SMT-based Schedule Synthesis for Time-Sensitive Networks

Silviu S. Craciunas
TTTech Computertechnik AG

Company Key Facts

**TTTech** provides highly reliable and networked electronic systems with solutions based on time-triggered networking technology and modular building blocks for safety controllers.

**Globally** oriented high-tech company, headquartered in Vienna, Austria.

**Innovation** leadership - successful transfer of groundbreaking research to high-volume production.

More than 540 employees with offices in 10 countries (2016).
R&D Funded Projects at a Value of 20 MEUR

- **Aerospace:** Airbus, Boeing, Diehl, Honeywell, Liebherr, Safran, Thales, UTC Aerospace Systems etc.
- **Automotive:** Audi, AVL, Continental, Delphi, Denso, Valeo, Volvo, etc.
- **Industrial:** Alstom, IBM, Sysgo, Thales Austria, etc.
- **Off-Highway:** Palfinger, Schwing, etc.
- **Semiconductors:** ams AG, Infineon, Intel, NXP, ON Semiconductor, etc.

- **EC-funded projects** in ARTEMIS, DREAMS, ENABLE-S3, ECSEL, ITEA 1&2, Eurostars, Greencars, Cleansky, Marie Curie and other R&D Projects directly funded in FP5, FP6, FP7, H2020
- **US programs:** NASA, DARPA, NSF

- **Universities:** Vienna University of Technology, Berkeley University of California, DTU, Chalmers University of Technology, KTH, University of Siegen, University of Kaiserslautern, etc.

- **Research Organizations:** Austrian Institute of Technology, Barcelona Supercomputing Center, CEA, Technalia, Fortiss GmbH, Fraunhofer Society, SRI, TNO, etc.

Strategic R&D of time-triggered communication platforms, prototypes for electronic modules, on-board software and safety platform elements for relevant future application domains.
R&D Funded Projects at a Value of 20 MEUR

R&D Cooperation with Industry

- **Aerospace**: Airbus, Boeing, Diehl, Honeywell, Liebherr, Safran, Thales, UTC Aerospace Systems etc.
- **Automotive**: Audi, AVL, Continental, Delphi, Denso, Valeo, Volvo, etc.
- **Industrial**: Alstom, IBM, Sysgo, Thales Austria, etc.
- **Off-Highway**: Palfinger, Schwing, etc.
- **Semiconductors**: ams AG, Infineon, Intel, NXP, ON Semiconductor, etc.

International Research Network

- **EC-funded projects** in ARTEMIS, DREAMS, ENABLE-S3, ECSEL, ITEA 1&2, Eurostars, Greencars, Cleansky, Marie Curie and other R&D Projects directly funded in FP5, FP6, FP7, H2020
- **US programs**: NASA, DARPA, NSF

R&D Cooperation with Universities & Research Organizations

- **Universities**: Vienna University of Technology, Berkeley University of California, DTU, Chalmers University of Technology, KTH, University of Siegen, University of Kaiserslautern, etc.
- **Research Organizations**: Austrian Institute of Technology, Barcelona Supercomputing Center, CEA, Technalia, Fortiss GmbH, Fraunhofer Society, SRI, TNO, etc.
How will the future look like?

Autonomous & Near Autonomous Operations

$1.9 Trillion

Economic impact of near autonomous cars by 2025

Real-Time Internet of Things

25+ Billion

Embedded and intelligent systems by 2020

Every 2nd

Embedded device will be safety relevant by 2020

Safety & Reliability
Time-sensitive domains
Time-sensitive networking

In the world that Industrial IoT is set to revolutionize, single-purpose control networks using proprietary communication protocols are becoming islands, connected to one another via gateways, and with limited data access and usability. As new functions and machines are added to systems, even more networks are installed, leaving industrial systems potentially containing tens of different networks using incompatible communication protocols. These systems are not flexible or reaching their maximum potential. In the world of IoT and interconnectivity, those who aren't accessing and using valuable data run the risk of being left behind.

Ethernet is the obvious choice for Industrial IoT connectivity. It is open, standard and already used in enterprise networks, making data access much simpler. It is often implemented today in industrial environments for replacing non-critical bus networks or for high bandwidth camera and visualization applications.

Indeed convergence on Ethernet is already common for audio, video, and data services, and Quality of Service (QoS) solutions can be used for critical control over Ethernet. However the determinism provided by QoS does not scale well to the larger converged networks and open infrastructures that are being driven by the Industrial Internet of Things.

The Time-Sensitive Networking (TSN) extension to IEEE 802 Ethernet enables exactly the large scale convergence described in Industrial IoT and Industry 4.0. Low latency and guarantees for communication of even the most critical control traffic means that all applications are able to share a single communications network. When critical and non-critical applications share the same communications infrastructure safely and securely, OT (Operational Technology) and IT (Information Technology) are brought together and data access is improved immeasurably. This will enable new business models, cut downtime, simplify system integration, and reduce the cost of maintenance.

As an example; consider a discrete automation plant with multiple robots working on production lines. Today these robots are controlled locally, with limited synchronization between them, and bottlenecks for data access from beyond the factory floor. Where there is connectivity, it is either done over proprietary networks or via gateways. By removing local control functions or converging non-critical traffic in the same network, one could jeopardize the guarantees for communication of critical messages.

Now consider a TSN connection between these robots. The controls communication is guaranteed across the network even when converged with non-critical traffic, and all robots are synchronized to the same global time. This means that controls networks can be integrated with data networks, and many control functions can be centralized away from the robot cell, where greater computing power can be utilized. Importantly, huge amounts of data from the robots are now also visible to higher layer networks without the need for gateways, enabling Machine as a Service (MaaS) type business models – simultaneously improving service and maintenance from machine builders and lowering capital expenditure for end user manufacturing companies.

Use cases for Deterministic Ethernet standards like TSN can be found in a wide range of application areas beyond manufacturing. For example in wind turbines, deploying control over Ethernet helps to cut downtime and increase production efficiency. And in railway applications, convergence of critical train control networks over Ethernet saves space, weight and power, in addition to improving system reliability.
Time-sensitive networking

- Calculable and guaranteed latencies
- Low and bounded jitter
- Low (ideally 0) packet loss
- Fault tolerance
Experiment

maximum latency = 4s
Experiment

maximum latency = 4s
Experiment

maximum latency = 4s

bob  mike  lea  sam
michelle ana tim alice

$t=0$
Experiment

t = 0

bob

mike

lea

sam

michelle

ana

tim

alice

maximum latency = 4s
Experiment

maximum latency = 4s
Experiment

Maximum latency = 4s

Maximum latency = 4s
Experiment

maximum latency = 4s
Experiment

maximum latency = 4s
Experiment

maximum latency = 4s
Experiment

maximum latency = 4s
Experiment

maximum latency = 4s

t = 2
Experiment

maximum latency = 4s
Experiment

 maximum latency = 4s
Experiment

maximum latency = 4s
Experiment

Maximum latency = 4s
Experiment

Maximum latency = 4s

t = 0

Bob                Mike                Lea                Sam

Michelle          Ana                Tim                Alice
Experiment

maximum latency = 4s
Experiment

maximum latency = 4s
Experiment

maximum latency = 4s

t = 3
Experiment

maximum latency = 4s

t= 4
Experiment

maximum latency = 4s
Experiment

maximum latency = 4s
How?

Assign priorities ➔ Analysis ➔ Reassign priorities

- no
- yes ➔ done

Compositionality?
Alternative
Sending and receiving of frames is done according to a global schedule.
Sending and receiving of frames is done according to a global schedule.

Devices (switches, end systems, etc.) have a common understanding of time.
## Technologies

<table>
<thead>
<tr>
<th>CAN</th>
<th>Profinet</th>
</tr>
</thead>
<tbody>
<tr>
<td>TTP</td>
<td>EtherCAT</td>
</tr>
<tr>
<td>TTEthernet</td>
<td>provides real-time and safety capabilities over Ethernet, in a way that is fully compatible with IEEE 802 Ethernet standards</td>
</tr>
</tbody>
</table>

**TSN**
Priority switch

Port A (ingress) → Switching fabric → Priority filter → Port C (egress)

Port B (ingress) → Switching fabric → Priority filter → Port C (egress)

Priority levels:
- Priority 1
- Priority 6
- Priority 7
- Priority 8
Priority switch

Port A (ingress)

Port B (ingress)

Switching fabric

Priority filter

Priority 8

Priority 7

Priority 6

Priority 1

Port C (egress)
Priority switch

Port A (ingress) -> Switching fabric -> Priority filter

Port B (ingress) -> Switching fabric -> Priority filter

Priority 1 -> Priority 6 -> Priority 7 -> Priority 8

Port C (egress)
Priority switch

Port A (ingress)

Port B (ingress)

Switching fabric

Priority filter

Priority 8

Priority 7

Priority 6

Priority 1

Port C (egress)
Priority switch

Port A (ingress)

Port B (ingress)

Switching fabric

Priority filter

Priority 8

Priority 7

Priority 6

Priority 1

Port C (egress)
Priority switch

Port A (ingress)

Switching fabric

Port B (ingress)

Priority filter

Port C (egress)

Priority 8

Priority 7

Priority 6

Priority 1
Priority switch

Port A (ingress)

Switching fabric

Port B (ingress)

Priority filter

Priority 8

Priority 7

Priority 6

Priority 1

Port C (egress)
TTEthernet switch

Port A (ingress) → Switching fabric → Buffer → Port C (egress)

Port B (ingress) → Switching fabric → Buffer → Port C (egress)

Buffer:
- Priority 1
- Priority 6
- Priority 7
- Priority 8

Switching fabric:
TTEthernet switch

- Port A (ingress)
- Port B (ingress)
- Port C (egress)
- Switching fabric
- Buffer
- Priority 8
- Priority 7
- Priority 6
- Priority 1
TTEthernet switch

Port A (ingress)

Switching fabric

Port B (ingress)

Buffer

Priority 8

Priority 7

Priority 6

Priority 1

Port C (egress)
TTEthernet switch

Port A (ingress)
Switching fabric
Buffer
Priority 8
Priority 7
Priority 6
Priority 1
Port C (egress)

Port B (ingress)
TTEthernet switch

Port A (ingress)

Switching fabric

Buffer

Priority 8

Priority 7

Priority 6

Priority 1

Port C (egress)

Port B (ingress)
TTEthernet switch

Port A (ingress)

Switching fabric

Port B (ingress)

Buffer

Priority 8

Priority 7

Priority 6

Priority 5

Priority 4

Priority 3

Priority 2

Priority 1

Port C (egress)
TTEthernet switch

Port A (ingress)

Port B (ingress)

Switching fabric

Buffer

Priority 8

Priority 7

Priority 6

Priority 1

Port C (egress)
TTEthernet switch

Switching fabric

Port A (ingress)

Buffer

Priority 8

Priority 7

Priority 6

Priority 1

Port C (egress)

Port B (ingress)
TSN switch

Port A (ingress) → Switching fabric → Priority filter → Port C (egress)

Port B (ingress) → Switching fabric → Priority filter → Port C (egress)
TSN switch

Port A (ingress)

Port B (ingress)

Port C (egress)

Switching

Priority filter

Priority 8

Priority 7

Priority 6

Priority 1

Switching
TSN switch

Port A (ingress)

Switching fabric

Priority 8

Priority 7

Priority 6

Priority 1

Port C (egress)

Port B (ingress)
TSN switch

Port A (ingress) -> Switching fabric -> Priority filter -> Priority 8

Port B (ingress) -> Switching fabric -> Priority filter -> Priority 7

Port C (egress) -> Priority filter -> Priority 6

Port C (egress) -> Priority filter -> Priority 1

Port C (egress) -> Priority filter -> Priority 3

Port C (egress) -> Priority filter -> Priority 2

Port C (egress) -> Priority filter -> Priority 4

Port C (egress) -> Priority filter -> Priority 5

Port C (egress) -> Priority filter -> Priority 6

Port C (egress) -> Priority filter -> Priority 7

Port C (egress) -> Priority filter -> Priority 8

Port C (egress)
TSN switch

Port A (ingress)

Port B (ingress)

Switching fabric

Priority filter

Priority 8

Priority 7

Priority 6

Priority 1

Port C (egress)
TSN switch

Port A (ingress)

Switching fabric

Port B (ingress)

Switching fabric

Priority filter

Priority 8

Priority 7

Priority 6

Priority 1

Port C (egress)

T=0
TSN switch

Port A (ingress)

Port B (ingress)

Switching fabric

Priority filter

Priority 8

Priority 7

Priority 6

Priority 1

Port C (egress)

T=0
TSN switch

Port A (ingress) → Switching fabric → Priority filter → Port C (egress)

Port B (ingress) → Switching fabric → Priority filter → Port C (egress)

Port C (egress) -> Priority 1

Priority 8 → Priority 7 → Priority 6

T = 1
TSN switch

Port A (ingress)

Switching fabric

Priority filter

Priority 8

Priority 7

Priority 6

Priority 1

Port C (egress)

Port B (ingress)

T = 1
TSN switch

Port A (ingress)

Port B (ingress)

Switching fabric

Priority filter

Priority 8

Priority 7

Priority 6

Priority 1

Port C (egress)

T = 2
TSN switch

Port A (ingress)

Switching fabric

Priority filter

Priority 8

Priority 7

Priority 6

Priority 1

Port C (egress)

T=2
TSN switch

Port A (ingress)
Switching fabric
Priority filter

Priority 8
Priority 7
Priority 6
Priority 1

Port C (egress)

T=3

Port B (ingress)
TSN switch

Port A (ingress) → Switching fabric → Priority filter → Port C (egress)

- Port B (ingress)
- Priority 8
- Priority 7
- Priority 6
- Priority 1

T = 3
TSN switch

Port A (ingress)

Switching fabric

Priority filter

Priority 8

Priority 7

Priority 6

Priority 1

Port C (egress)

Port B (ingress)

T = 4
TSN switch

Port A (ingress)

Switching fabric

Port B (ingress)

Priority filter

Priority 8

Priority 7

Priority 6

Priority 1

Port C (egress)

T = 5
# Time-Sensitive Networks

IEEE TSN task group - collection of sub-standards that enhance 802 Ethernet with real-time capabilities

<table>
<thead>
<tr>
<th>Standard</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>802.1ASrev</td>
<td>Timing &amp; Synchronization</td>
</tr>
<tr>
<td>802.1Qbv</td>
<td>Enhancements for Scheduled Traffic (Timed Gates for Egress Queues)</td>
</tr>
<tr>
<td>802.1Qbu</td>
<td>Frame Preemption</td>
</tr>
<tr>
<td>802.1Qca</td>
<td>Path Control and Reservation</td>
</tr>
<tr>
<td>802.1Qcc</td>
<td>Central Configuration Management</td>
</tr>
<tr>
<td>802.1Qci</td>
<td>Per-Stream Time-based Ingress Filtering and Policing</td>
</tr>
<tr>
<td>802.1CB</td>
<td>Redundancy, Frame Replication &amp; Elimination</td>
</tr>
</tbody>
</table>
## Time-Sensitive Networks

IEEE TSN task group - collection of sub-standards that enhance 802 Ethernet with real-time capabilities

<table>
<thead>
<tr>
<th>Standard</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>802.1ASrev</td>
<td>Timing &amp; Synchronization</td>
</tr>
<tr>
<td>802.1Qbv</td>
<td>Enhancements for Scheduled Traffic (Timed Gates for Egress Queues)</td>
</tr>
<tr>
<td>802.1Qbu</td>
<td>Frame Preemption</td>
</tr>
<tr>
<td>802.1Qca</td>
<td>Path Control and Reservation</td>
</tr>
<tr>
<td>802.1Qcc</td>
<td>Central Configuration Management</td>
</tr>
<tr>
<td>802.1Qci</td>
<td>Per-Stream Time-based Ingress Filtering and Policing</td>
</tr>
<tr>
<td>802.1CB</td>
<td>Redundancy, Frame Replication &amp; Elimination</td>
</tr>
</tbody>
</table>
# Time-Sensitive Networks

IEEE TSN task group - collection of sub-standards that enhance 802 Ethernet with real-time capabilities

<table>
<thead>
<tr>
<th>Standard</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>802.1ASrev</td>
<td>Timing &amp; Synchronization</td>
</tr>
<tr>
<td>802.1Qbv</td>
<td>Enhancements for Scheduled Traffic (Timed Gates for Egress Queues)</td>
</tr>
<tr>
<td>802.1Qbu</td>
<td>Frame Preemption</td>
</tr>
<tr>
<td>802.1Qca</td>
<td>Path Control and Reservation</td>
</tr>
<tr>
<td>802.1Qcc</td>
<td>Central Configuration Management</td>
</tr>
<tr>
<td>802.1Qci</td>
<td>Per-Stream Time-based Ingress Filtering and Policing</td>
</tr>
<tr>
<td>802.1CB</td>
<td>Redundancy, Frame Replication &amp; Elimination</td>
</tr>
</tbody>
</table>
Network & traffic model

- multi-hop layer 2 switched network via full-duplex multi-speed links
- (multicast) TSN streams with multiple frames per stream
- synchronised time (<1 usec precision)
- wire and device delays

- Scheduled 802.1 Qbv-compatible devices (Sw + Es)
- Scheduled (mutually exclusive) & priority queues
- Guaranteed delivery of critical traffic with known latency, small & bounded jitter
Functional parameters

\[ \langle G(E), G(Q) \rangle \]

Device capabilities

\[ G(E) \]

- \( V_e \) \quad Scheduled Es
- \( V_s \) \quad Scheduled Sw
- \( V_{e+s} \) \quad Scheduled Es+Sw

Queue configuration

\[ G(Q) = \langle N, N_{tr}, N_{prio} \rangle \]
Functional parameters

\[ \langle G(E), G(Q) \rangle \]

Device capabilities

\[ G(E) \]

- \( V_e \): Scheduled Es
- \( V_s \): Scheduled Sw
- \( V_{e+s} \): Scheduled Es+Sw

Queue configuration

\[ G(Q) = \langle N, N_{tt}, N_{prio} \rangle \]
Functional parameters

\[ \langle G(E), G(Q) \rangle \]

Device capabilities

\[ G(E) = \langle V_e, V_s, V_{e+s} \rangle \]

- \( V_e \): Scheduled Es
- \( V_s \): Scheduled Sw
- \( V_{e+s} \): Scheduled Es+Sw

Queue configuration

\[ G(Q) = \langle N, N_{tt}, N_{prio} \rangle \]

- \( N_{tt} \geq 1 \)
Functional parameters

\[ G(E), G(Q) \]

Device capabilities

\[ G(E) \]

- \( V_e \) - Scheduled Es
- \( V_s \) - Scheduled Sw
- \( V_{e+s} \) - Scheduled Es+Sw

Queue configuration

\[ G(Q) = \langle N, N_{tt}, N_{prio} \rangle \]

- \( N_{tt} \geq 1 \)

• Critical traffic assigned to the scheduled queues
• Non-critical traffic assigned to priority queues (post-analysis through network calculus [Frances@ERTS06])
• Isolation: non-critical streams may interfere with each other in priority queues, but not with critical streams (isolated in the scheduled queues)
### 802.1Qbv configurations

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>{V_{e+s}, \langle 1</td>
<td>1</td>
</tr>
<tr>
<td>{V_{e+s}, \langle n</td>
<td>1</td>
</tr>
<tr>
<td>{V_{e+s}, \langle n</td>
<td>n</td>
</tr>
<tr>
<td>{V_{e+s}, \langle n</td>
<td>m</td>
</tr>
<tr>
<td>{V_{e+s}, \langle n</td>
<td>0</td>
</tr>
</tbody>
</table>
Deterministic Ethernet Constraints

Frame

\[ \phi \]

\begin{align*}
\text{Period} & : 0 \rightarrow 8 \\
\end{align*}

Stream

\begin{align*}
a & \rightarrow d \\
\text{Period} & : 0 \rightarrow 8 \\
\end{align*}

Link

\begin{align*}
(1, 4) & \quad (1, 6) \\
\text{Period} & : 0 \rightarrow 12 \\
\end{align*}

see also [Steiner@RTSS10] or [Craciunas@RTNS14]
Stream and e2e latency constraints

Link 1

Link 2

Link 3

max. allowed e2e latency

δ

δ

12

see also [Steiner@RTSS10] or [Graciunas@RTNS14]
Queue Interleaving
Queue Interleaving
Queue Interleaving

In order to maintain jitter and latency requirements we expect at each device a certain timely order of frames.
In order to maintain jitter and latency requirements we expect at each device a certain timely order of frames.
Queue Interleaving
Queue Interleaving
Queue Interleaving

expected
Queue Interleaving

expected
Queue Interleaving

expected
Queue Interleaving

- synchronization errors, frame loss, time-based ingress policing (e.g. IEEE 802.1Qci) may lead to non-deterministic placement in queues during runtime
- timed gates control events on the egress port, not the order of frames in the queue
- placing of frames in the scheduled queues at runtime may be non-deterministic

Timely behaviour of streams may oscillate, accumulating jitter for the overall end-to-end transmission
Queue Isolation
Queue Isolation
Queue Isolation
Queue Isolation
Queue Isolation

expected
Queue Isolation

Solves the non-determinism problem but reduces the solution space
Stream (Flow) isolation
Stream (Flow) isolation
Stream (Flow) isolation
Stream (Flow) isolation
Stream (Flow) isolation

expected
Stream (Flow) isolation

expected
Stream (Flow) isolation
Stream (Flow) isolation

- Once a flow has arrived, no other flow can arrive in the same queue until the first flow has been completely sent
- Better than queue isolation but still restrictive
Stream (Flow) isolation

- Once a flow has arrived, no other flow can arrive in the same queue until the first flow has been completely sent
- Better than queue isolation but still restrictive
Stream (Flow) isolation

- Once a flow has arrived, no other flow can arrive in the same queue until the first flow has been completely sent
- Better than queue isolation but still restrictive
Frame isolation
Frame isolation
Frame isolation
Frame isolation

expected
Frame isolation

expected
Frame isolation
Frame isolation
Frame isolation

expected
Frame isolation
Frame isolation

expected
Frame isolation

expected
Frame isolation

- Ensure that there are only frames of one flow in the queue at a time
- Frames from another flow may only enter the queue if the already queued frames of the initial flow have been serviced
- Less performant than stream isolation since the solver has to consider at all frame interleavings
The constraint for minimum jitter scheduling of critical traffic for 802.1Qbv networks is:

- isolate framesstreams in the **time domain**
- OR
- isolate streams in **different queues**
Scheduling problem
Scheduling problem

Find **offsets** and **queue assignments** for individual frames of TSN streams along the route that conform to the constraints.
Scheduling problem

Find **offsets** and **queue assignments** for individual frames of TSN streams along the route that conform to the constraints

Reduces to finding a solution for a set of inequalities resulting from

- frame constraints
- link constraints
- stream constraints
- end-to-end latency constraints
- stream or frame isolation constraints

802.1 Qbv
Scheduling problem

Find offsets and queue assignments for individual frames of TSN streams along the route that conform to the constraints.

Reduces to finding a solution for a set of inequalities resulting from:
- frame constraints
- link constraints
- stream constraints
- end-to-end latency constraints
- stream or frame isolation constraints

NP-complete
Satisfiability Modulo Theories

satisfiability of logical formulas in first-order formulation

background theories \( \mathcal{LA}(\mathbb{Z}) \ BV \)

variables \( x_1, x_2, \ldots, x_n \)

logical symbols \( \lor, \land, \neg, (, ) \)

non-logical symbols \( +, =, \%, \leq \)

quantifiers \( \exists, \forall \)

optimization (OMT) [Bjørner@TACAS15]

A lot of solvers and a very active community

OpenSMT [Bruttomesso@TACAS10] Yices [Dutertre@CAV14]

CVC4 [Barrett@CAV11] Z3 [de Moura@TACAS08]
Satisfiability Modulo Theories

satisfiability of logical formulas in first-order formulation
background theories $\mathcal{L}(\mathbb{Z}) \mathcal{BV}$
variables $x_1, x_2, \ldots, x_n$
logical symbols $\lor, \land, \neg, (, )$
non-logical symbols $+, =, \%, \leq$
quantifiers $\exists, \forall$

optimization (OMT) [Bjørner@TACAS15]

A lot of solvers and a very active community

OpenSMT [Bruttomesso@TACAS10]  Yices [Dutertre@CAV14]
CVC4 [Barrett@CAV11]  Z3 [de Moura@TACAS08]
Optimization

Optimize schedule with respect to certain properties of the system (e.g. minimize end-to-end latency of selected streams)

802.1 Qbv-specific optimizations:

- **QoS properties**: minimize required scheduled queues in order to increase QoS properties of non-critical traffic
- **Design space exploration** in case of infeasible use-cases, i.e. find the minimal number of queues required for scheduled traffic such that a schedule is found

Many more optimization opportunities in combination with other TSN sub-standards (e.g. frame preemption)
Experiments

- **Z3** v4.4.1 solver (64bit) (Yices v2.4.2 with quantifier-free linear integer arithmetic)
- 64bit 4-core **3.40GHz** Intel Core-i7 PC with 4GB memory
- 3 predefined topologies ranging from 3 end-systems connected to one switch to 7 end-systems connected through 5 switches via **1Gbit/s** links with a **1usec** macrotick granularity (generate high utilization on the links)
- Time-out value for a run to **5 hours**
- System configuration: \( \{ V_{e+s}, \langle 8,8,0 \rangle \} \)

Scalability and schedulability experiments
Evaluation
Evaluation

time
Evaluation

time
Evaluation

time

?
Evaluation

time

- hyperperiod
- link/CPU utilization
- topology
- periods
- macrotick
- size of network
Scalability Experiments

- **Frame isolation** method (using an incremental backtracking algorithm with step size of 1)
- Vary the problem set in 3 dimensions:
  1. topology size,
  2. number of flows,
  3. flow periods (chosen randomly from 3 sets of predefined periods)
- Data size uniformly between 2 and 8 MTU-sized frames
- Senders and receivers are chosen randomly
Scalability Experiments

Number of flows (Average number of frame instances)

P1={10, 20}[ms]
P2={10, 25, 50, 100}[ms]
P3={5, 10, 200, 500}[ms]
Scalability Experiments

Number of flows (Average number of frame instances)

- P1={10, 20}[ms]
- P2={10, 25, 50, 100}[ms]
- P3={5, 10, 200, 500}[ms]
Scalability Experiments

Number of flows (Average number of frame instances)

- P1 = {10, 20}[ms]
- P2 = {10, 25, 50, 100}[ms]
- P3 = {5, 10, 200, 500}[ms]
Frame vs. Stream Isolation

- 381 randomly generated test cases with up to 1000 streams
- 17 reached the time-out
- Stream isolation was on average 13% faster with a median of 8.03%
- 36.7h for stream isolation and 59h for frame isolation - 30.73% improvement

M1 runtime as percentage of M2 runtime
Runtime reduction through M1

M1 over M2 reduction [%]
0 20 40 60 80 100

M1 runtime as percentage of M2 runtime

simple/P1/5
simple/P2/5
simple/P3/5
simple/P1/15
simple/P2/15
simple/P3/15
simple/P1/25
simple/P2/25
simple/P3/25
simple/P1/50
simple/P2/50
simple/P3/50
medium/P1/5
medium/P2/5
medium/P3/5
medium/P1/25
medium/P2/25
medium/P3/25
medium/P1/50
medium/P2/50
medium/P3/50
complex/P1/10
complex/P2/10
complex/P3/10
complex/P1/50
complex/P2/50
complex/P3/50
complex/P1/100
complex/P2/100
complex/P3/100
Schedulability Experiments

- Generated inputs that force streams to **interleave** if scheduled in the same egress queue
- Runs **w/ and w/o optimization** objectives using both stream and frame isolation methods
- Minimize **accrued sum** of the number of **queues** used per egress port
- No incremental steps for optimization runs
Schedulability Experiments

Flow isolation  Optimized Flow isolation  Frame isolation  Optimized Frame isolation

runtime

queues

10 min
1 min
10 sec
1 sec
100 ms

Flow isolation  Frame isolation  

T01  T03  T04  T05  T06  T07  T08  T09  T10  T11  T12  T13  T14  T15  T16  T17  T18  T19  T20
Heuristics

For large networks we have to use heuristics, e.g:

Greedy Randomized Adaptive Search Procedure (GRASP)-based metaheuristic together with M. L. Raagaard and P. Pop (c.f. [2])

<table>
<thead>
<tr>
<th>ID</th>
<th>running time (s)</th>
<th>queue usage</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ILP</td>
<td>OMT</td>
</tr>
<tr>
<td>T01</td>
<td>0.66</td>
<td>0.81</td>
</tr>
<tr>
<td>T04</td>
<td>2.49</td>
<td>2.46</td>
</tr>
<tr>
<td>T05</td>
<td>3.73</td>
<td>3.43</td>
</tr>
<tr>
<td>T10</td>
<td>4.70</td>
<td>5.12</td>
</tr>
<tr>
<td>T11</td>
<td>16.54</td>
<td>12.94</td>
</tr>
<tr>
<td>T12</td>
<td>210.03</td>
<td>34.33</td>
</tr>
<tr>
<td>T14</td>
<td>39.06</td>
<td>22.87</td>
</tr>
<tr>
<td>T18</td>
<td>10.98</td>
<td>7.17</td>
</tr>
</tbody>
</table>

Table 2: Comparison of ILP, OMT, and GRASP
Conclusions

Scheduling problem arising from the IEEE 802.1Qbv extension on multi-hop fully switched TSN networks
Conclusions

Scheduling problem arising from the IEEE 802.1Qbv extension on multi-hop fully switched TSN networks

• key functional parameters affecting the behaviour of 802.1Qbv networks
Conclusions

Scheduling problem arising from the IEEE 802.1Qbv extension on multi-hop fully switched TSN networks

- **key functional parameters** affecting the behaviour of 802.1Qbv networks
- **802.1Qbv-specific constraints** for creating correct offline schedules guaranteeing **low and bounded jitter** as well as **deterministic end-to-end latencies** for critical traffic
Conclusions

Scheduling problem arising from the IEEE 802.1Qbv extension on multi-hop fully switched TSN networks

• **key functional parameters** affecting the behaviour of 802.1Qbv networks

• **802.1Qbv-specific constraints** for creating correct offline schedules guaranteeing **low and bounded jitter** as well as **deterministic end-to-end latencies** for critical traffic

• use **SMT/OMT** solvers to create static schedules for 802.1Qbv devices
Conclusions

Scheduling problem arising from the IEEE 802.1Qbv extension on multi-hop fully switched TSN networks

- **key functional parameters** affecting the behaviour of 802.1Qbv networks

- **802.1Qbv-specific constraints** for creating correct offline schedules guaranteeing *low and bounded jitter* as well as *deterministic end-to-end latencies* for critical traffic

- use **SMT/OMT** solvers to create static schedules for 802.1Qbv devices

- **optimization** directions & system **configurations** and their **trade-offs**
Conclusions

Scheduling problem arising from the IEEE 802.1Qbv extension on multi-hop fully switched TSN networks

- **key functional parameters** affecting the behaviour of 802.1Qbv networks
- **802.1Qbv-specific constraints** for creating correct offline schedules guaranteeing **low and bounded jitter** as well as **deterministic end-to-end latencies** for critical traffic
- use **SMT/OMT** solvers to create static schedules for 802.1Qbv devices
- **optimization** directions & system **configurations** and their **trade-offs**
- **evaluation** in terms of scalability and schedulability
References and further reading


IEEE 802.1 Time Sensitive Networking (TSN) task group - http://www.ieee802.org/1/pages/tns.html
Thank you!