v,p]\frac{\sum_{j=1}^{p-1}\mathbf{M}_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j]} = \mathbf{A}_B[u,v,p]$$

$$\beta_{u,v,p} = \left(R_{u,v}t - t\sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j] - \sum_{j=1}^{p-1}\mathbf{M}_B[u,v,j] - 3084\right)^+$$

$$\mathbf{T}^{MHM}[u,v,p] = \frac{\sum_{j=1}^{p}\mathbf{M}_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j]}$$

Let‘s see how this looks like graphically… at a link with 3 priority queues

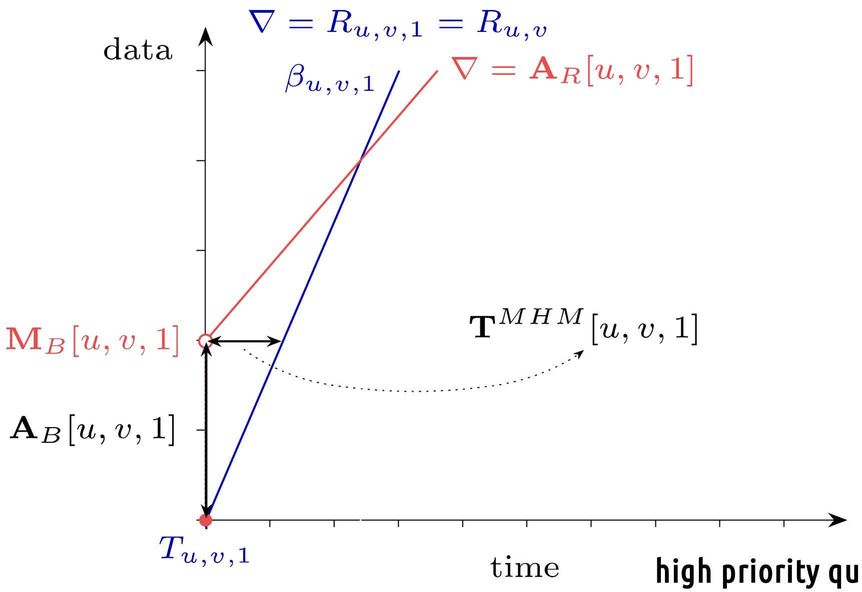


$$\mathbf{M}_B[u,v,p] + \mathbf{A}_R[u,v,p]\frac{\sum_{j=1}^{p-1}\mathbf{M}_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j]} = \mathbf{A}_B[u,v,p]$$

$$\beta_{u,v,p} = \left(R_{u,v}t - t\sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j] - \sum_{j=1}^{p-1}\mathbf{M}_B[u,v,j] - 3084\right)^+$$

$$\mathbf{T}^{MHM}[u,v,p] = \frac{\sum_{j=1}^{p}\mathbf{M}_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j]}$$

Let's see how this looks like graphically... at a link with 3 priority queues



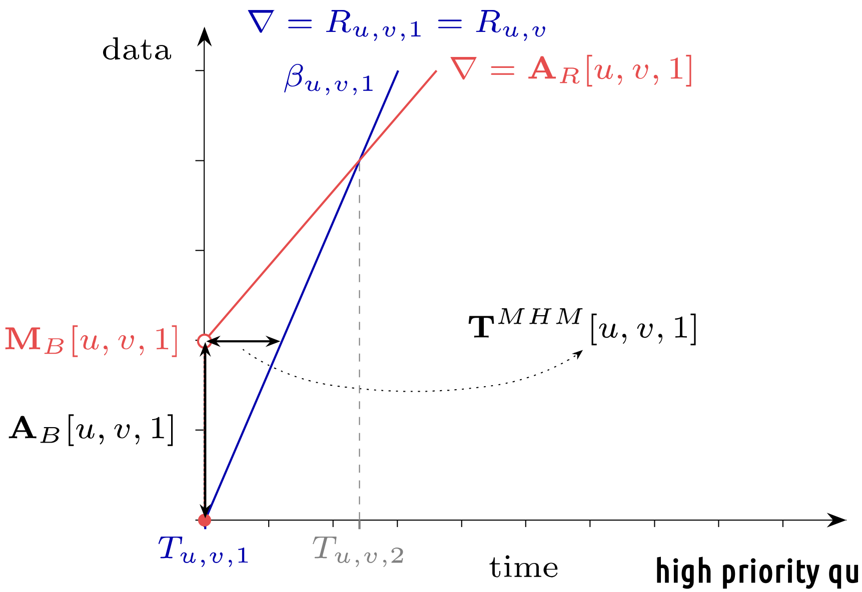
high priority queue

$$\mathbf{M}_B[u,v,p] + \mathbf{A}_R[u,v,p]\frac{\sum_{j=1}^{p-1}\mathbf{M}_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j]} = \mathbf{A}_B[u,v,p]$$

$$\beta_{u,v,p} = \left(R_{u,v}t - t\sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j] - \sum_{j=1}^{p-1}\mathbf{M}_B[u,v,j] - 3084\right)^+$$

$$\mathbf{T}^{MHM}[u,v,p] = \frac{\sum_{j=1}^{p}\mathbf{M}_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j]}$$

Let‘s see how this looks like graphically… at a link with 3 priority queues



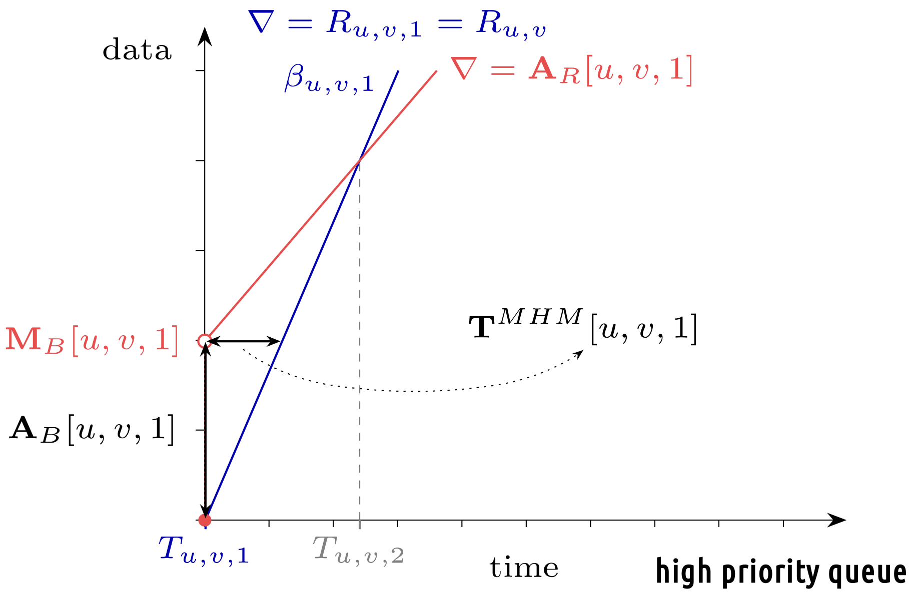
high priority queue

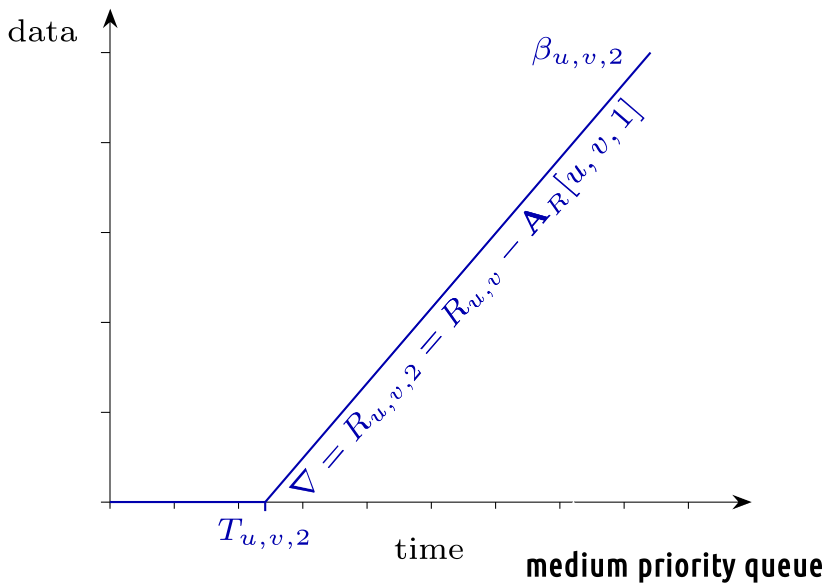
$$\mathbf{M}_B[u,v,p] + \mathbf{A}_R[u,v,p]\frac{\sum_{j=1}^{p-1}\mathbf{M}_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j]} = \mathbf{A}_B[u,v,p]$$

$$\beta_{u,v,p} = \left(R_{u,v}t - t\sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j] - \sum_{j=1}^{p-1}\mathbf{M}_B[u,v,j] - 3084\right)^+$$

$$\mathbf{T}^{MHM}[u,v,p] = \frac{\sum_{j=1}^{p}\mathbf{M}_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j]}$$

Let's see how this looks like graphically... at a link with 3 priority queues



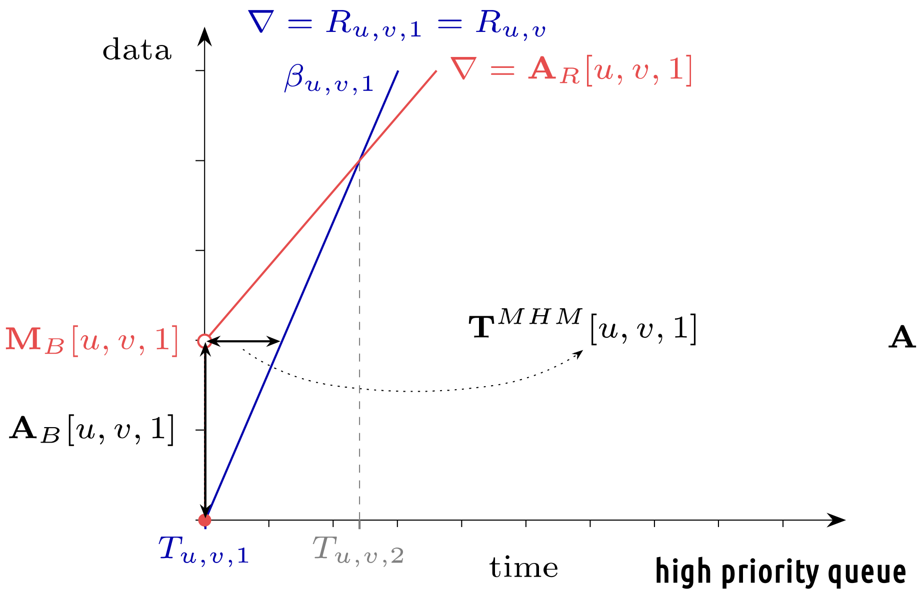


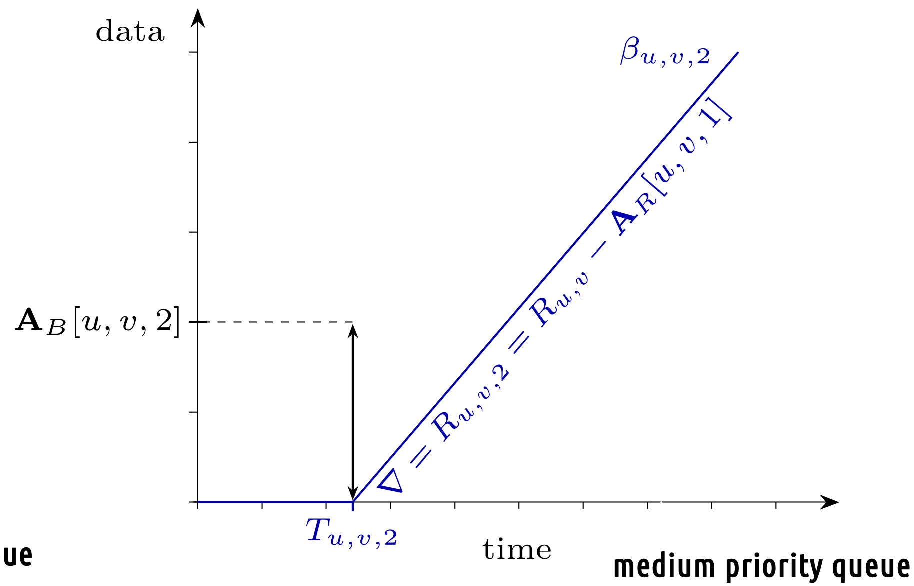
$$\mathbf{M}_B[u,v,p] + \mathbf{A}_R[u,v,p]\frac{\sum_{j=1}^{p-1}\mathbf{M}_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j]} = \mathbf{A}_B[u,v,p]$$

$$\beta_{u,v,p} = \left(R_{u,v}t - t\sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j] - \sum_{j=1}^{p-1}\mathbf{M}_B[u,v,j] - 3084\right)^+$$

$$\mathbf{T}^{MHM}[u,v,p] = \frac{\sum_{j=1}^{p}\mathbf{M}_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j]}$$

Let‘s see how this looks like graphically… at a link with 3 priority queues



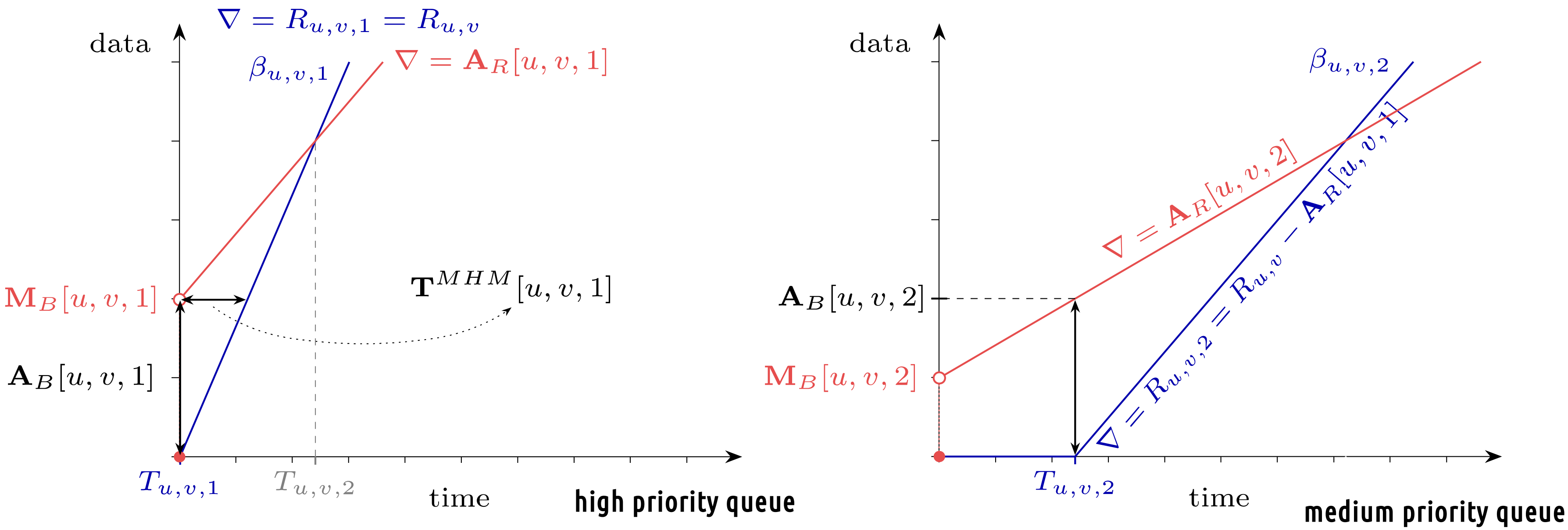


$$\mathbf{M}_B[u,v,p] + \mathbf{A}_R[u,v,p]\frac{\sum_{j=1}^{p-1}\mathbf{M}_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j]} = \mathbf{A}_B[u,v,p]$$

$$\beta_{u,v,p} = \left(R_{u,v}t - t\sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j] - \sum_{j=1}^{p-1}\mathbf{M}_B[u,v,j] - 3084\right)^+$$

$$\mathbf{T}^{MHM}[u,v,p] = \frac{\sum_{j=1}^{p}\mathbf{M}_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j]}$$

Let‘s see how this looks like graphically… at a link with 3 priority queues

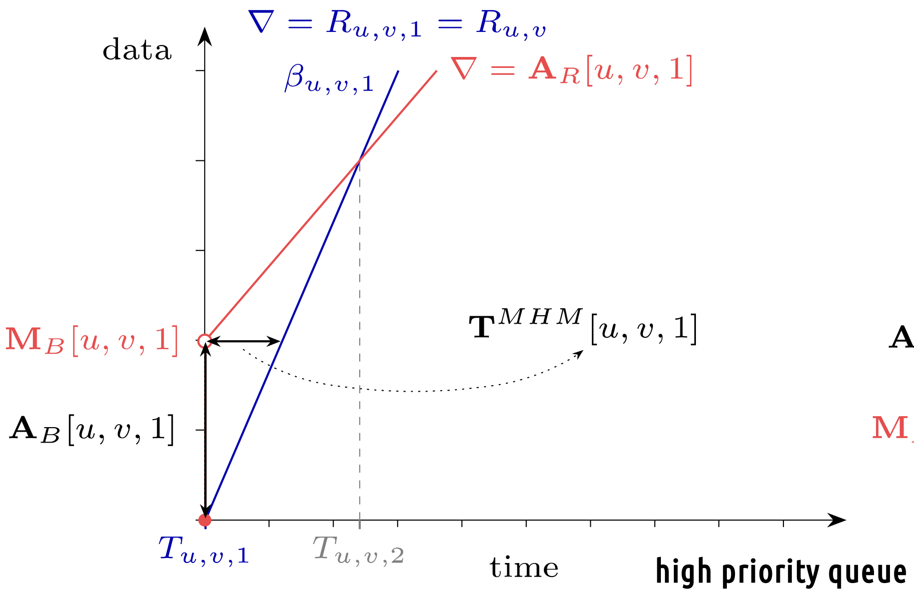


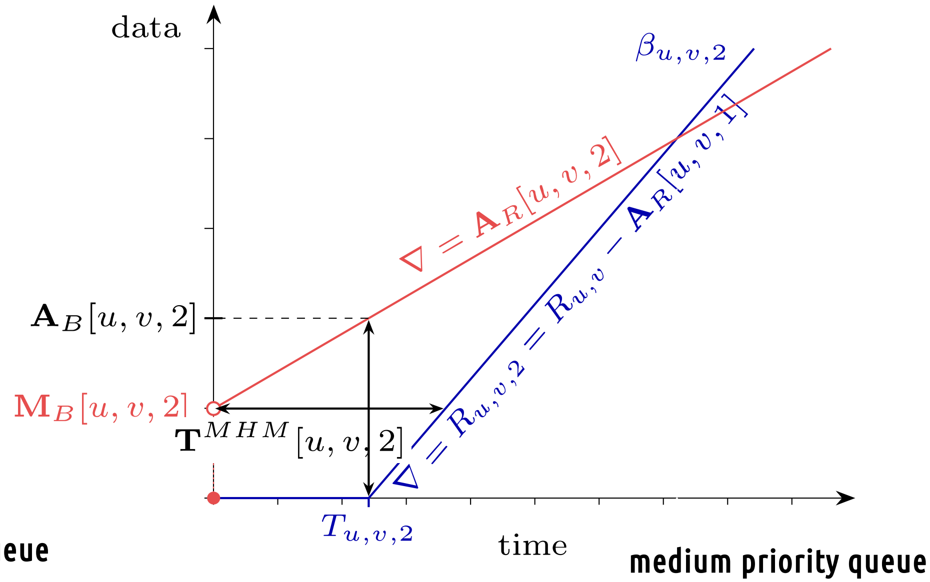
$$\mathbf{M}_B[u,v,p] + \mathbf{A}_R[u,v,p]\frac{\sum_{j=1}^{p-1}\mathbf{M}_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j]} = \mathbf{A}_B[u,v,p]$$

$$\beta_{u,v,p} = \left(R_{u,v}t - t\sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j] - \sum_{j=1}^{p-1}\mathbf{M}_B[u,v,j] - 3084\right)^+$$

$$\mathbf{T}^{MHM}[u,v,p] = \frac{\sum_{j=1}^{p}\mathbf{M}_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j]}$$

Let's see how this looks like graphically… at a link with 3 priority queues



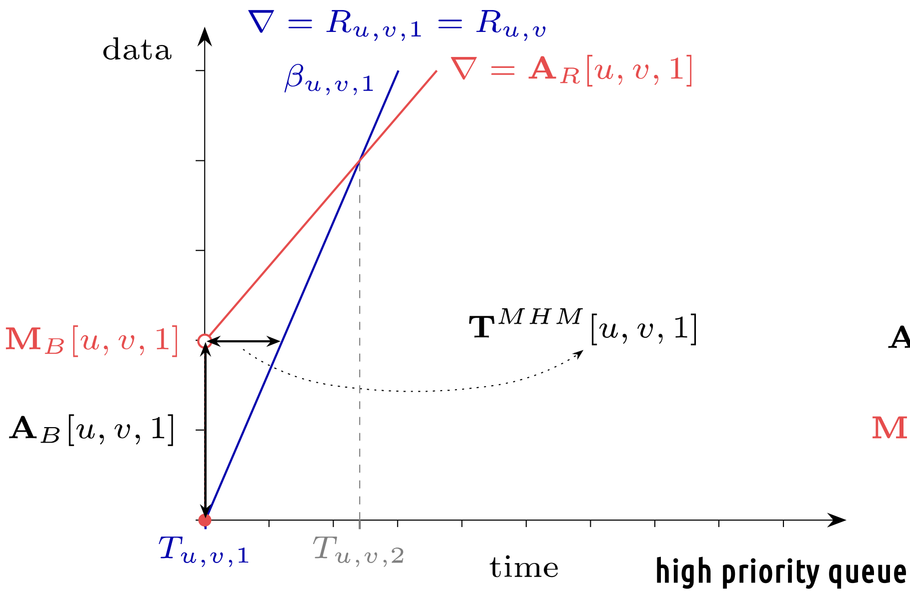


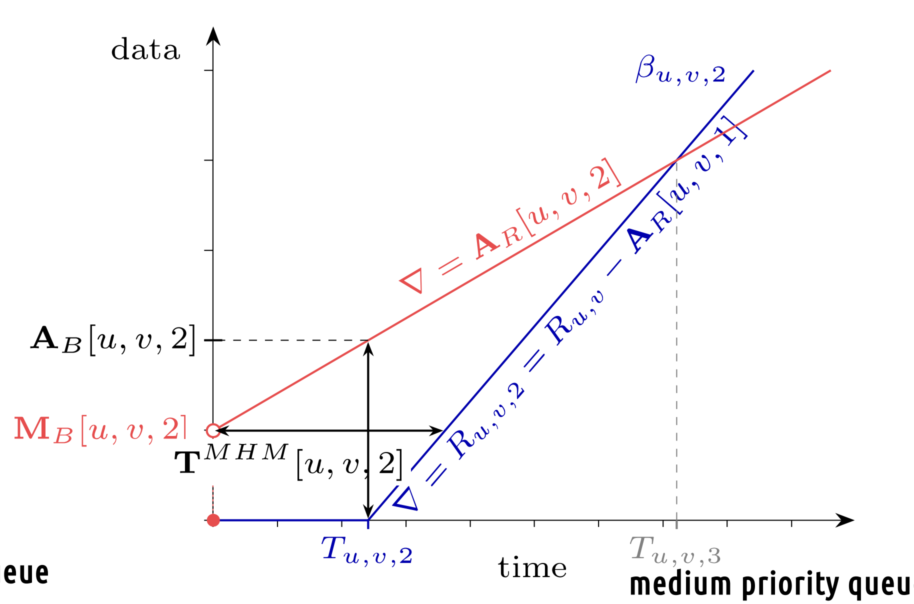
$$\mathbf{M}_B[u,v,p] + \mathbf{A}_R[u,v,p]\frac{\sum_{j=1}^{p-1}\mathbf{M}_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j]} = \mathbf{A}_B[u,v,p]$$

$$\beta_{u,v,p} = \left(R_{u,v}t - t\sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j] - \sum_{j=1}^{p-1}\mathbf{M}_B[u,v,j] - 3084\right)^+$$

$$\mathbf{T}^{MHM}[u,v,p] = \frac{\sum_{j=1}^{p}\mathbf{M}_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j]}$$

Let‘s see how this looks like graphically... at a link with 3 priority queues



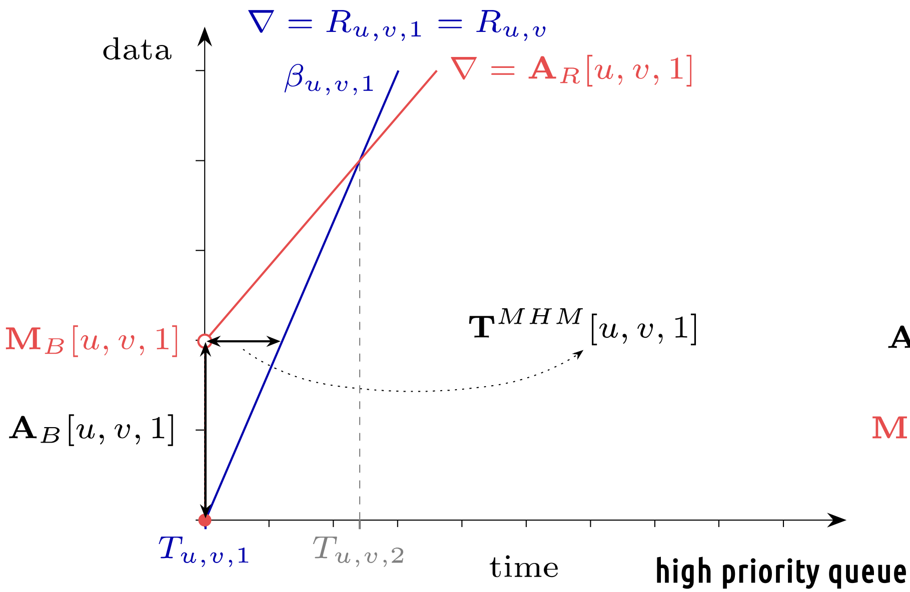


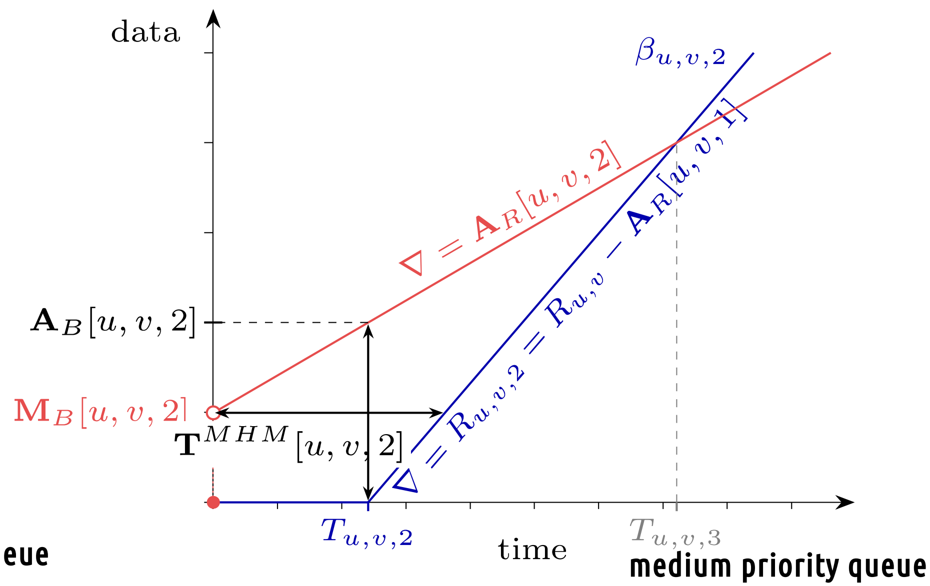
$$\mathbf{M}_B[u,v,p] + \mathbf{A}_R[u,v,p]\frac{\sum_{j=1}^{p-1}\mathbf{M}_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j]} = \mathbf{A}_B[u,v,p]$$

$$\beta_{u,v,p} = \left(R_{u,v}t - t\sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j] - \sum_{j=1}^{p-1}\mathbf{M}_B[u,v,j] - 3084\right)^+$$

$$\mathbf{T}^{MHM}[u,v,p] = \frac{\sum_{j=1}^{p}\mathbf{M}_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j]}$$

Let's see how this looks like graphically... at a link with 3 priority queues

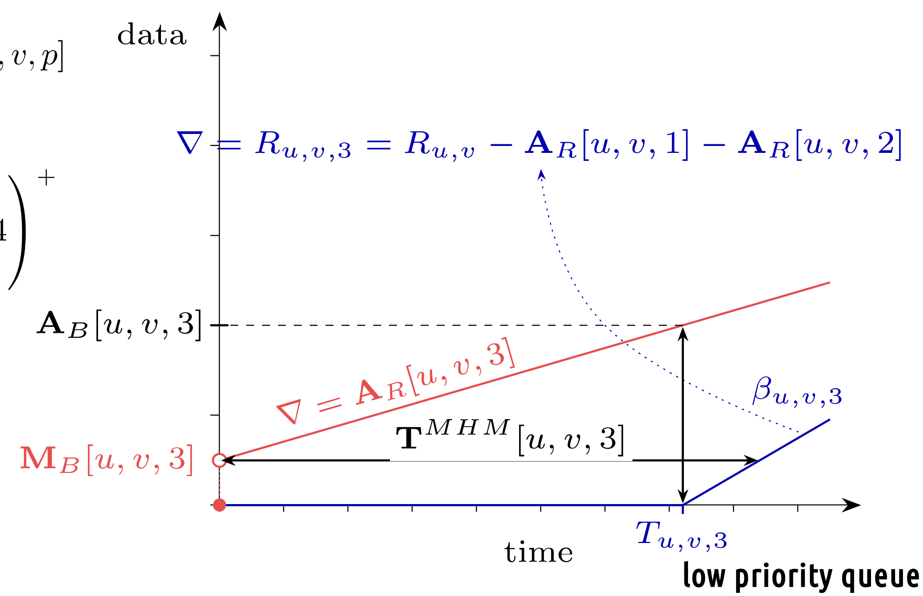




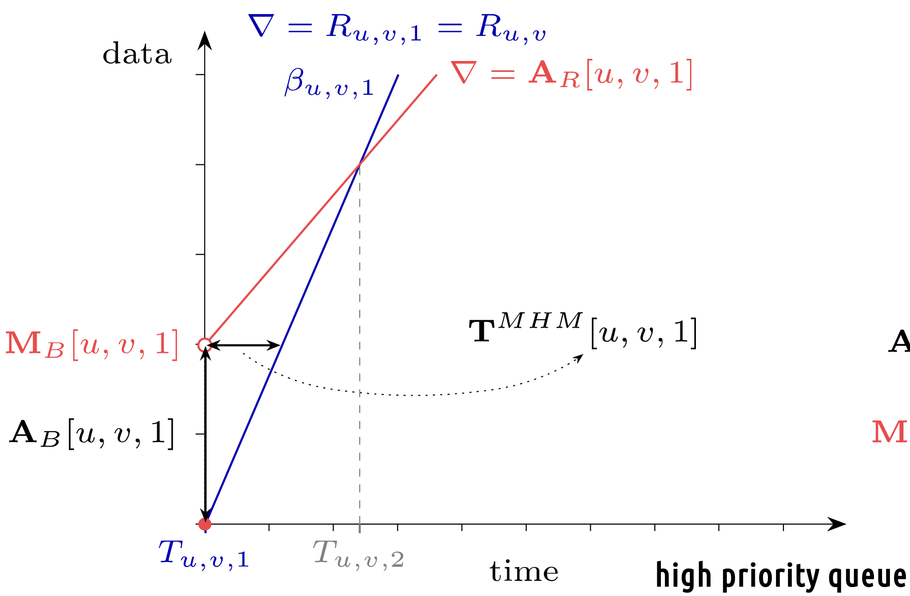
$$\mathbf{M}_B[u,v,p] + \mathbf{A}_R[u,v,p]\frac{\sum_{j=1}^{p-1}\mathbf{M}_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j]} = \mathbf{A}_B[u,v,p]$$

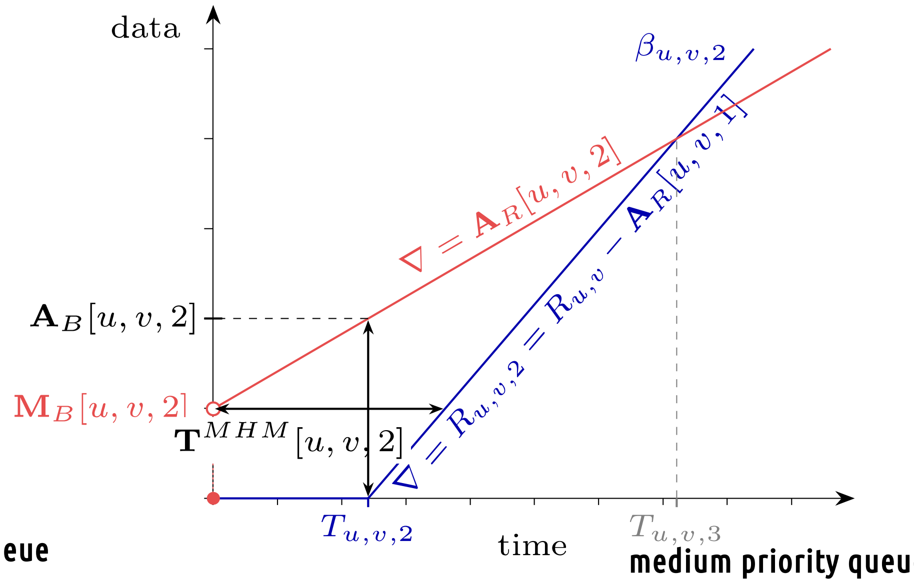
$$\beta_{u,v,p} = \left(R_{u,v}t - t\sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j] - \sum_{j=1}^{p-1}\mathbf{M}_B[u,v,j] - 3084\right)^+$$

$$\mathbf{T}^{MHM}[u,v,p] = \frac{\sum_{j=1}^{p}\mathbf{M}_B[u,v,j] + 3084}{R_{u,v} - \sum_{j=1}^{p-1}\mathbf{A}_R[u,v,j]}$$

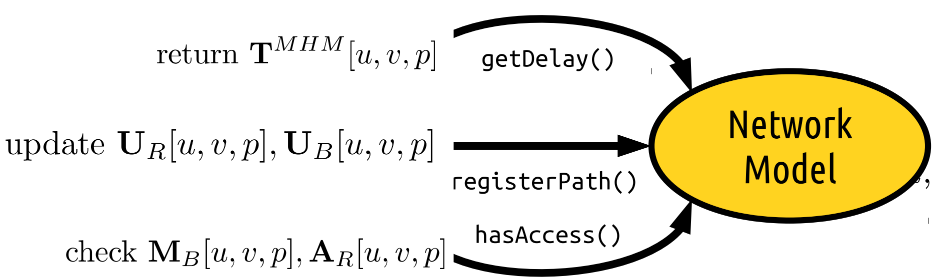


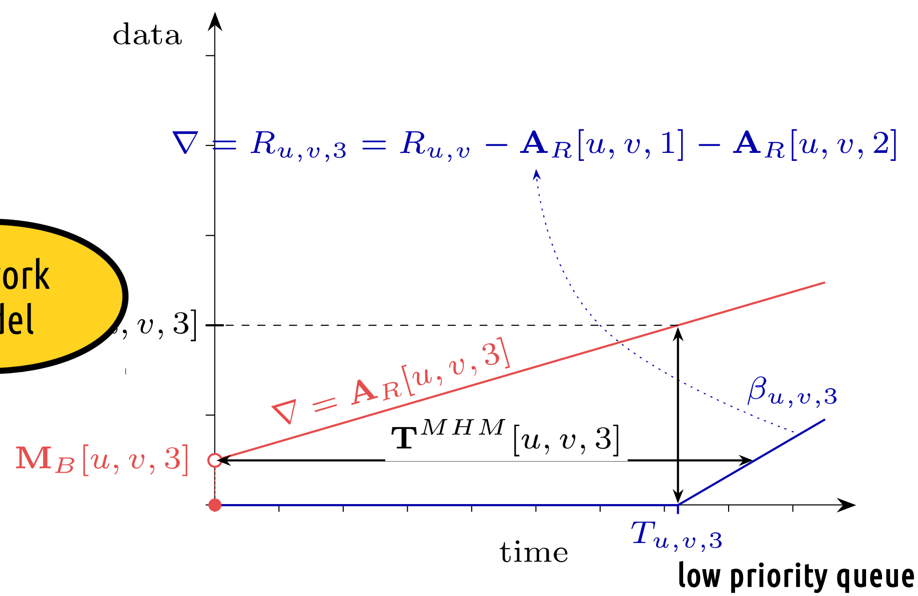
Let's see how this looks like graphically... at a link with 3 priority queues

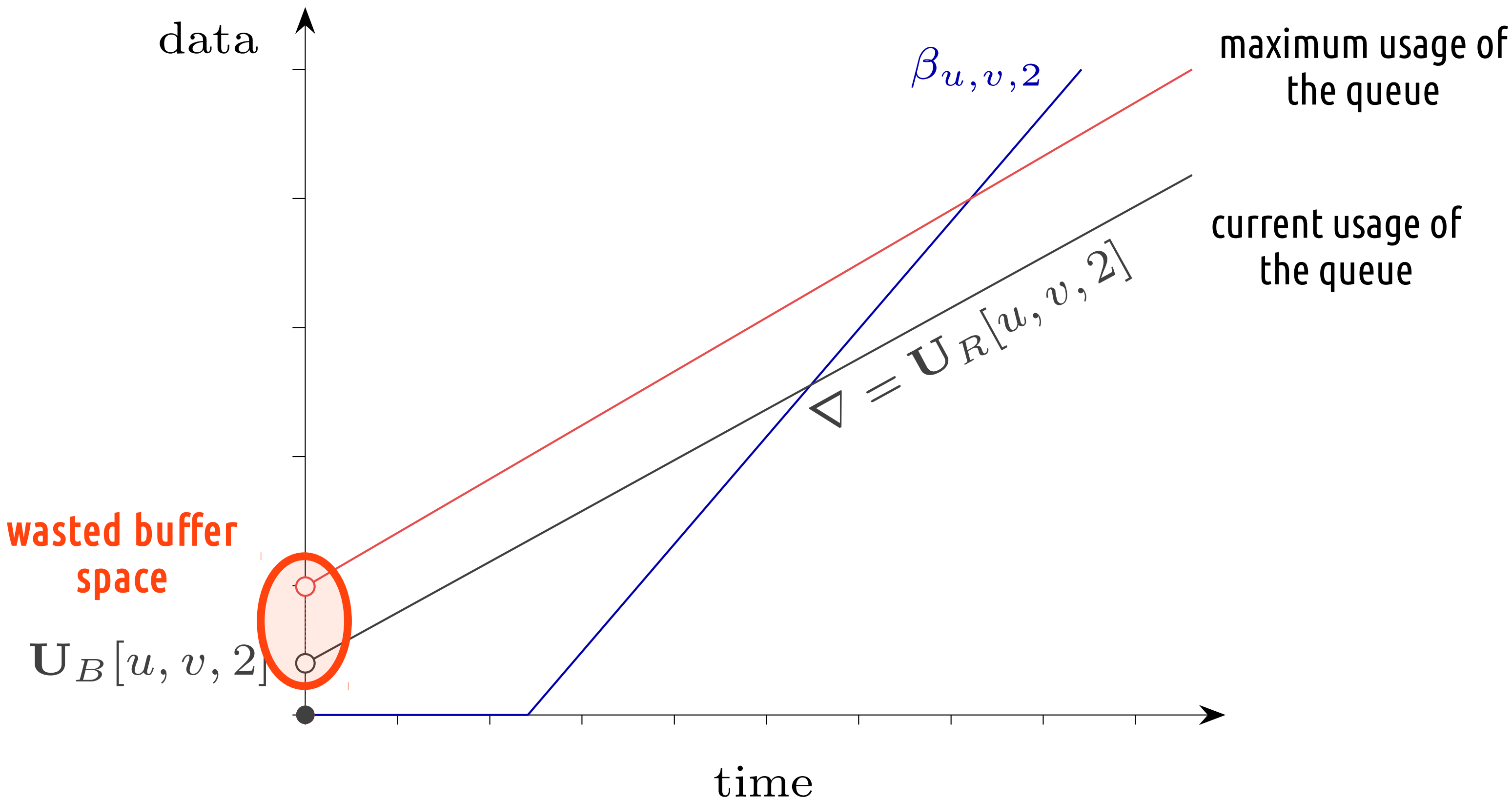




The Multi-Hop Model (MHM)





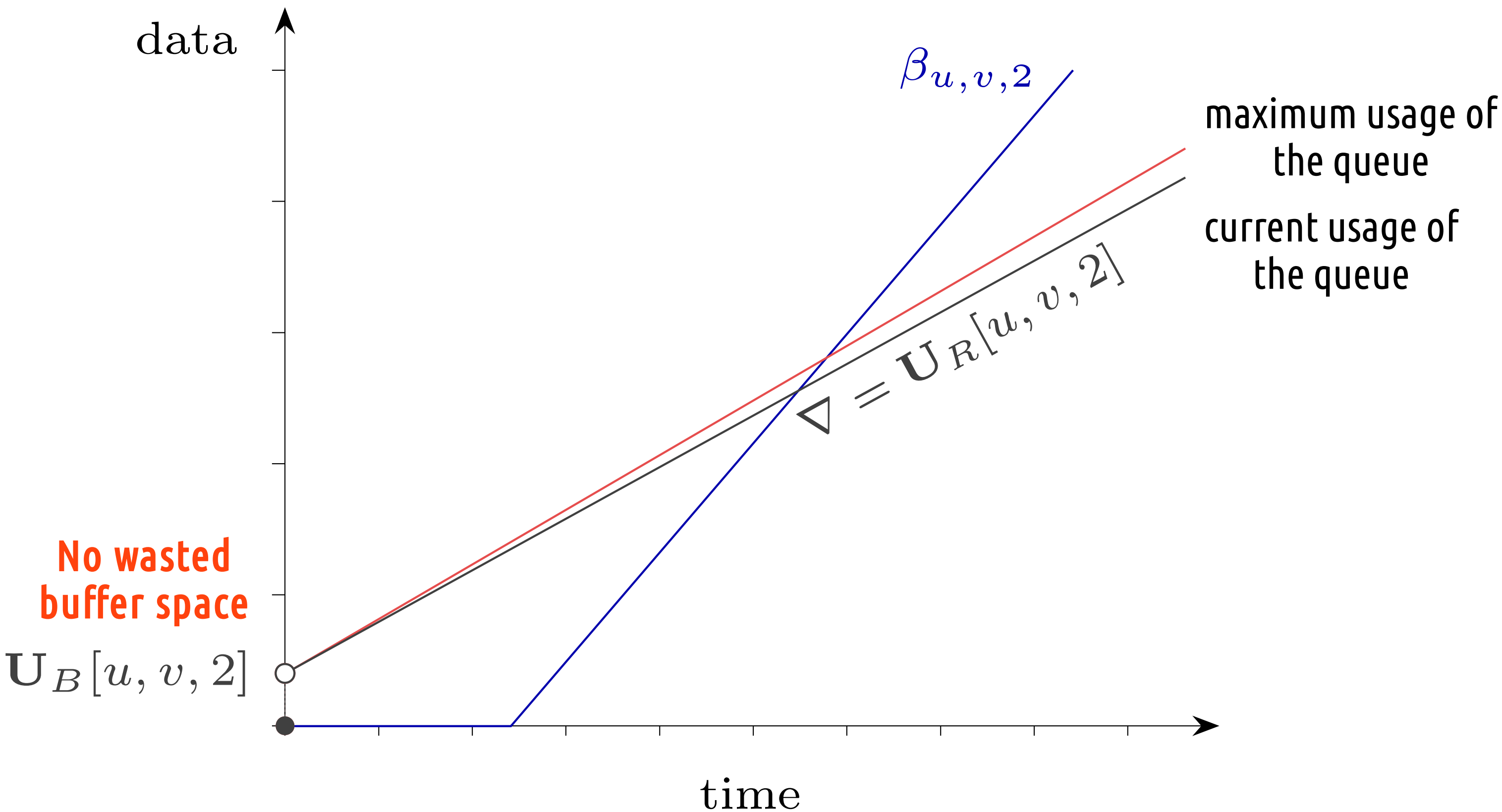


In such a situation, the MHM leads to a waste of resources

The buffer budget will never be used!

because the rate blocks acceptance of other flows

A solution is to artificially reduce the buffer budget!



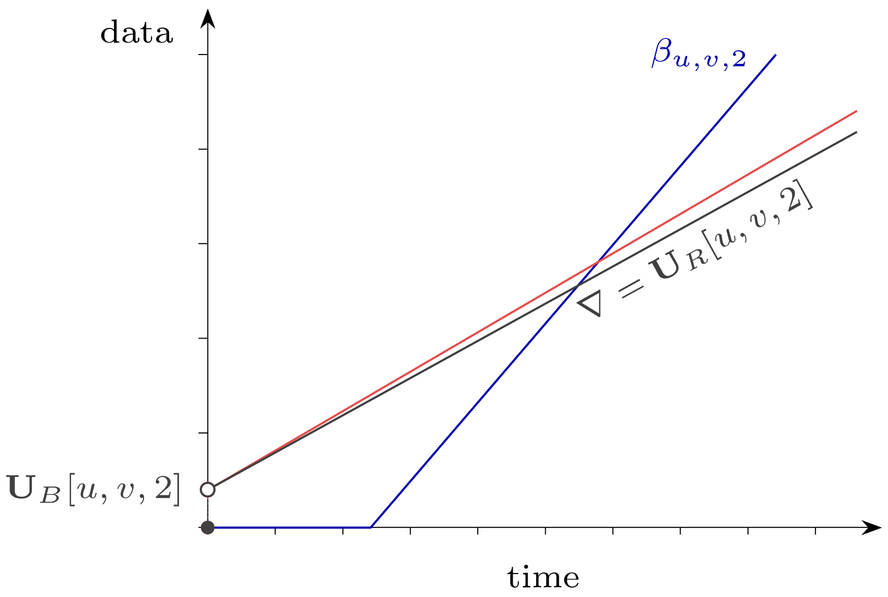
This also reduces the delay of the queue,
And hence the lower priority queues can have
- a lower delay, or
- a higher burst budget, or
- a higher data rate budget

delay/burst/data rate budget trade-off

The resource allocation hence has to **allocate**

- $\mathbf{A}_R[u, v, p]$: a data rate
- $\mathbf{M}_B[u, v, p]$: 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**...

$$\mathbf{T}[u,v,p] = \frac{\sum_{j=1}^{p} \mathbf{U}_B[u,v,j] + \max_{p+1 \le j \le Q_{u,v}} \{l_{u,v,j}^{max}\} + l_{u,v,p}^{max}}{R_{u,v} - \sum_{j=1}^{p-1} \mathbf{U}_R[u,v,j]}$$

The MHM does this by bounding $\mathbf{A}_R[u,v,p]$, $\mathbf{M}_B[u,v,p]$ and $l_{u,v,p}^{max}$

The resource allocation algorithm can rather bound the delay itself

$$\mathbf{T}[u,v,p] = \frac{\sum_{j=1}^{p} \mathbf{U}_B[u,v,j] + \max_{p+1 \le j \le Q_{u,v}} \{l_{u,v,j}^{max}\} + l_{u,v,p}^{max}}{R_{u,v} - \sum_{j=1}^{p-1} \mathbf{U}_R[u,v,j]} \le \mathbf{T}^{TBM}[u,v,p]$$

and let everything vary as long as

$$\mathbf{T}[u,v,p] \le \mathbf{T}^{TBM}[u,v,p] \qquad \forall (u,v,p)$$

$$B_{max}(u,v,p) \le \mathbf{A}_B[u,v,p] \qquad \forall\ (u,v,p)$$

The resource allocation algorithm can rather bound the delay itself

$$\mathbf{T}[u,v,p] = \frac{\sum_{j=1}^{p} \mathbf{U}_B[u,v,j] + \max\limits_{p+1 \leq j \leq Q_{u,v}} \{l_{u,v,j}^{max}\} + l_{u,v,p}^{max}}{R_{u,v} - \sum_{j=1}^{p-1} \mathbf{U}_R[u,v,j]} \leq \mathbf{T}^{TBM}[u,v,p]$$

set by the resource allocation algorithm

and let everything vary as long as

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

$$B_{max}(u,v,p) \leq \mathbf{A}_B[u,v,p] \qquad \forall\ (u,v,p)$$

$$B_{max}(u,v,p) = \mathbf{U}_B[u,v,p] + \mathbf{U}_R[u,v,p] \frac{\sum_{j=1}^{p-1} \mathbf{U}_B[u,v,j] + \max\limits_{p+1 \leq j \leq Q_{u,v}} \{l_{u,v,j}^{max}\} + l_{u,v,p}^{max}}{R_{u,v} - \sum_{j=1}^{p-1} \mathbf{U}_R[u,v,j]}$$

$$\leq \mathbf{A}_B[u,v,p]$$

buffer capacity

The resource allocation algorithm can rather bound the delay itself

$$\mathbf{T}[u,v,p] = \frac{\sum_{j=1}^{p} \mathrm{U}_B[u,v,j] + \max_{p+1\leq j\leq Q_{u,v}} \{l_{u,v,j}^{max}\} + l_{u,v,p}^{max}}{R_{u,v} - \sum_{j=1}^{p-1} \mathrm{U}_R[u,v,j]} \leq \mathbf{T}^{TBM}[u,v,p]$$

set by the resource allocation algorithm

and let everything vary as long as

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

$$B_{max}(u,v,p) \leq \mathbf{A}_B[u,v,p] \qquad \forall\ (u,v,p)$$

$$B_{max}(u,v,p) = \mathbf{U}_B[u,v,p] + \mathbf{U}_R[u,v,p] \frac{\sum_{j=1}^{p-1} \mathrm{U}_B[u,v,j] + \max_{p+1\leq j\leq Q_{u,v}} \{l_{u,v,j}^{max}\} + l_{u,v,p}^{max}}{R_{u,v} - \sum_{j=1}^{p-1} \mathrm{U}_R[u,v,j]}$$

$$\leq \mathbf{A}_B[u,v,p]$$

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

$$\mathbf{T}[u,v,p] = \frac{\sum_{j=1}^{p} \mathrm{U}_B[u,v,j] + 1542 + l_{u,v,p}^{max}}{R_{u,v} - \sum_{j=1}^{p-1} \mathrm{U}_R[u,v,j]} \leq \mathbf{T}^{TBM}[u,v,p]$$

set by the resource allocation algorithm

and let everything vary as long as

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

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

$$B_{max}(u,v,p) = \mathbf{U}_B[u,v,p] + \mathbf{U}_R[u,v,p] \frac{\sum_{j=1}^{p-1} \mathrm{U}_B[u,v,j] + 1542 + l_{u,v,p}^{max}}{R_{u,v} - \sum_{j=1}^{p-1} \mathrm{U}_R[u,v,j]}$$

$$\leq \mathbf{A}_B[u,v,p]$$

buffer capacity

Requires to check **lower priority queues** and ~~higher priority queues~~
before the addition of a new flow

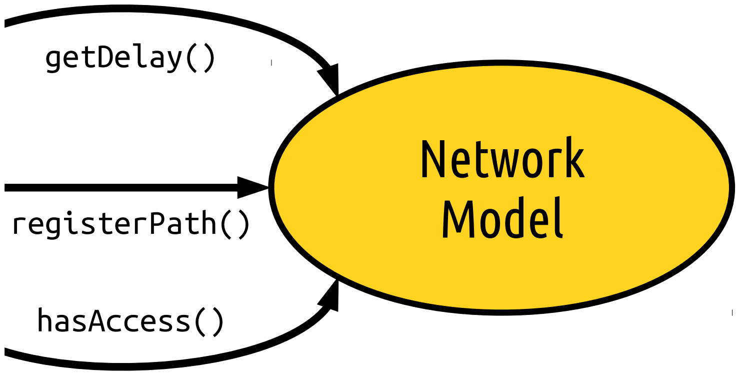
because there might be **unknown** best-effort traffic in the lowest priority queue

The Threshold-based Model (TBM)

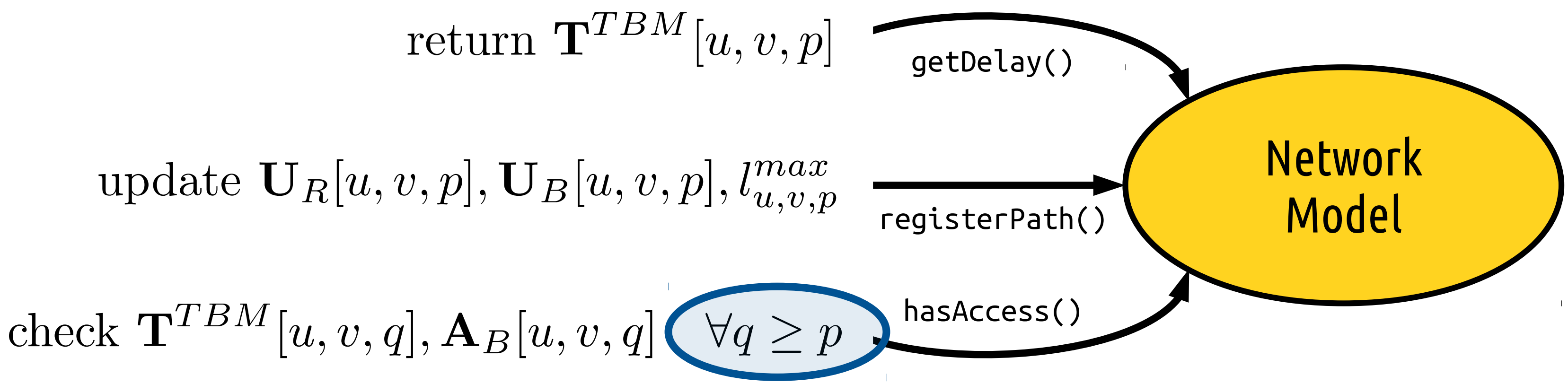
return $\mathbf{T}^{TBM}[u, v, p]$

update $\mathbf{U}_R[u, v, p], \mathbf{U}_B[u, v, p], l_{u,v,p}^{max}$

check $\mathbf{T}^{TBM}[u, v, q], \mathbf{A}_B[u, v, q] \quad \forall q \geq p$



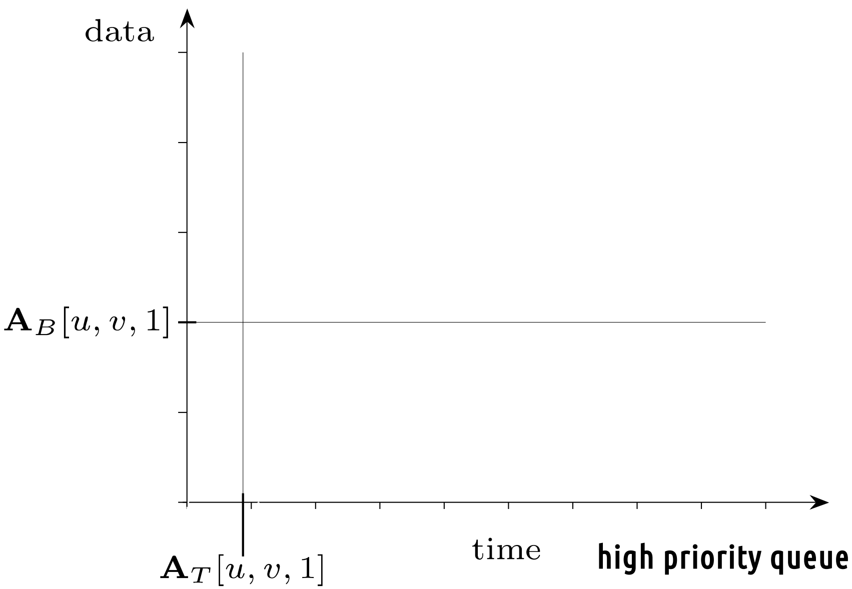
The Threshold-based Model (TBM)

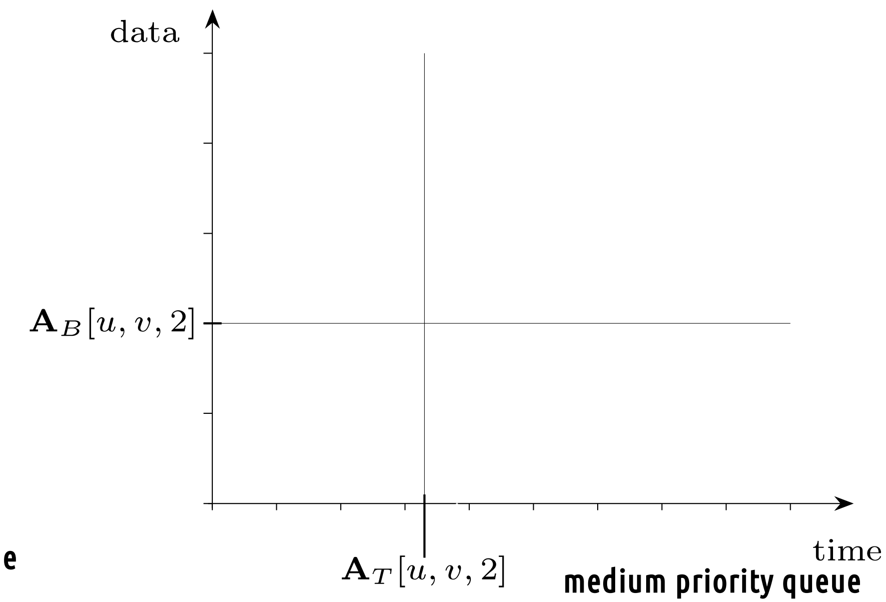


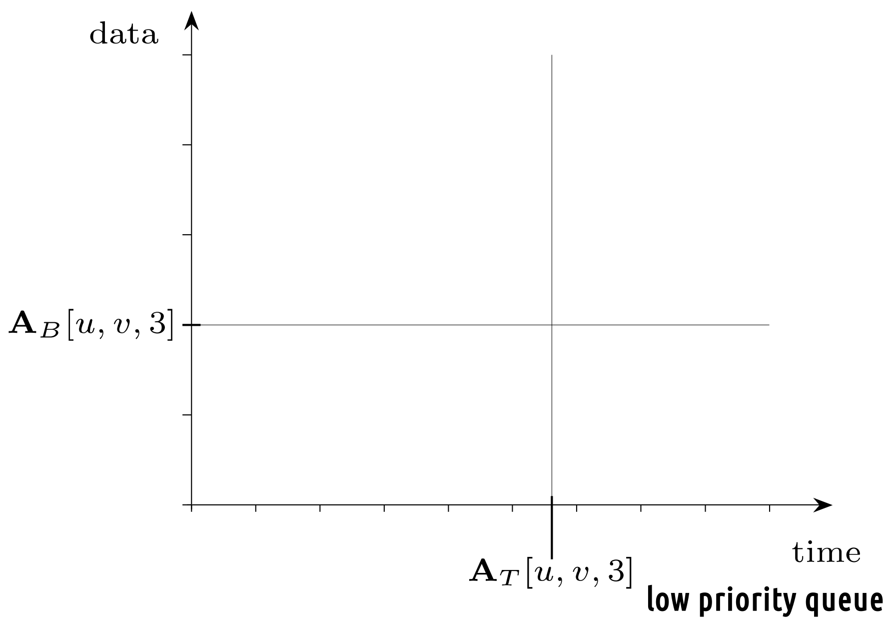
Requires up to $Q_{u,v}$ times more work than the MHM

But no a priori choice on the burst/rate/delay trade-off

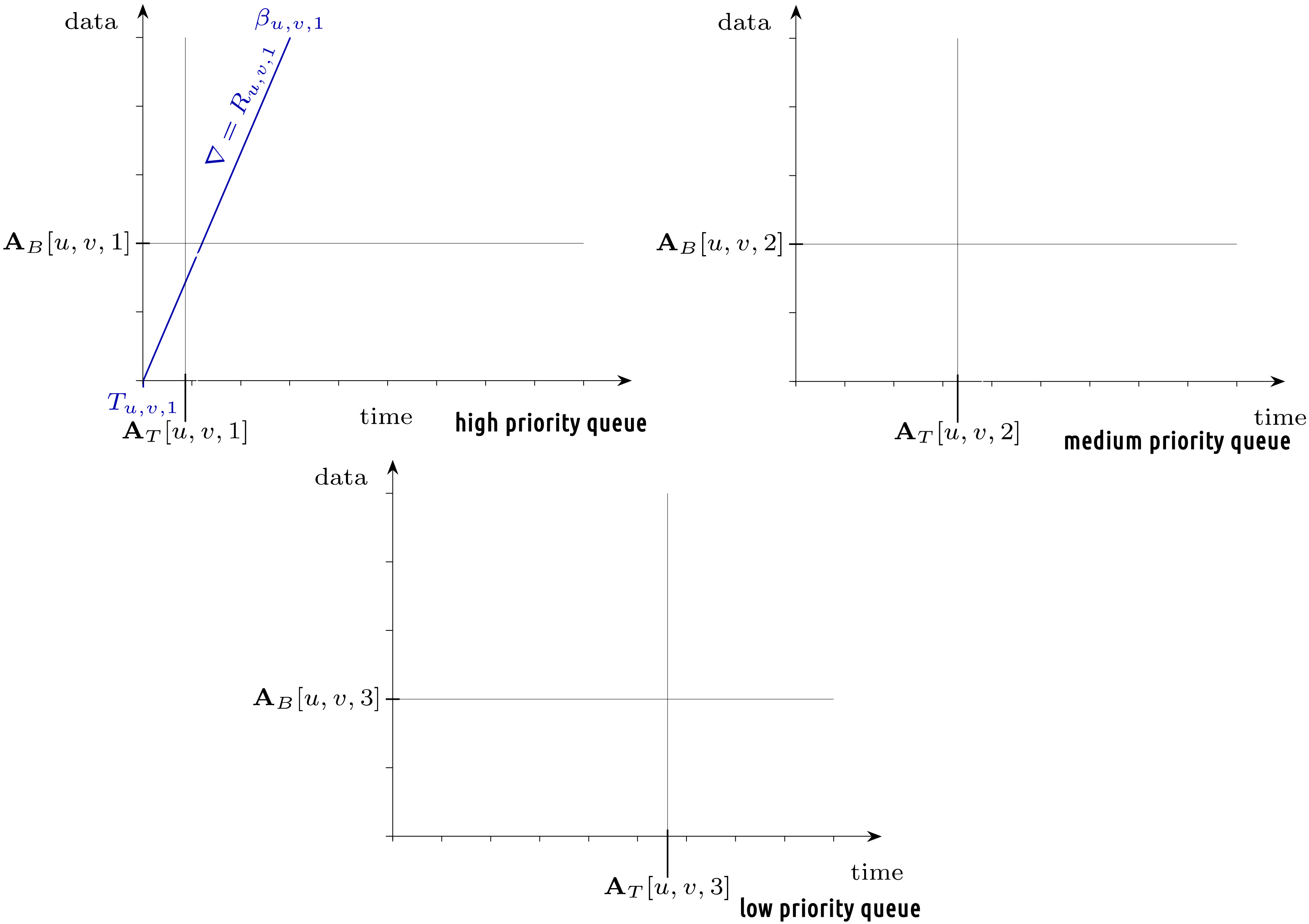
Let‘s see how this looks like graphically... at a link with 3 priority queues



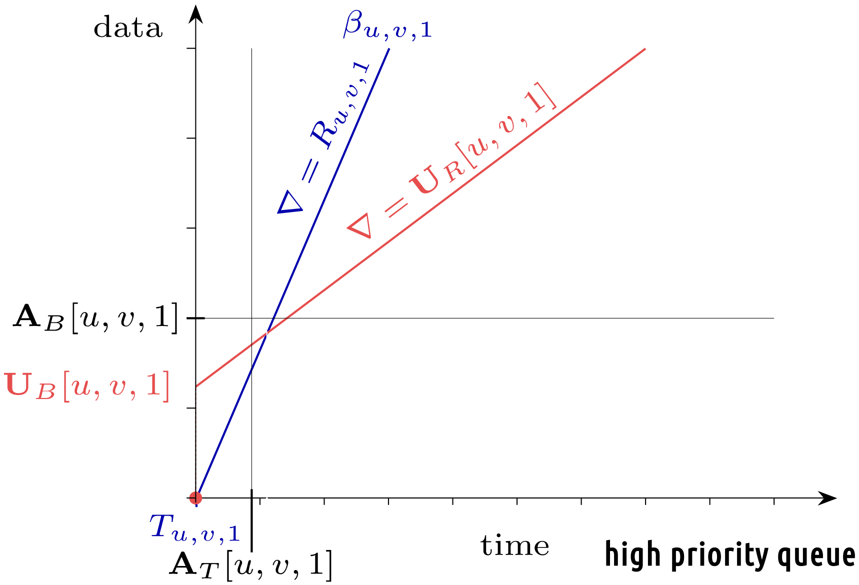


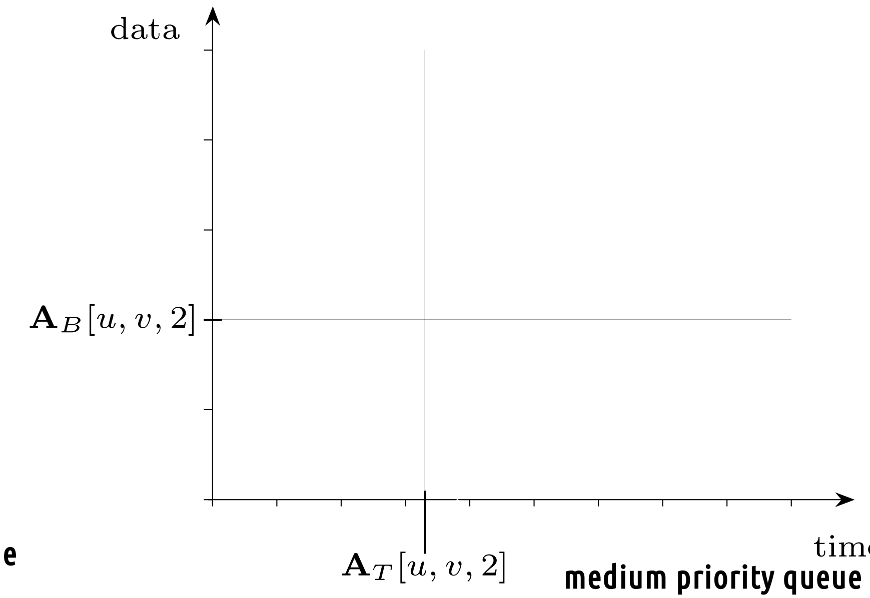


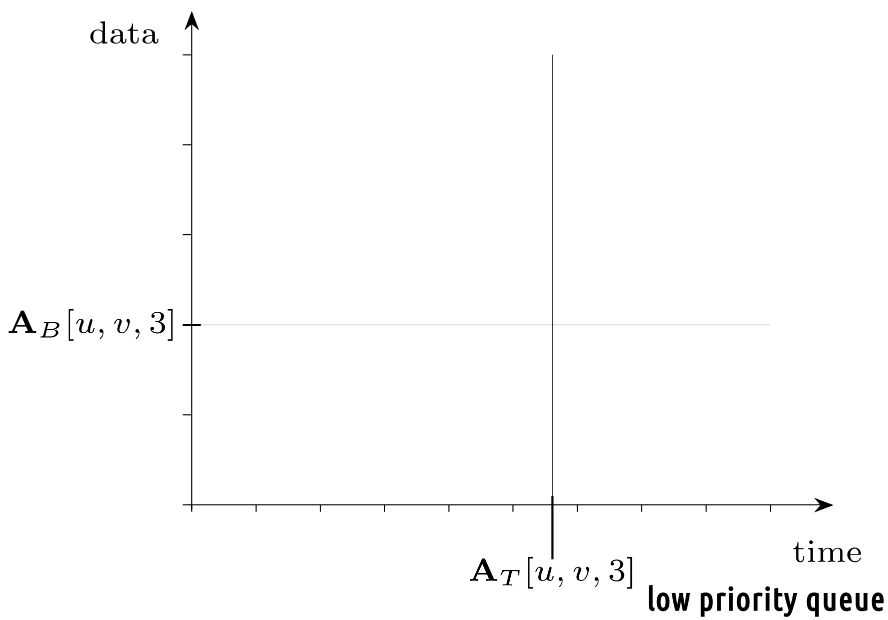
Let‘s see how this looks like graphically… at a link with 3 priority queues



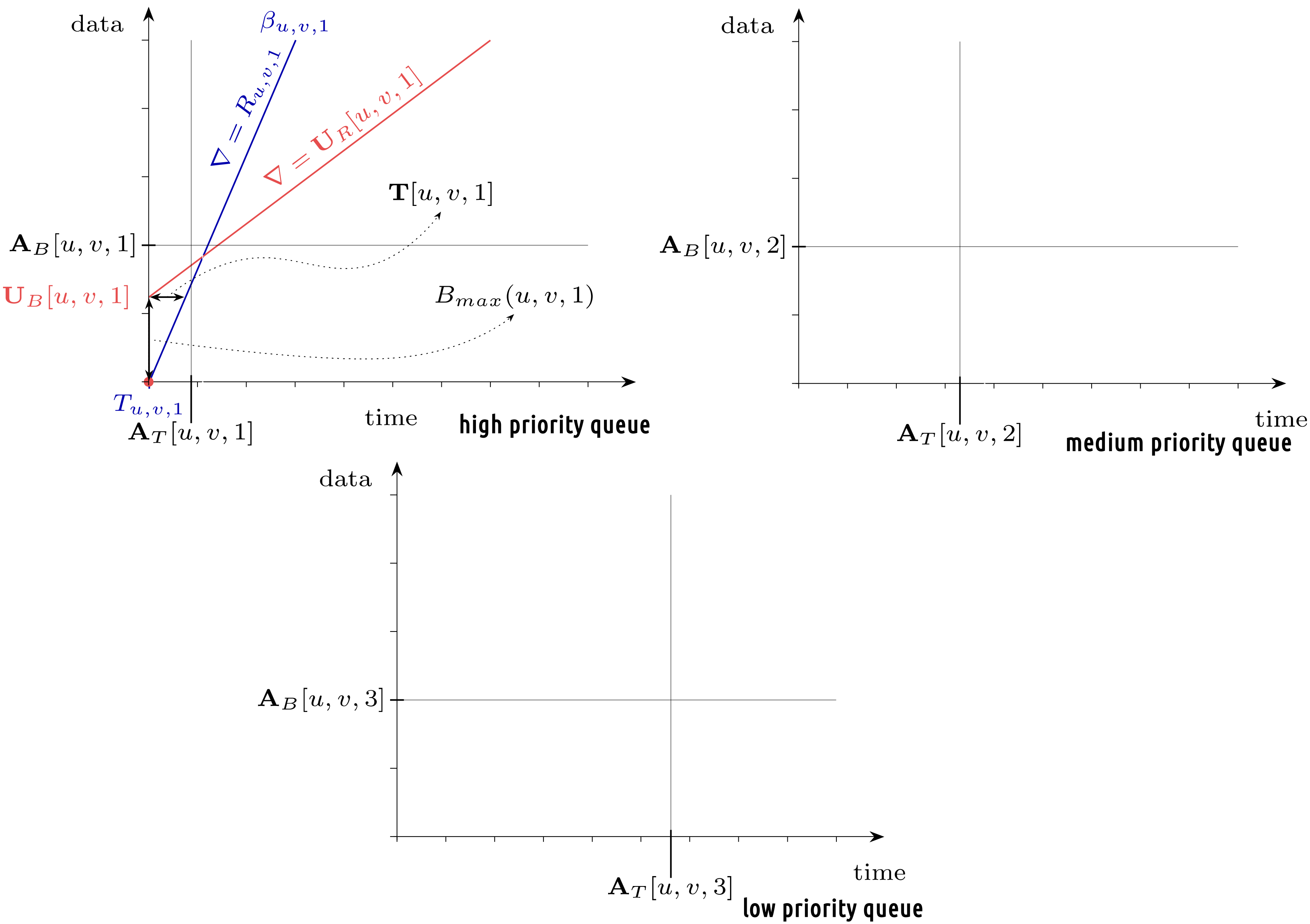
Let‘s see how this looks like graphically… at a link with 3 priority queues



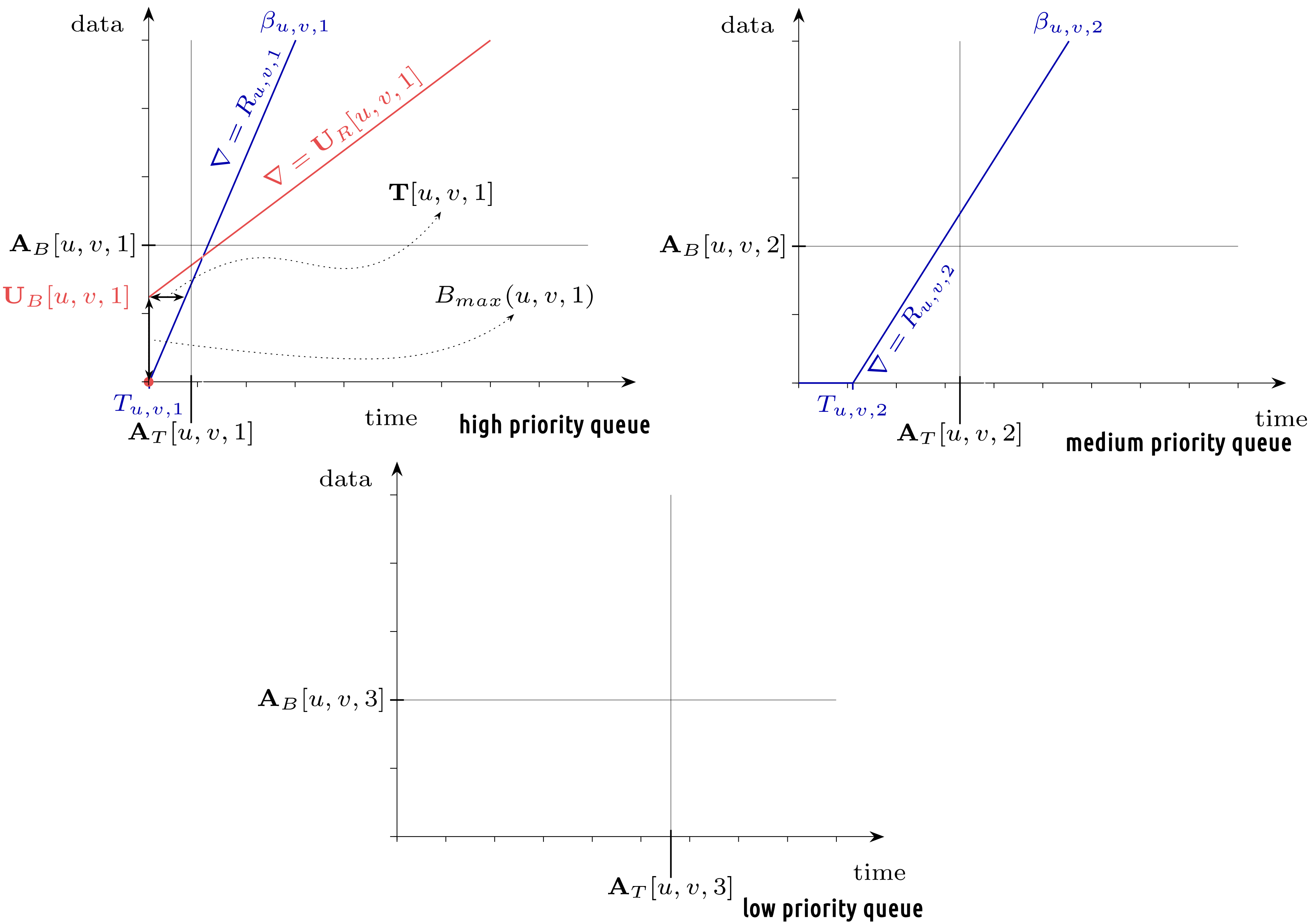




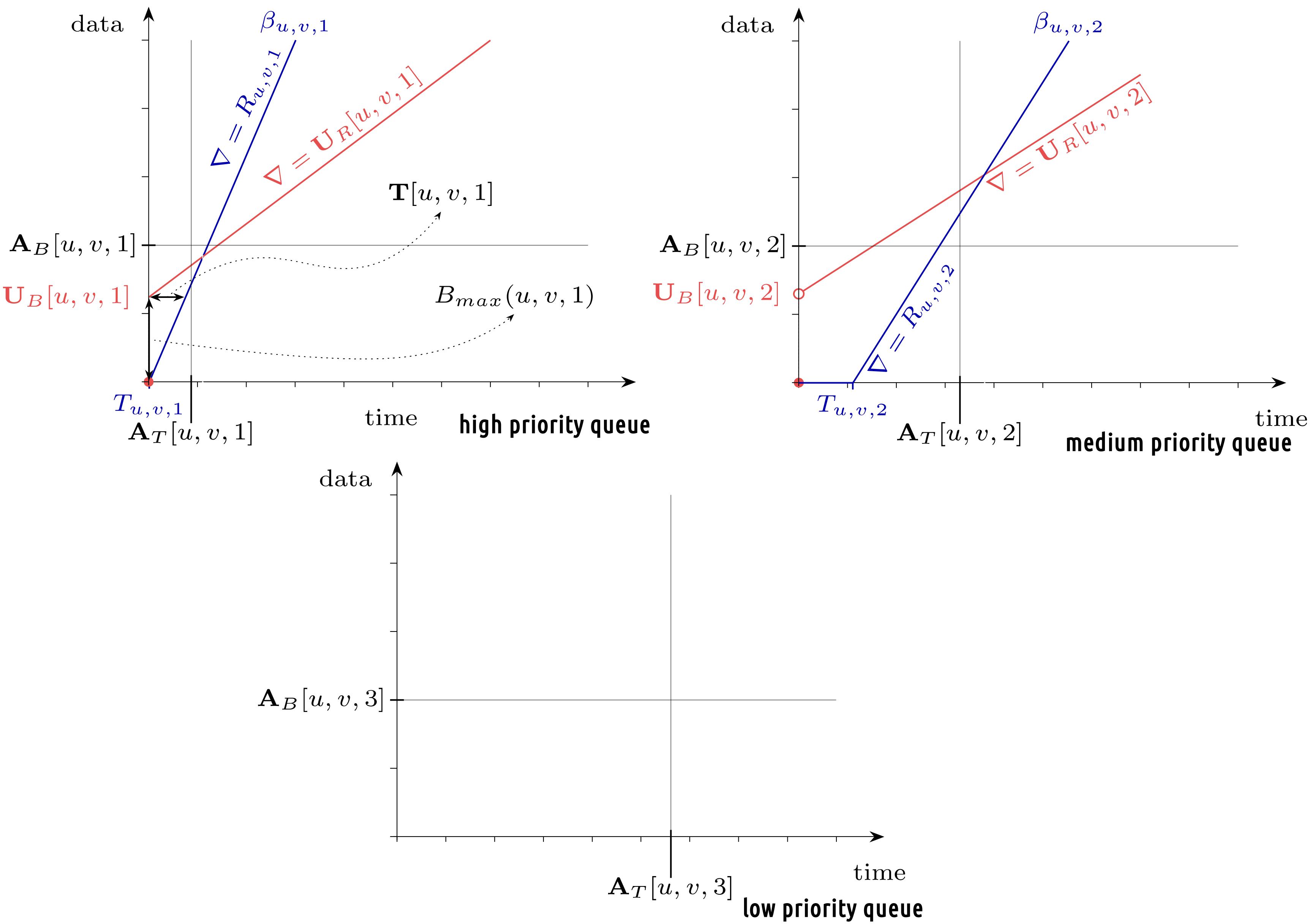
Let‘s see how this looks like graphically… at a link with 3 priority queues



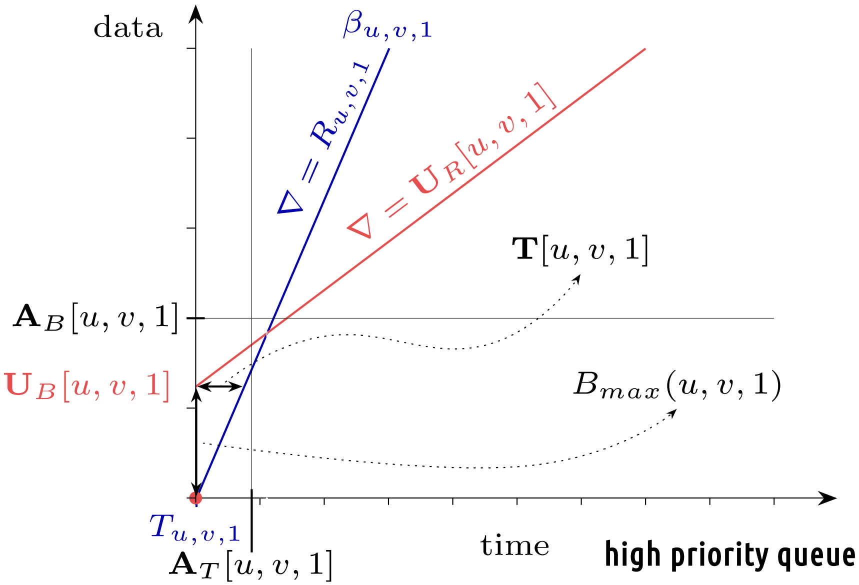
Let‘s see how this looks like graphically… at a link with 3 priority queues

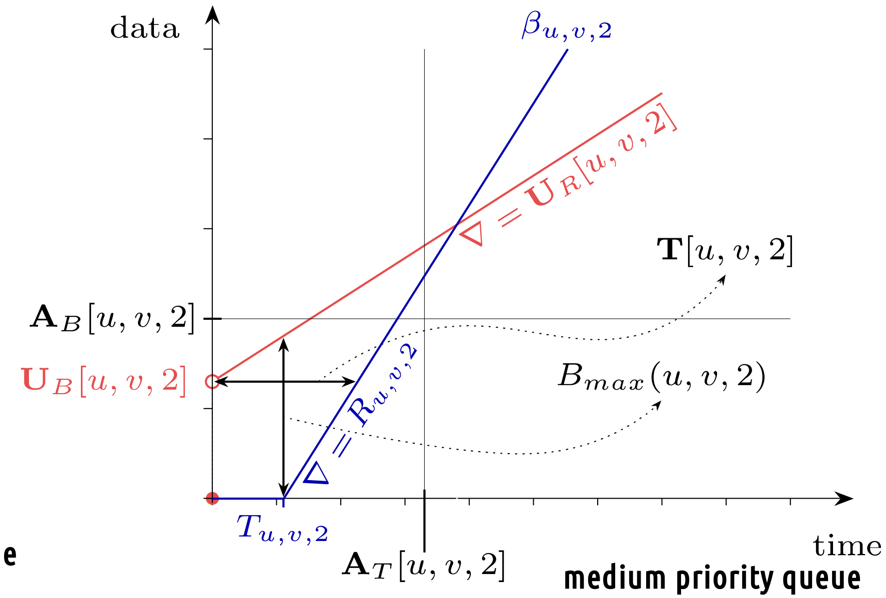


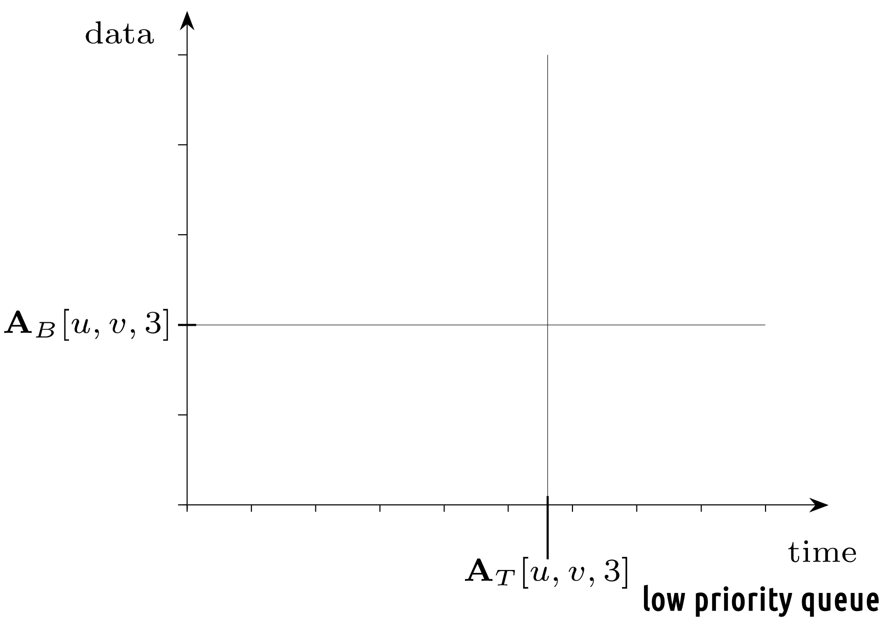
Let‘s see how this looks like graphically... at a link with 3 priority queues



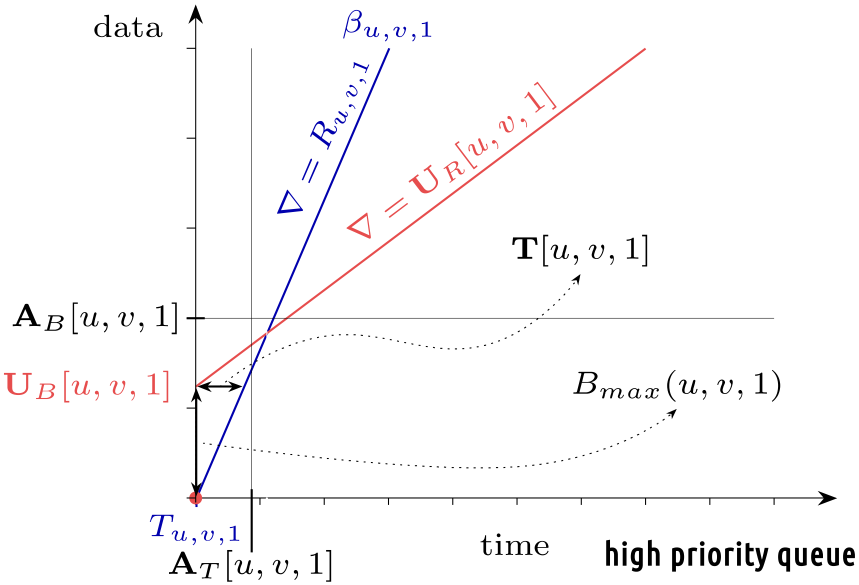
Let‘s see how this looks like graphically… at a link with 3 priority queues

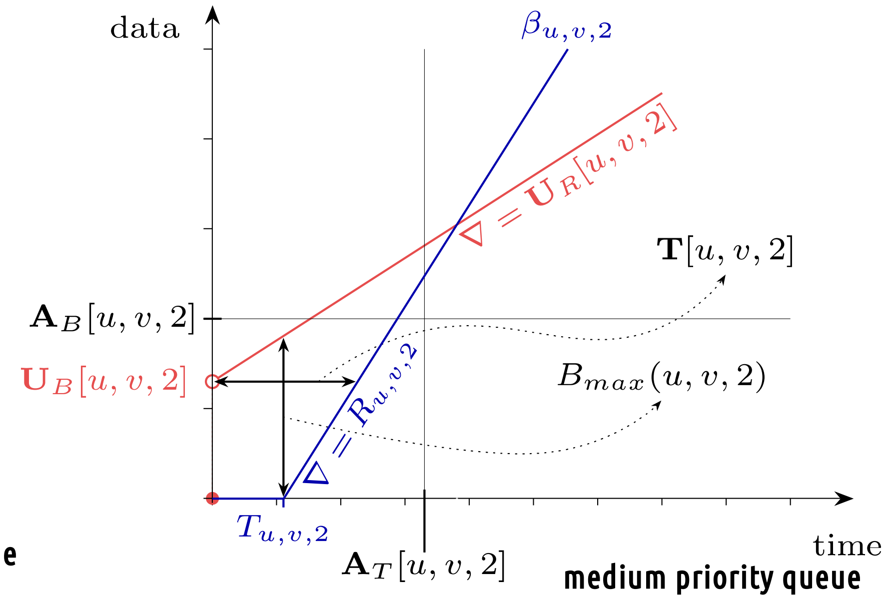


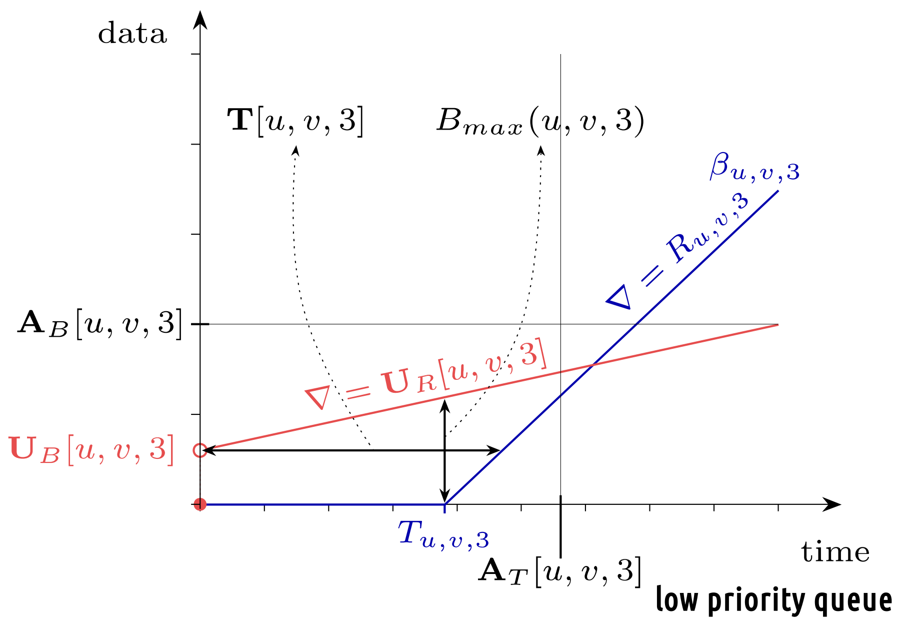




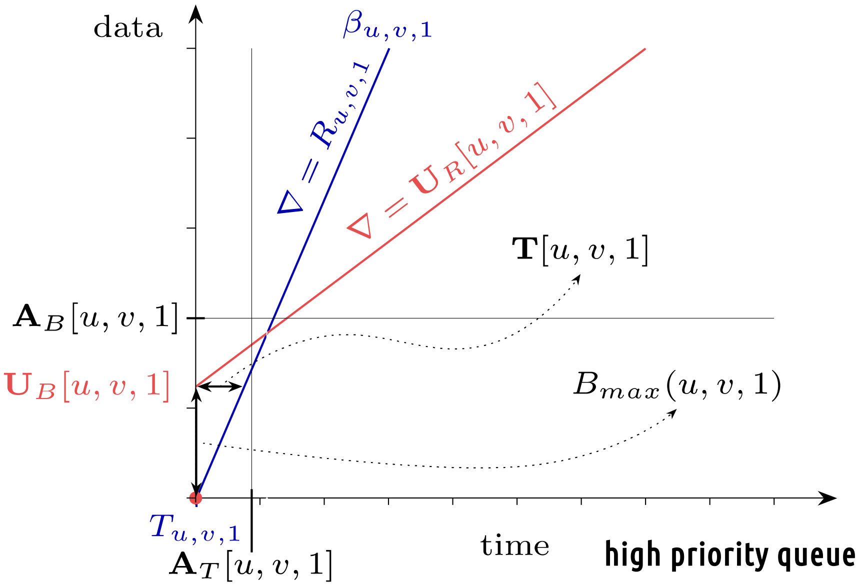
Let‘s see how this looks like graphically… at a link with 3 priority queues

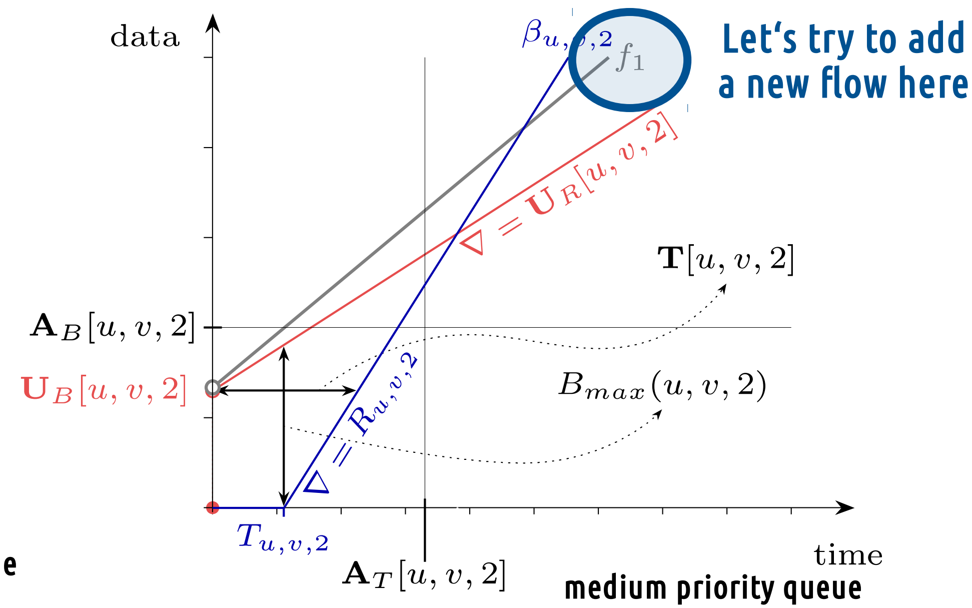


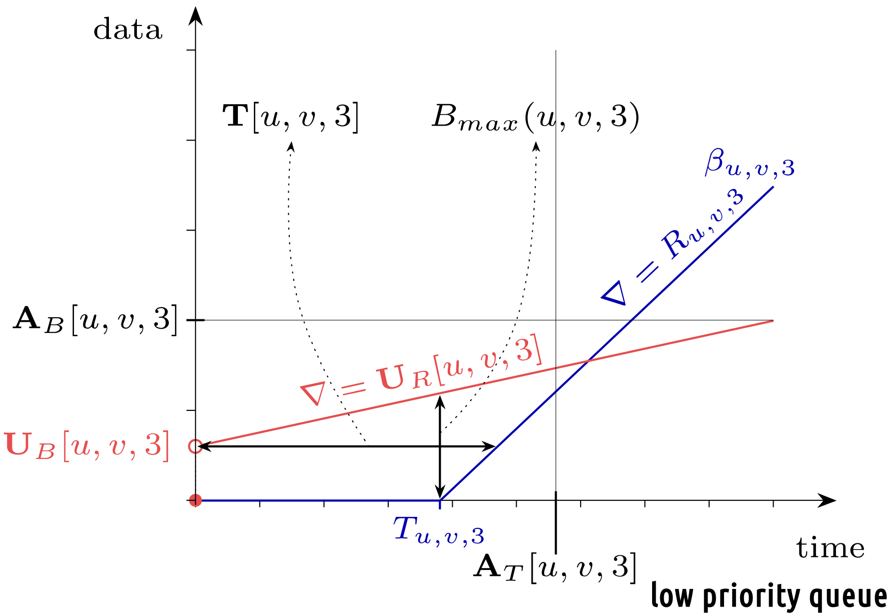




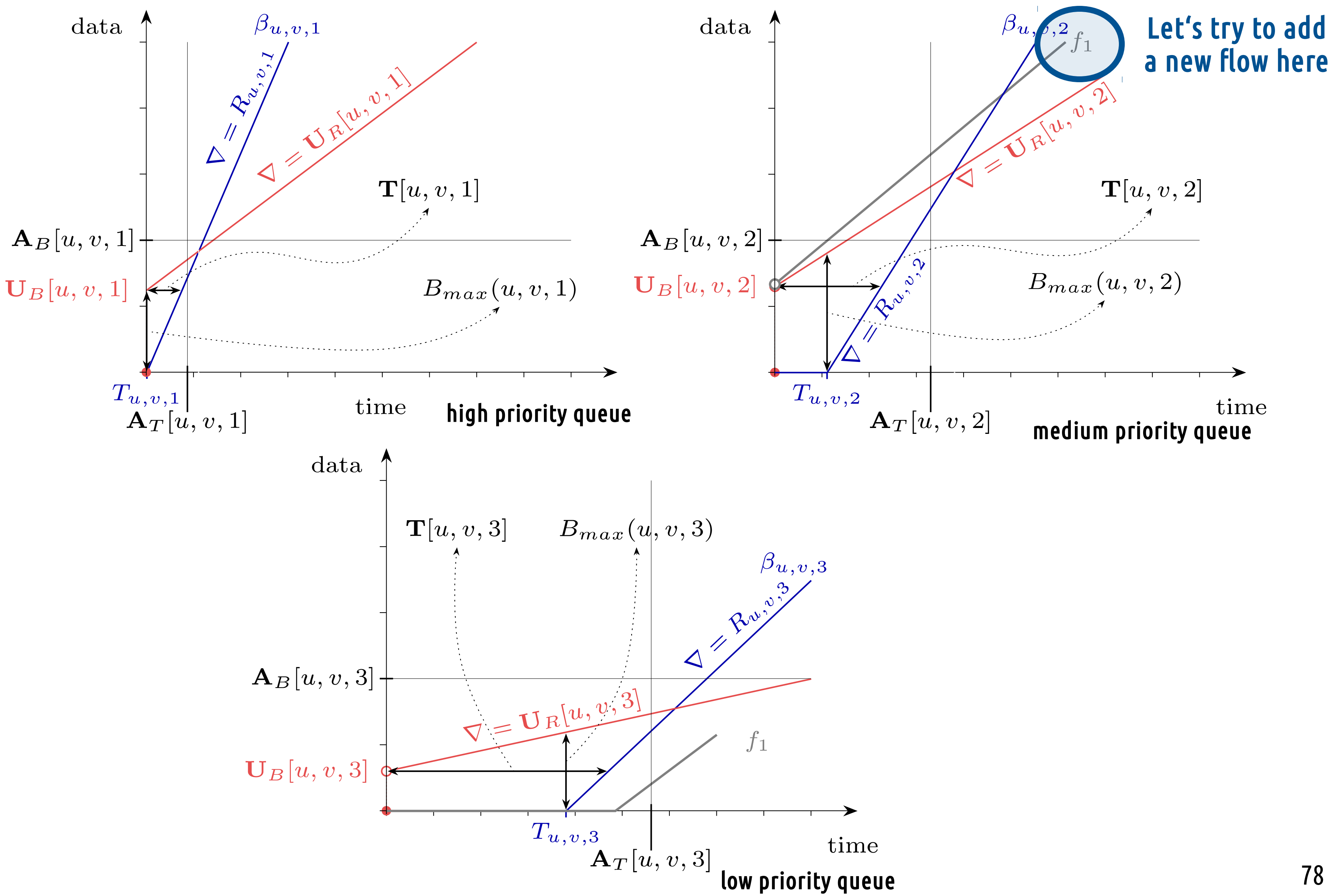
Let‘s see how this looks like graphically… at a link with 3 priority queues



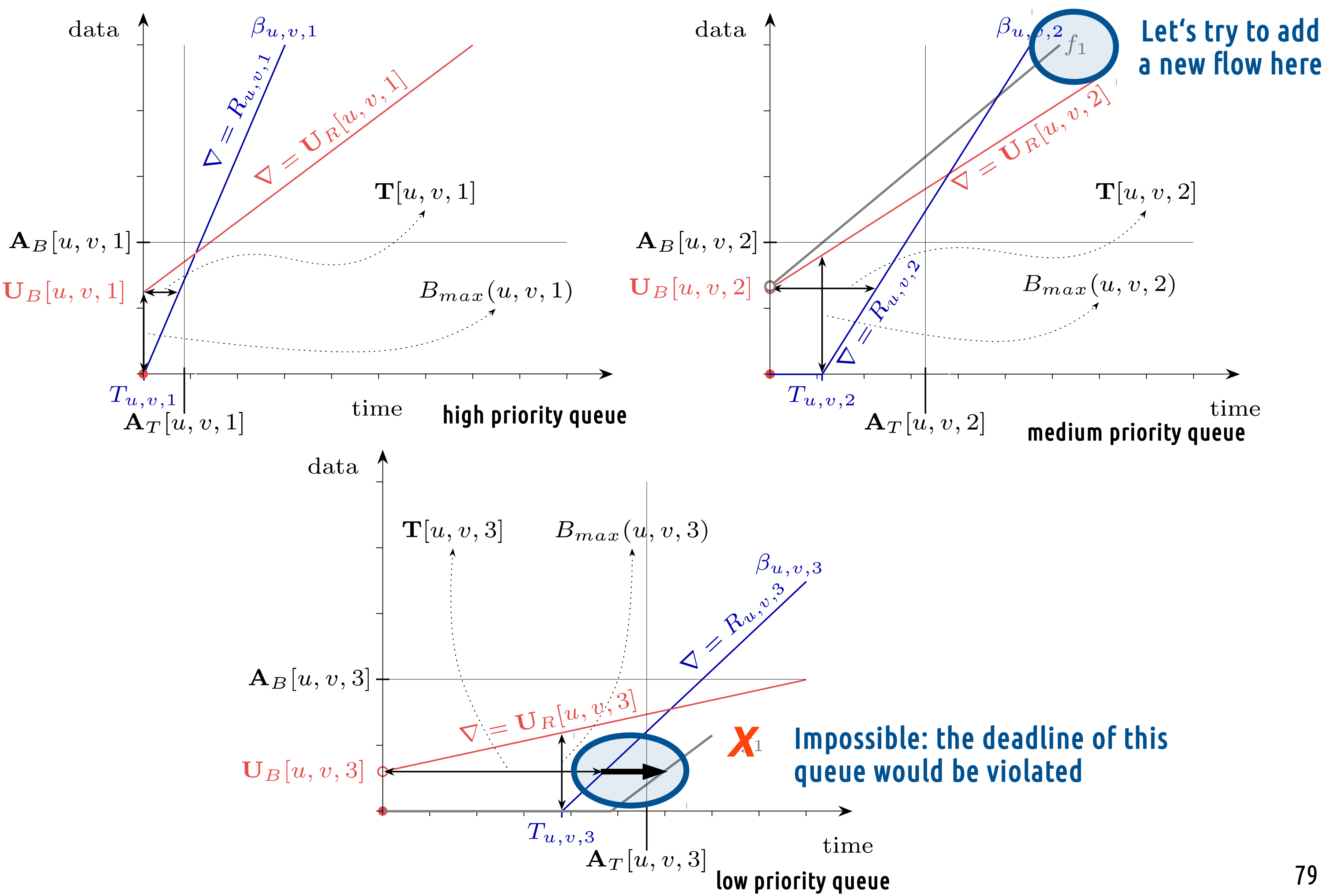




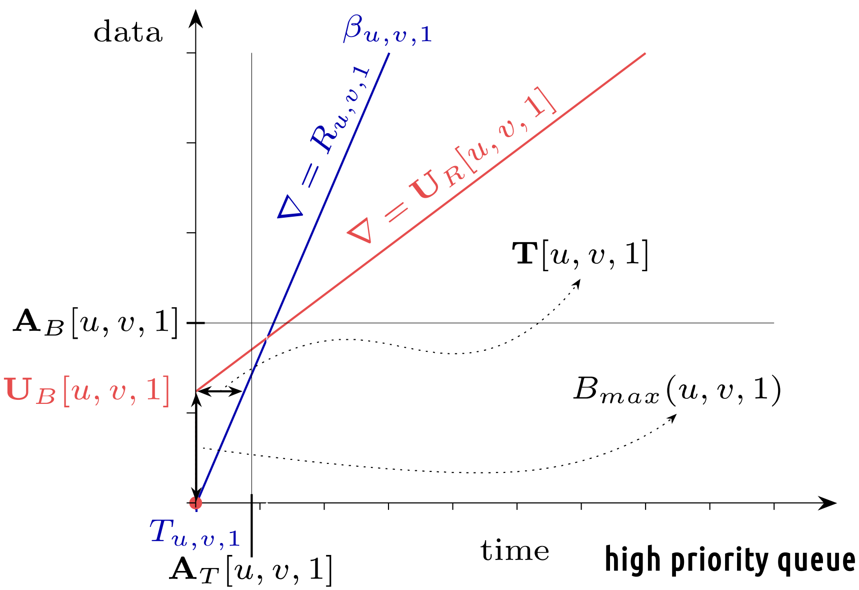
Let's see how this looks like graphically... at a link with 3 priority queues

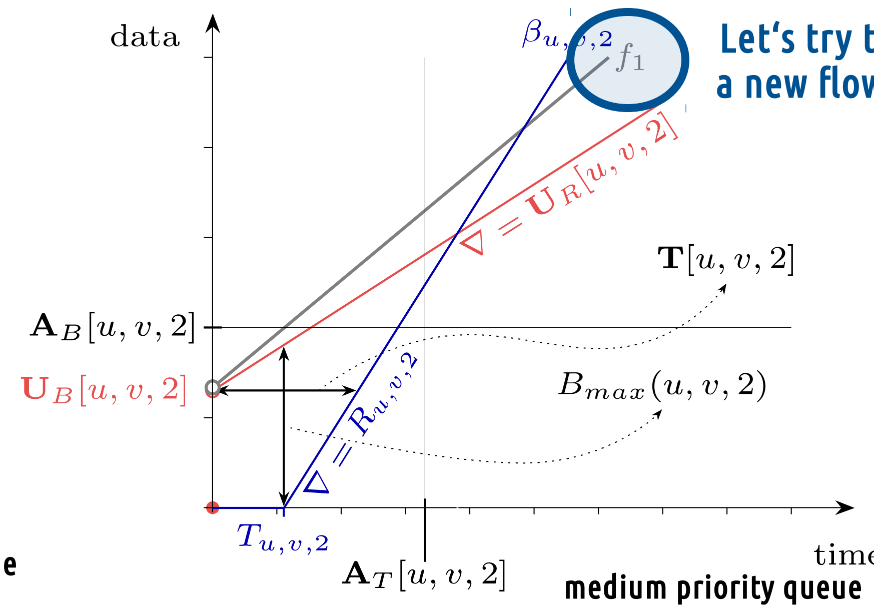


Let's see how this looks like graphically… at a link with 3 priority queues

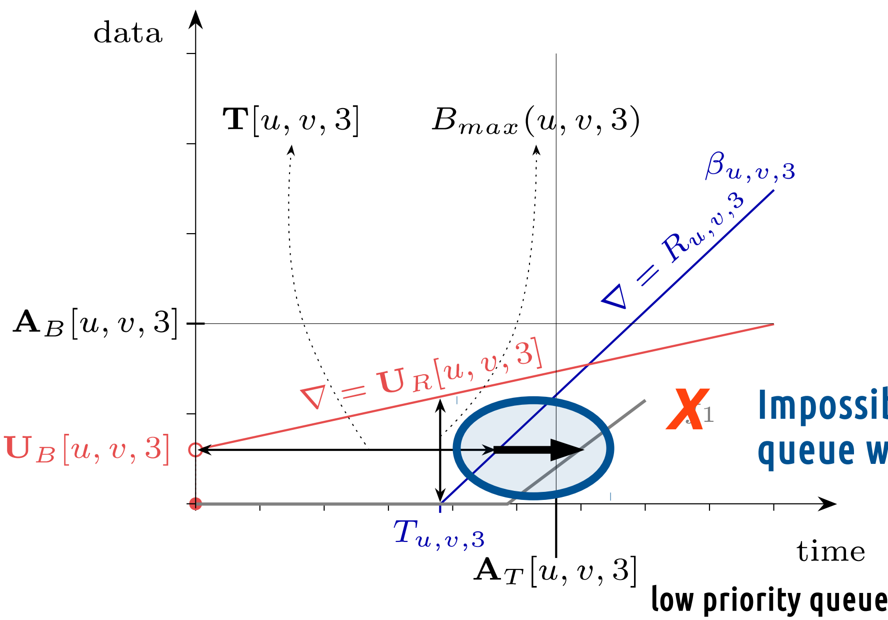


Let‘s see how this looks like graphically… at a link with 3 priority queues





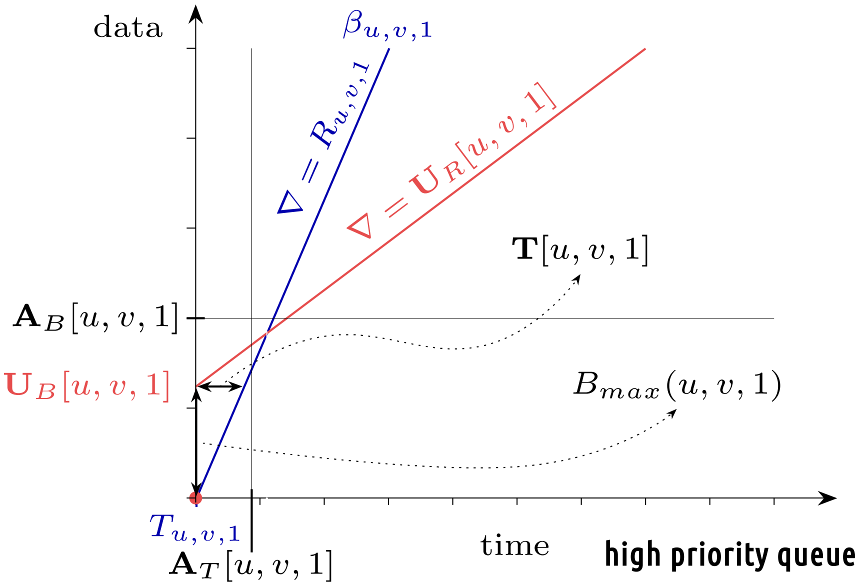
Let‘s try to add a new flow here

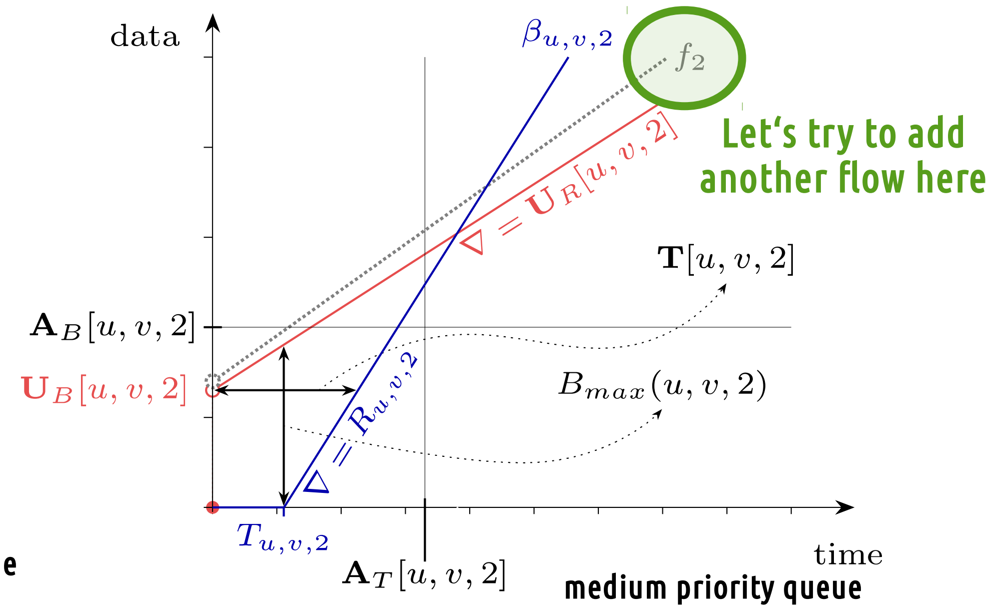


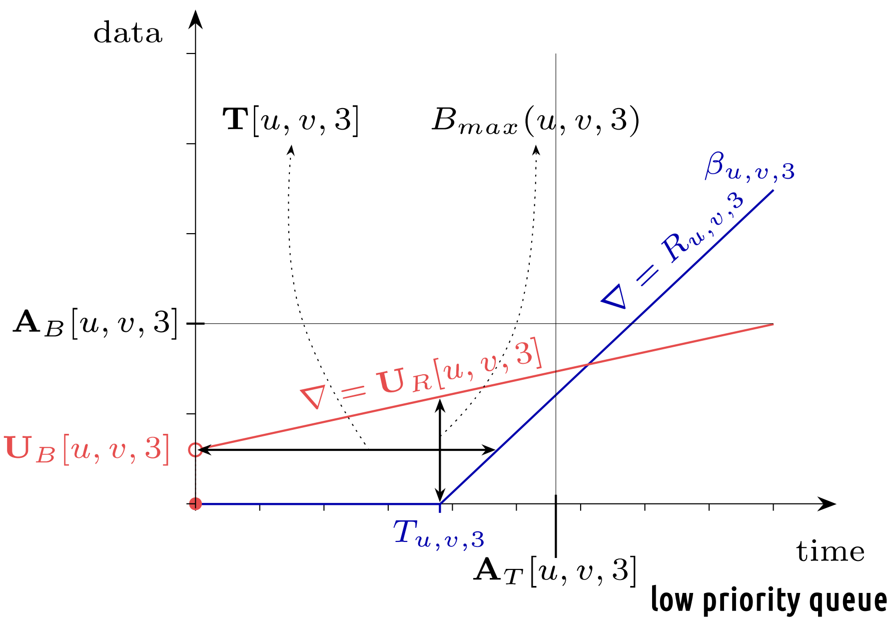
Low priority blocking problem

Impossible: the deadline of this queue would be violated

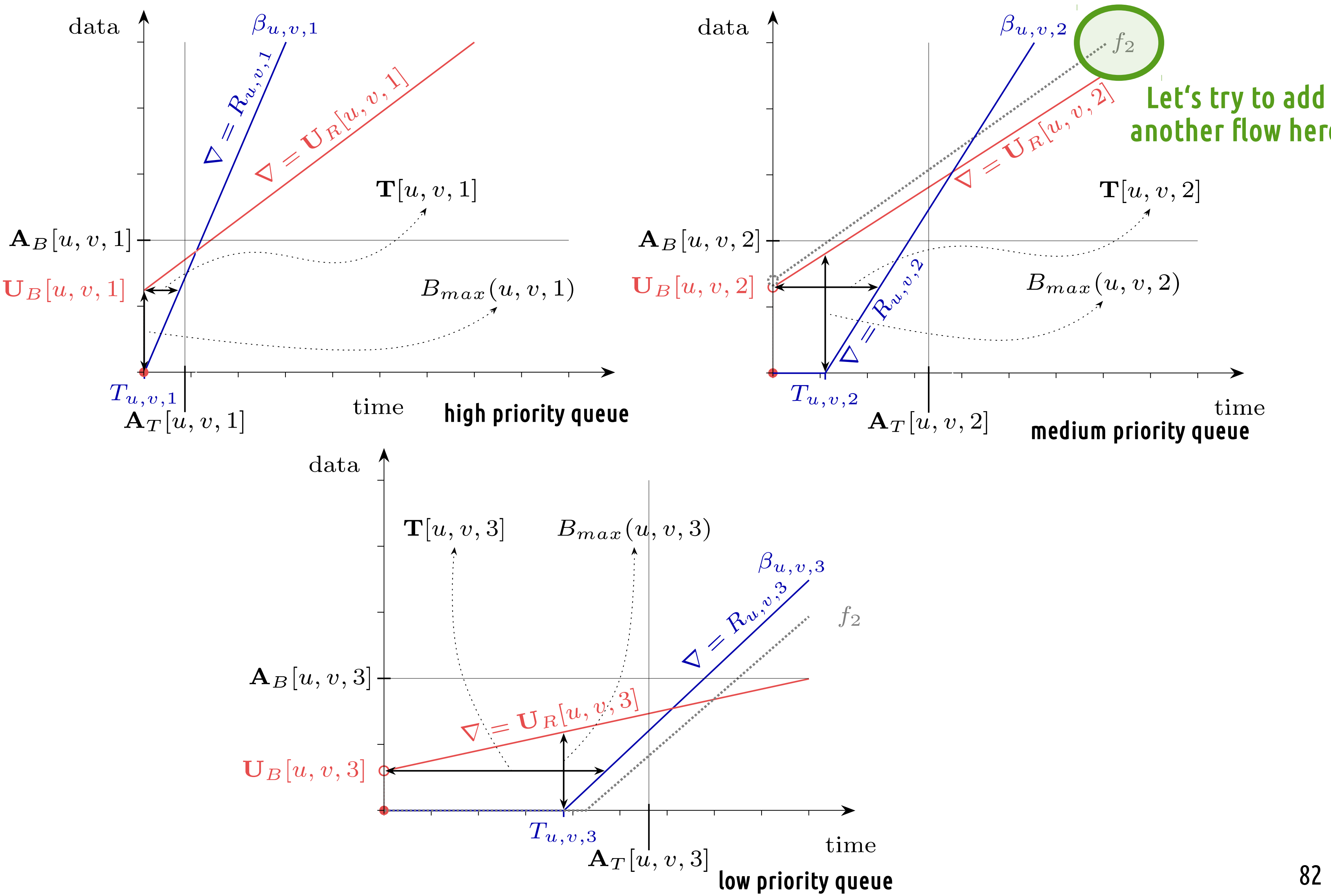
Let‘s see how this looks like graphically... at a link with 3 priority queues



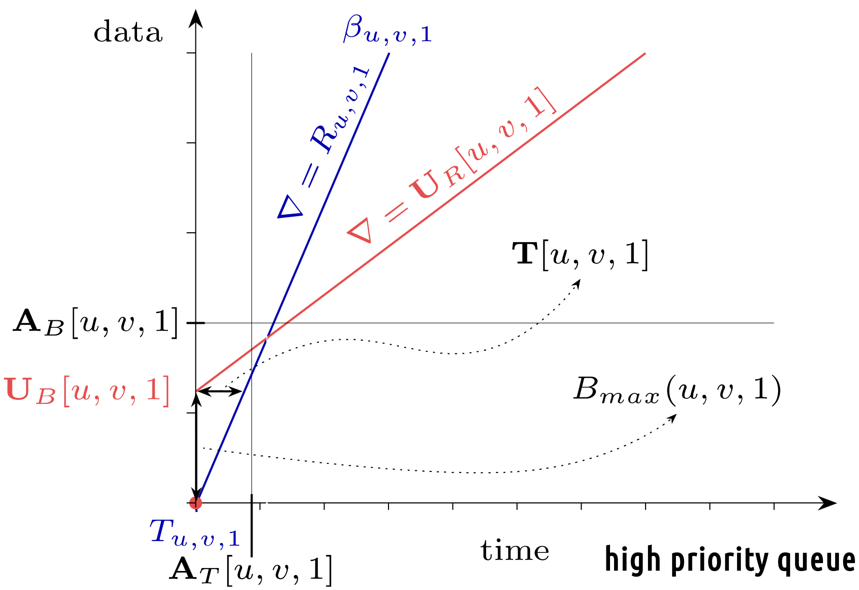


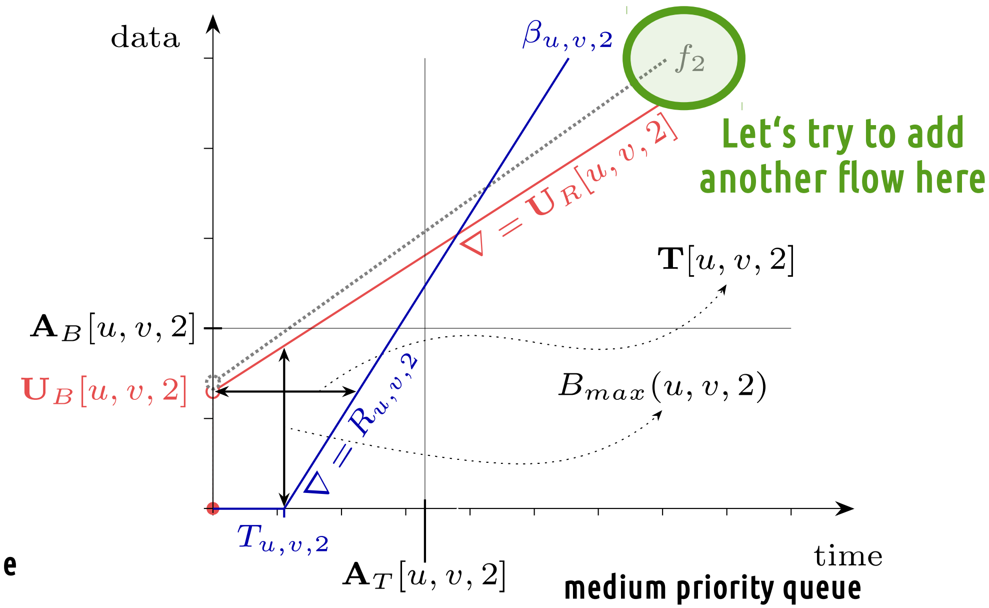


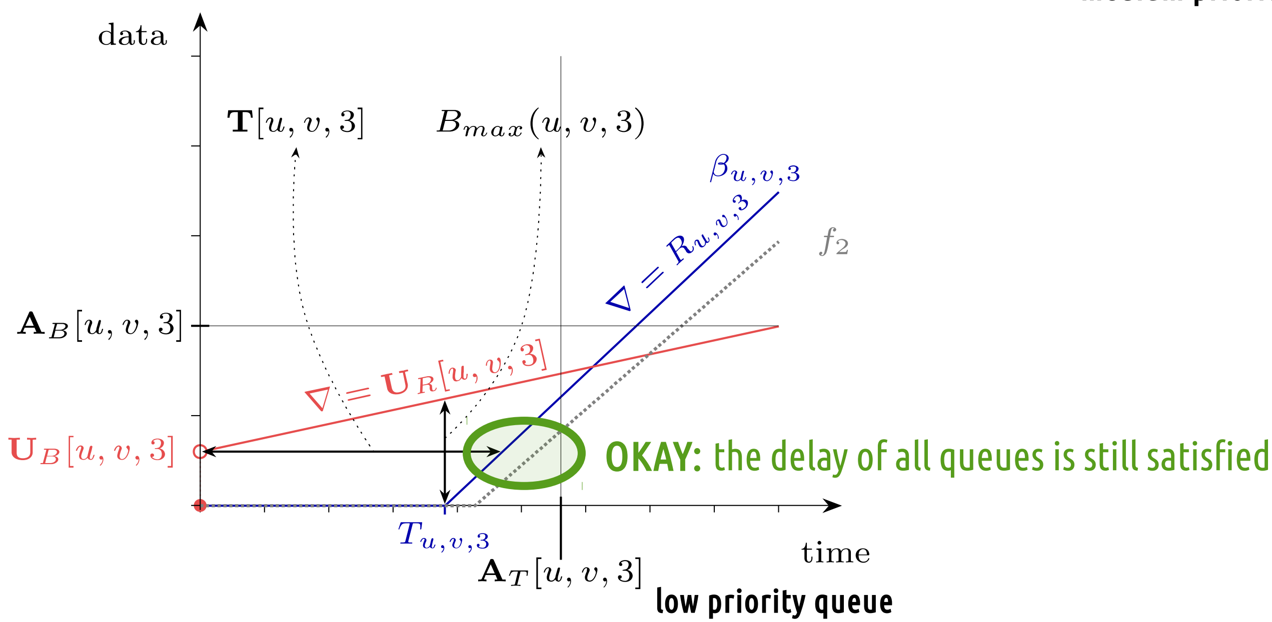
Let's see how this looks like graphically... at a link with 3 priority queues



Let‘s see how this looks like graphically… at a link with 3 priority queues





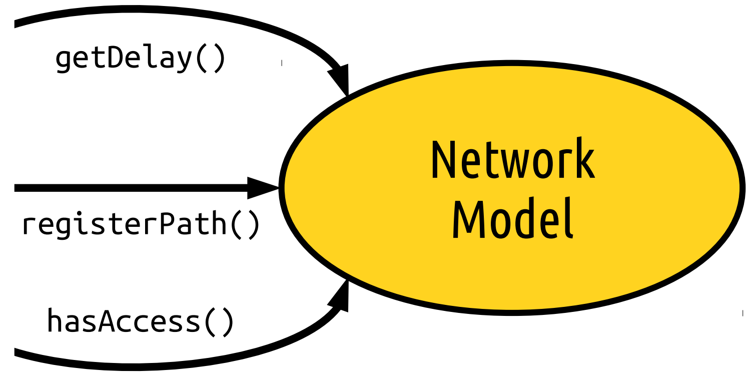


The Multi-Hop Model (MHM)

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

update $\mathbf{U}_R[u, v, p], \mathbf{U}_B[u, v, p]$

check $\mathbf{M}_B[u, v, p], \mathbf{A}_R[u, v, p]$ are not exceeded

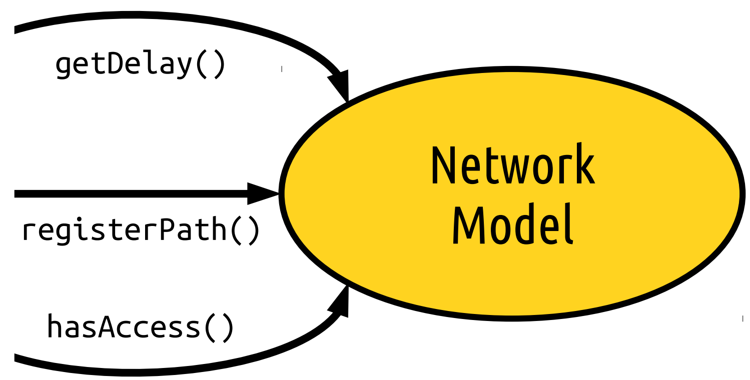


The Threshold-based Model (TBM)

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

update $\mathbf{U}_R[u, v, p], \mathbf{U}_B[u, v, p], l^{max}_{u,v,p}$

check $\mathbf{T}^{TBM}[u, v, q], \mathbf{A}_B[u, v, q] \quad \forall q \geq p$

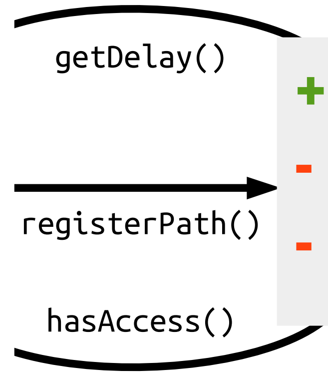


The Multi-Hop Model (MHM)

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

update $\mathbf{U}_R[u,v,p], \mathbf{U}_B[u,v,p]$

check $\mathbf{M}_B[u,v,p], \mathbf{A}_R[u,v,p]$ are not exceeded



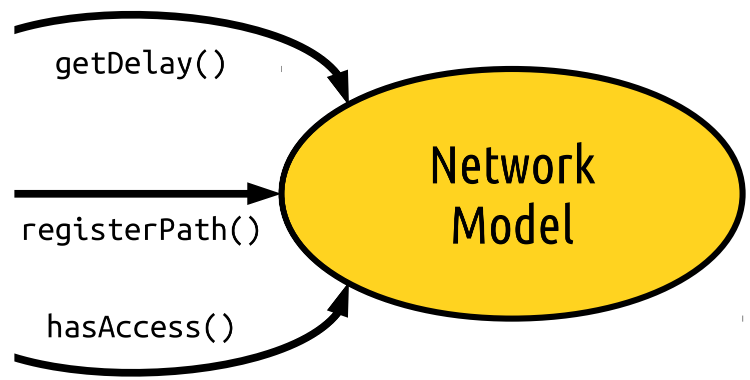
- \+ fast
- \- a priori choice
- \- 2 parameters to allocate

The Threshold-based Model (TBM)

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

update $\mathbf{U}_R[u,v,p], \mathbf{U}_B[u,v,p], l^{max}_{u,v,p}$

check $\mathbf{T}^{TBM}[u,v,q], \mathbf{A}_B[u,v,q] \quad \forall q \geq p$



The Multi-Hop Model (MHM)

return $\mathbf{T}^{MHM}[u,v,p]$ — getDelay()

update $\mathbf{U}_R[u,v,p], \mathbf{U}_B[u,v,p]$ — registerPath()

check $\mathbf{M}_B[u,v,p], \mathbf{A}_R[u,v,p]$ are not exceeded — hasAccess()

- \+ fast
- \- a priori choice
- \- 2 parameters to allocate

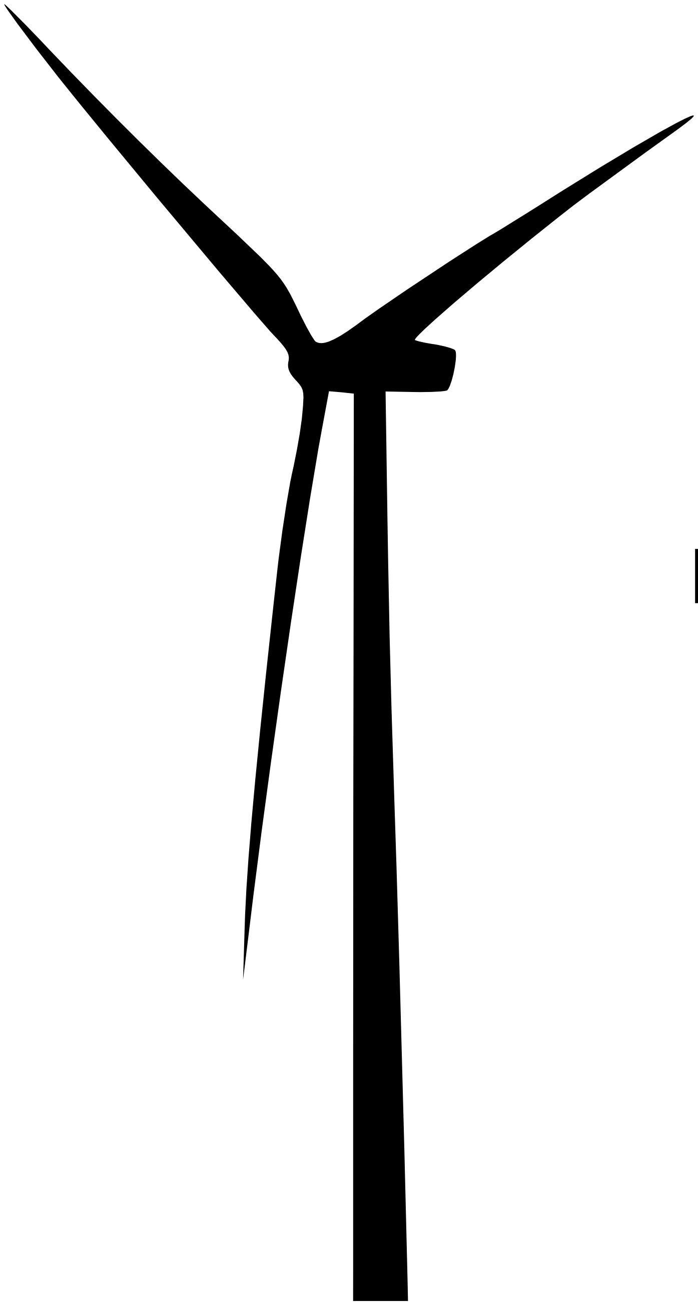
The Threshold-based Model (TBM)

return $\mathbf{T}^{TBM}[u,v,p]$ — getDelay()

update $\mathbf{U}_R[u,v,p], \mathbf{U}_B[u,v,p], l^{max}_{u,v,p}$ — registerPath()

check $\mathbf{T}^{TBM}[u,v,q], \mathbf{A}_B[u,v,q] \quad \forall q \geq p$ — hasAccess()

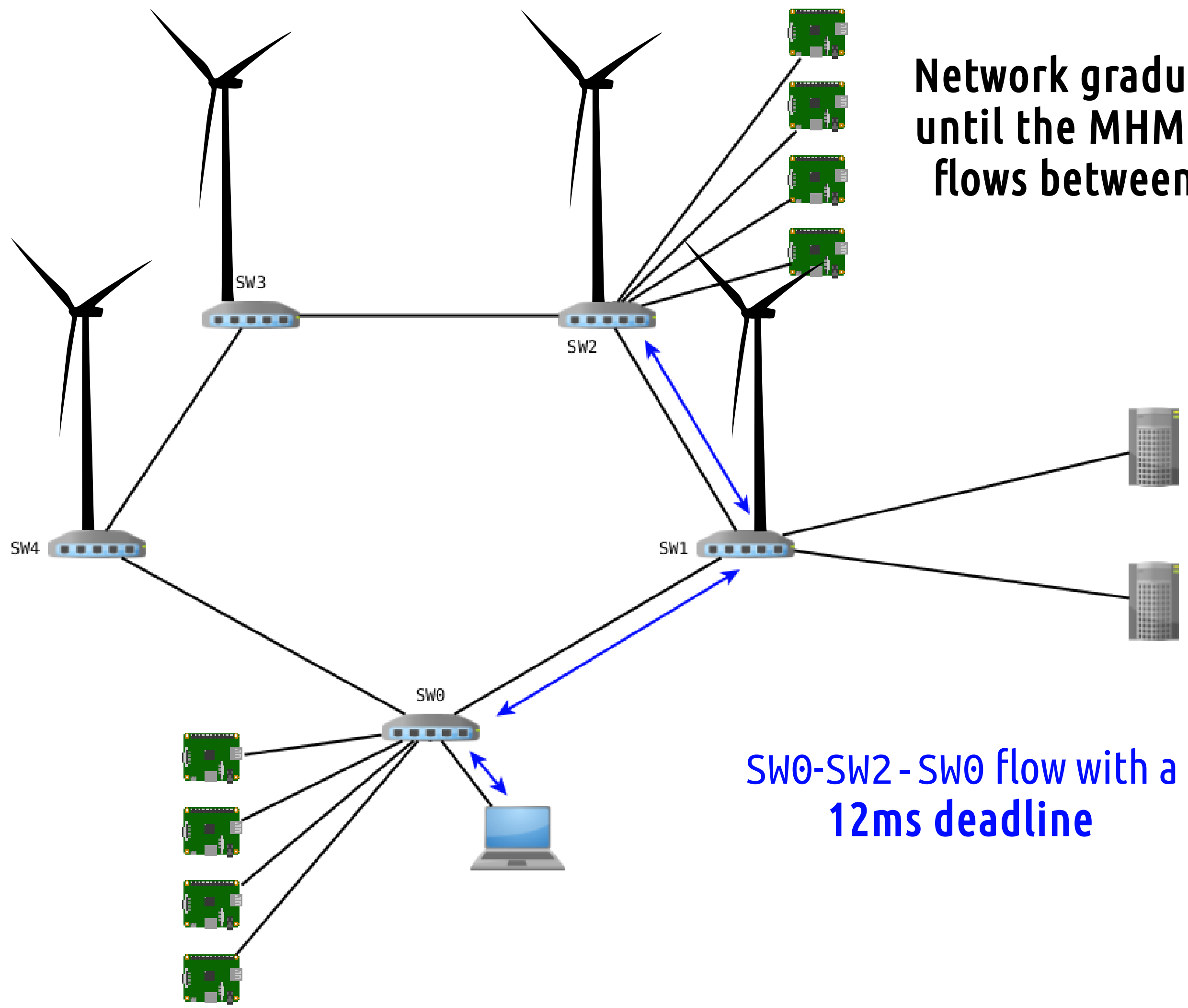
- \+ no a priori choice
- \+ 1 parameter to allocate
- \- blocking problem
- \- slower



Evaluation of the models

Evaluation of the MHM in a **real wind park setup**

Model running on top of OpenDaylight



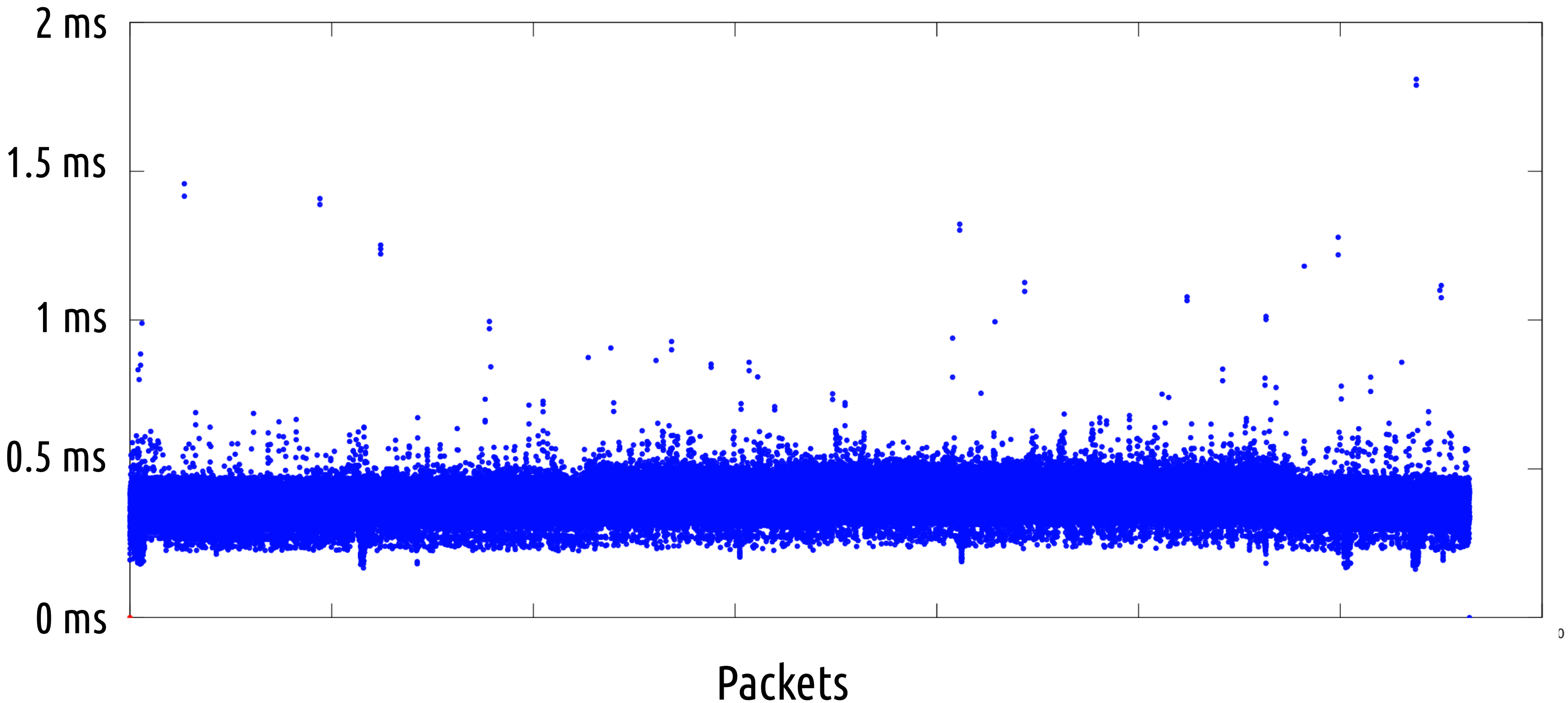
Network gradually congested until the MHM rejects all the flows between SW0 and SW2

SW0-SW2-SW0 flow with a **12ms deadline**

Evolution of the **packet delay** for the SW0-SW2-SW0 flow

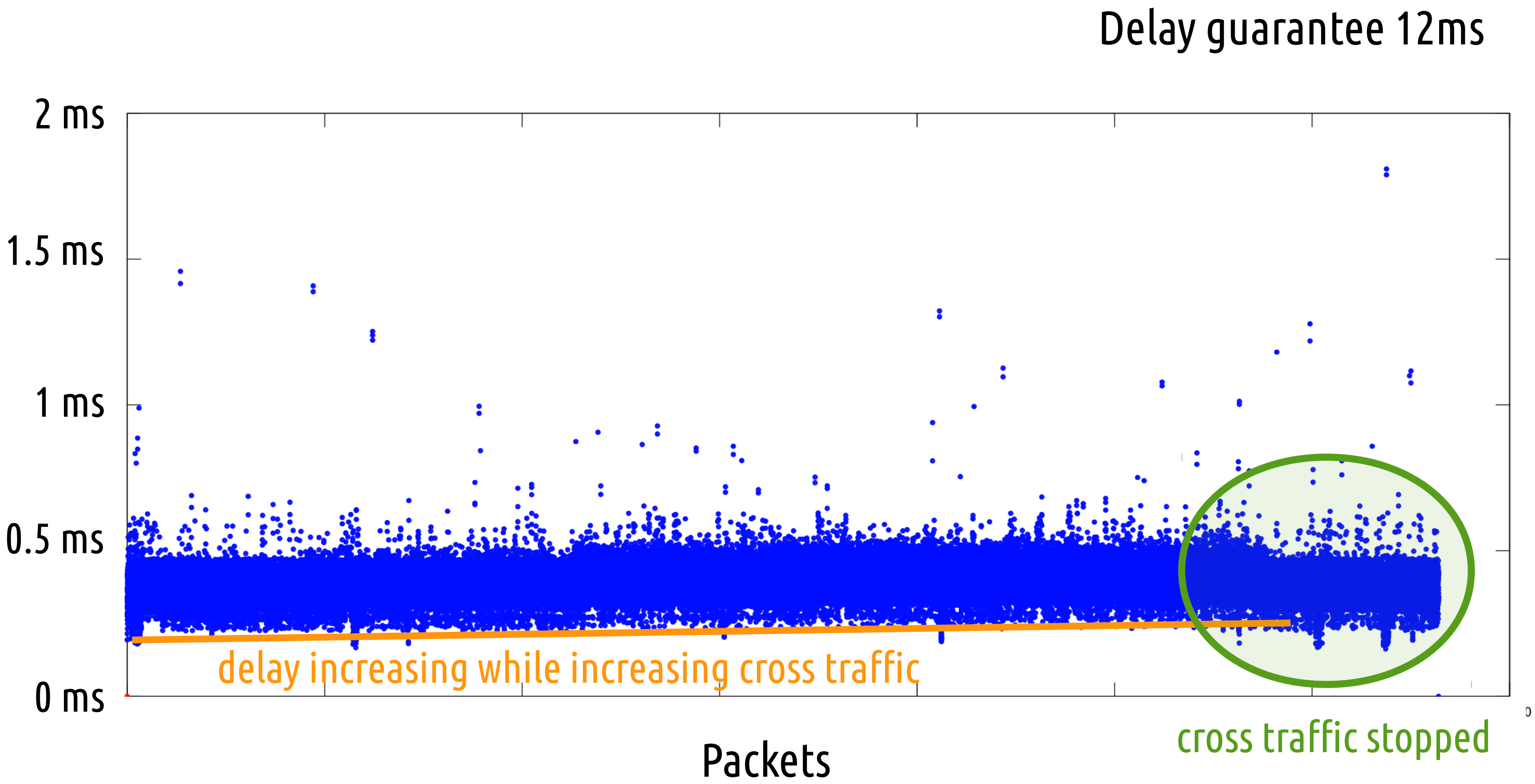
Delay border never violated, no packet loss

Delay guarantee 12ms



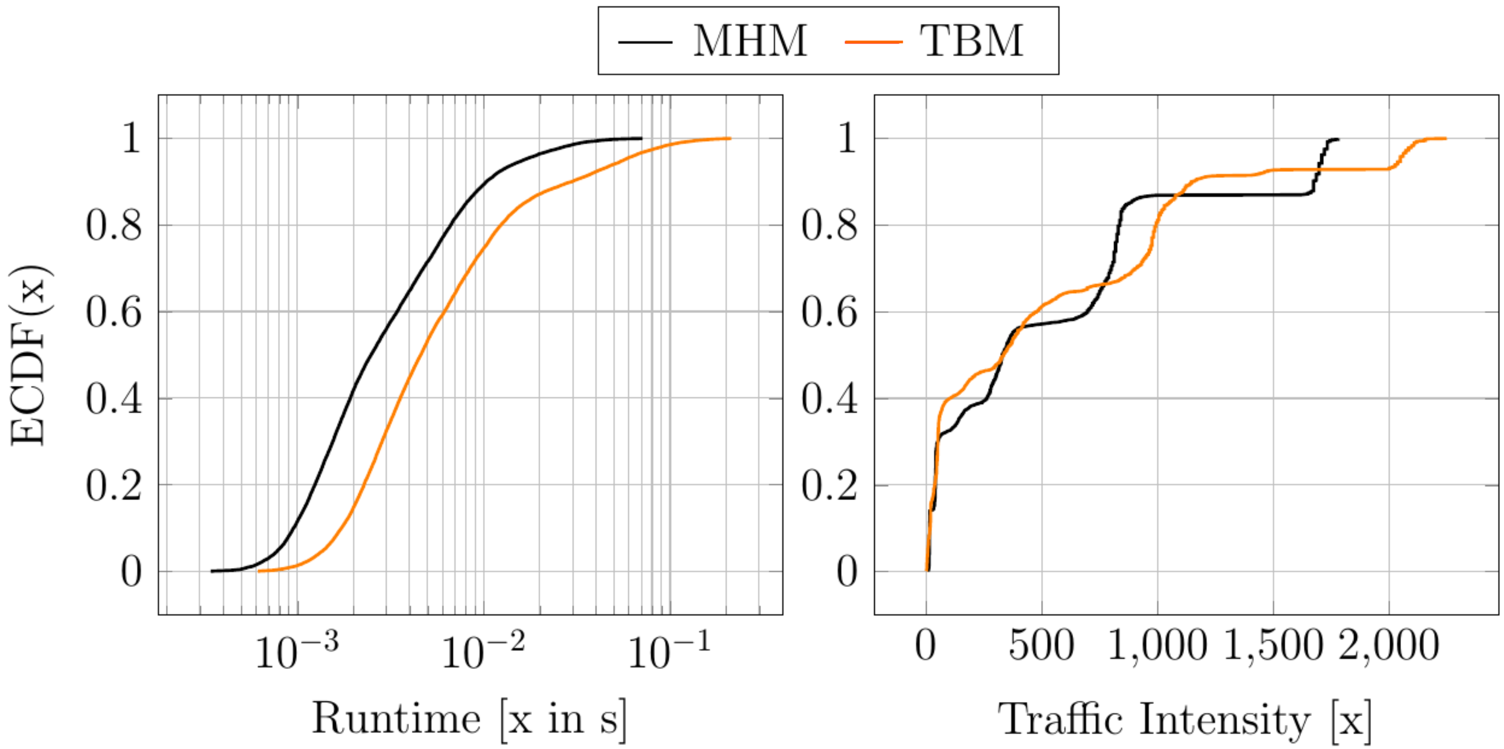
Evolution of the **packet delay** for the SW0-SW2-SW0 flow

Delay border never violated, no packet loss



Simulation of the MHM and TBM

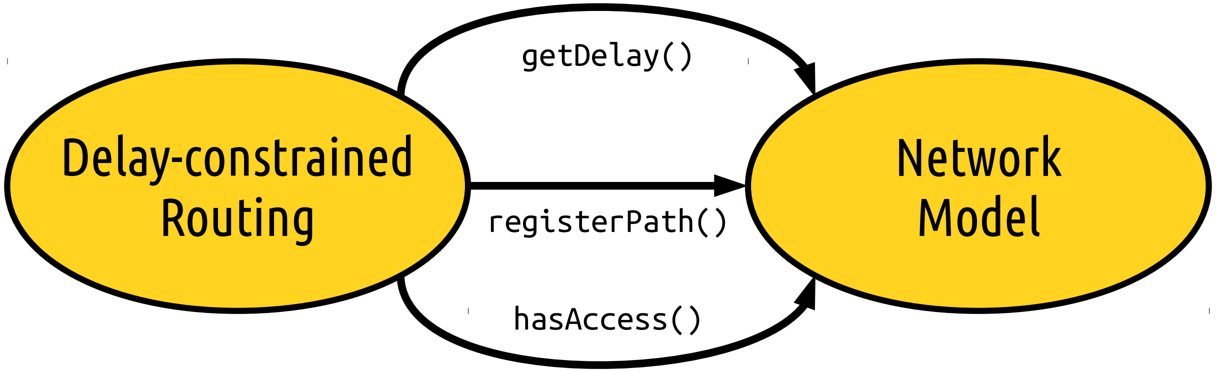
4 queues, various topologies, various routing procedures, various delay constraints



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



[2]

[1] **This presentation!**

388 IEEE COMMUNICATIONS SURVEYS & TUTORIALS, VOL. 20, NO. 1, FIRST QUARTER 2018

Unicast QoS Routing Algorithms for SDN: A Comprehensive Survey and Performance Evaluation

Jochen W. Guck, Amaury Van Bemten, Martin Reisslein, *Fellow, IEEE*, and Wolfgang Kellerer, *Senior Member, IEEE*

***Abstract*—A variety of communication networks, such as industrial communication systems, have to provide strict delay guarantees to the carried flows. Fast and close to optimal quality of service (QoS) routing algorithms, e.g., delay-constrained least-cost (DCLC) routing algorithms, are required for routing flows in such networks with strict delay requirements. The emerging software-defined networking (SDN) paradigm centralizes the network control in SDN controllers that can centrally execute QoS routing algorithms. A wide range of QoS routing algorithms have been proposed in the literature and examined in individual studies. However, a comprehensive evaluation framework and quantitative comparison of QoS routing algorithms that can serve as a basis for selecting and further advancing QoS routing in SDN networks is missing in the literature. This makes it difficult to select the most appropriate QoS routing algorithm for a particular use case, e.g., for SDN controlled industrial communications. We close this gap in the literature by conducting a comprehensive up-to-date survey of centralized QoS routing algorithms. We introduce a novel four-dimensional (4D) evaluation framework for QoS routing algorithms, whereby the 4D correspond to the type of topology, two forms of scalability of a topology, and the tightness of the delay constraint. We implemented 26 selected DCLC algorithms and compared their runtime and cost inefficiency within the 4D evaluation framework. While the main conclusion of this evaluation is that the best algorithm depends on the specific sub-space of the 4D space that is targeted, we identify two algorithms, namely *Lagrange relaxation-based aggregated cost* (LARAC) and *search space reduction delay-cost-constrained routing* (SSR+DCCR), that perform very well in most of the 4D evaluation space.**

***Index Terms*—Delay-constrained least-cost (DCLC) routing, performance evaluation framework, quality of service (QoS), scalability, software-defined networking (SDN).**

Manuscript received March 7, 2017; revised July 24, 2017; accepted August 29, 2017. Date of publication September 7, 2017; date of current version February 26, 2018. This work was supported in part by the European Union's Horizon 2020 Research and Innovation Programme under Grant 671648 (VirtuWind), in part by ERC Grant 647158 (FlexNets), in part by the Alexander von Humboldt Foundation through a Friedrich Wilhelm Bessel Research Award, and in part by the U.S. National Science Foundation under Grant #1716121. *(Corresponding author: Wolfgang Kellerer.)*

J. W. Guck, A. Van Bemten, and W. Kellerer are with the Lehrstuhl für Kommunikationsnetze, Technical University of Munich, 80290 Munich, Germany (e-mail: guck@tum.de; amaury.van-bemten@tum.de;

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 I-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 networked 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 routing algorithms*.

Generally, the route determination (computation) is either carried out in distributed nodes, e.g., the control modules in individual distributed Internet Protocol (IP) routers, or by a centralized controller, e.g., a Software-Defined Networking (SDN) controller [4]–[8]. Distributed routing algorithms had been intensely researched for traditional IP routing, e.g., [9]–[11], and more recently for ad hoc 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 examining 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 [25]–[30]. The pur-

IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT, VOL. 14, NO. 4, DECEMBER 2017 1003

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

Jochen W. Guck, Amaury Van Bemten, and Wolfgang Kellerer, *Senior Member, IEEE*

***Abstract*—Industrial networks require real-time guarantees for the flows they carry. That is, flows have 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 access 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-hop model* (MHM), assigns a rate and a buffer budget to each queue in the network, our second model, the *threshold-based model* (TBM), simply fixes a maximum delay for each queue. Via a packet-level simulation, we confirm that the delay bounds guaranteed by both models are never exceeded and that no packet loss occurs. We further show that the TBM provides more flexibility with respect to the characteristics of the flows to be embedded and that it has the potential of accepting more flows in a given network. Finally, we show that the runtime cost for this increase in flexibility stays reasonable for online request processing in industrial scenarios.**

***Index Terms*—Access control, real-time, industrial network, network modeling, network calculus, quality of service (QoS), software-defined networking (SDN).**

I. Introduction

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 delay bounds 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 runs control functions on a centralized controller which is then able to program the Ethernet 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 SDN forwarding elements that can be changed and updated independently [5], SDN is considered as an inexpensive solution. Therefore, as elaborated in Section II, a plethora of work has been considering the usage of SDN for the provisioning of QoS [6]–[18]. However, the QoS control provided by these approaches is either too *inaccurate* or *slow* for industrial applications [18].

As initiated by Jasperneite *et al.* [19], Guck *et al.* [16]–[18] propose to overcome the two 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. First, network calculus being a deterministic framework, *accurate* bounds can be computed on a per-flow basis. Second, keeping a deterministic model in the centralized control plane allows to avoid the QoS control loop to go through the forwarding plane, thereby allowing to *quickly* provision new flow requests [17]. As such, the two drawbacks of existing approaches are overcome.

C. Contribution: DetServ: Network Models for Deterministic Worst-Case Delay Computation and Access Control

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 determin-

Manuscript received March 19, 2017; revised July 22, 2017 and September 11, 2017; accepted September 19, 2017. Date of publication September 22, 2017; date of current version December 8, 2017. This work was supported by the European Unions Horizon 2020 Research and Innovation Programme under Grant 671648 (VirtuWind) and ERC Grant 647158 (FlexNets). The associate editor coordinating the review of this paper and approving it for publication was F. De Turck. *(Corresponding author: Jochen W. Guck.)*

The authors are with the Lehrstuhl für Kommunikationsnetze, Technical

References

[1] J. W. Guck, A. Van Bemten and W. Kellerer, "DetServ: Network Models for Real-Time QoS Provisioning in SDN-Based Industrial Environments," in IEEE Transactions on Network and Service Management, vol. 14, no. 4, pp. 1003-1017, Dec. 2017.

[2] J. W. Guck, A. Van Bemten, M. Reisslein and W. Kellerer, "Unicast QoS Routing Algorithms for SDN: A Comprehensive Survey and Performance Evaluation," in IEEE Communications Surveys & Tutorials, vol. 20, no. 1, pp. 388-415, Firstquarter 2018.