TSNsched: Automated Schedule Generation for Time Sensitive Networking

Aellison Cassimiro T. dos Santos*, Ben Schneider*, Vivek Nigam*
* Universidade Federal da Paraíba, `fortiss

FMCAD 2019 · Oct 23, 2019
1. TSN: Time-Sensitive Networking
2. The Scheduling Problem
3. TSNsched
4. Experimental Evaluation
TSN: TIME-SENSITIVE NETWORKING

Determinism, real-time and reliability

A set of standards in development

Different scheduling mechanisms available

Some fields of application
- Industry automation
- Vehicular applications
- Voltage Sampling
System block diagram of a vehicle’s communication module.

TIME-AWARE SHAPER (IEEE 802.1Qbv)

Uses time scheduling to control the egress queue gates

Time-aware queue-draining procedures

VLAN tag encoded priority values
TSN Switch

TAS mechanisms in a switch.
Concept of a cycle.

Representation of cycles in a timeline.
THE SCHEDULING PROBLEM

Unknown core variables of the problem
○ Priorities of the flows
○ Closing and opening time of the gates
○ Cycle start and duration

The problem is NP-Complete

Priority per switch and V-LAN retagging (IEEE 802.1Qci)
PROBLEM EXAMPLE

Topology with 3 publishers and 3 subscribers.
Example of cycle for the proposed topology.
TSNsched

**Automatic Schedule Generation**
- Topology (switches, flows) as input
- Priority, timing variables and cycles information as output
- Java and Z3

**High flexibility and expressiveness**

**Stand-alone and library version**
TSNsched KEY FEATURES

Jitter and latency constraints

Flexibility of the constraints

Multicast flows

Allows for rapid prototyping

Supports convergent networks
Schedule generation process.

- Automatic Execution
- Manual Intervention

Network Requisites

Generating Input

User Input

Generating Schedule

Setting Scheduling Rules

Setting Flow Fragments

Generating Output

Tool Logs

Switches’ GCLs
Simple Unicast Flow: $F_1$

A flow broken into fragments.
Path to dev30: dev13, switch2(flow1Fragment1), switch6(flow1Fragment2), dev30

switch2: Cycle start: 2000.0
Cycle duration: 1991.0
Priorities used -
Port name: switch2Port5
Connects to: switch6
Fragments: flow1Fragment1,
Priority number: 1
Slot start: 0.0
Slot duration: 50.0

switch6: Cycle start: 413.0
Cycle duration: 400.0
Priorities used -
Port name: switch6Port9
Connects to: dev30
Fragments: flow1Fragment2,
Priority number: 1
Slot start: 1.0
Slot duration: 50.0
TSNsched VARIABLES AND CONSTRAINTS

Constraints are applied at port and flow level (25 constraints)

Main variables of the problem compose the cycles and flow fragments
- Each fragment is composed by its priority and 3 timing variables per scheduled packet
- Each cycle is composed by its starting time, duration and a gate opening and closing time per priority

+ packets per fragment, priorities per cycle, switches in path ⇒ + size of the problem
+ fragments per port ⇒ + complex to solve

1. Set of basic constraints (11)

2. Cycle and time slot constraints (9)

3. Core packet timing constraints (5)
\forall S \in SW.
\forall p \in S.
\forall ff_1, ff_2 \in p.PFF.

ff_1.prt = ff_2.prt \implies
p.c.SD(ff_1.prt) = p.c.SS(ff_2.prt) \land
p.c.SD(ff_1.prt) = p.c.SD(ff_2.prt)

Constraint 14: Same Priority, Same Slot
CONSTRAINT EXAMPLE

\[ \forall S \in \text{SW}. \]
\[ \forall p \in S. \]
\[ \text{best} E \geq 1 - \frac{\sum_{i=1}^{8} p.c.SD(i)}{p.c.d} \]

Constraint 20: Best-Effort Bandwidth Reservation
\[ \forall S \in \text{SW}. \forall p \in S. \forall ff \in p.\text{PFF}; \quad i, j \in \mathbb{Z}; \quad 0 < i \leq ff.\text{T.size}; \quad 0 < j \leq \text{numCycles}. \]

\[ \text{ff.}T(i).\text{st} \geq \text{p.c.s} + \text{p.c.d} \times (j - 1) + \text{p.c.SS}(\text{ff.prt}) + \text{p.transT} \land \]

\[ \text{ff.}T(i).\text{st} \leq \text{p.c.s} + \text{p.c.d} \times (j - 1) + \text{p.c.SS}(\text{ff.prt}) + \text{p.c.SD}(\text{ff.prt}) \]

**Constraint 23: Transmission Inside a Time Slot**
\[ \forall s \in SW. \]
\[ \forall p \in S. \]
\[ \forall ff_1, ff_2 \in p.\text{PFF}; k, l \in Z; ff_1 \neq ff_2; 0 < k \leq ff_1.\text{T.size}; 0 < l \leq ff_2.\text{T.size}. \]
\[ ff_1.\text{prt} = ff_2.\text{prt} \implies ff_1.\text{T}(k).\text{st} \leq ff_2.\text{T}(l).\text{at} \lor ff_1.\text{T}(k).\text{at} \geq ff_2.\text{T}(l).\text{st} \]

**Constraint 24: Frame Isolation**
Constraint 25: Send As Soon As Possible

A. Send After Another Packet
B. Arrived Before Slot Start
C. Arrived Inside Slot
D. Arrived After Slot End

∀ff₁, ff₂ ∈ p.PFF.

SendAfterAnotherPacket

(ArrivedBeforeSlotStart ∧ ArrivedInsideSlot ∧ ArrivedAfterSlotEnd)
EXPERIMENTAL EVALUATION

Use a topology generator to create network topologies

Scenarios match in size with real-life use cases
○ VDMA R+A demonstrator: 26 nodes and 28 unicast flows

Maximum latency of 1000μs and jitter of 25μs
# SUMMARY OF RESULTS

## Normal Performance Flows (Periodicity = 2000μs)

<table>
<thead>
<tr>
<th>No of Flows</th>
<th>1</th>
<th>3</th>
<th>5</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Small Only</td>
<td>0.68μs/509.55μs (4)</td>
<td>4.13μs/689.69μs (12)</td>
<td>4.54μs/631.54μs (22)</td>
<td>3.01μs/840.77μs (44)</td>
</tr>
<tr>
<td>Medium Only</td>
<td>1.38μs/887.18μs (9)</td>
<td>3.18μs/891.93μs (28)</td>
<td>4μs/653.78μs (46)</td>
<td>TO (93)</td>
</tr>
<tr>
<td>Large Only</td>
<td>2.75μs/972.25μs (14)</td>
<td>4.85μs/621.16μs (44)</td>
<td>3.64μs/755.73μs (73)</td>
<td>TO (147)</td>
</tr>
</tbody>
</table>

## High Performance Flows (Periodicity = 1000μs)

<table>
<thead>
<tr>
<th>No of Flows</th>
<th>1</th>
<th>3</th>
<th>5</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Small Only</td>
<td>2.51μs/997.07μs (4)</td>
<td>1.79μs/903.69μs (13)</td>
<td>2.83μs/751.49μs (21)</td>
<td>1.95μs/946.43μs (45)</td>
</tr>
<tr>
<td>Medium Only</td>
<td>3.55μs/957.18μs (10)</td>
<td>2.81μs/797.70μs (28)</td>
<td>4.75μs/807.82μs (46)</td>
<td>TO (91)</td>
</tr>
<tr>
<td>Large Only</td>
<td>1.07μs/993.55μs (14)</td>
<td>2.64μs/860.32μs (44)</td>
<td>3.02μs/936.40μs (69)</td>
<td>NA</td>
</tr>
</tbody>
</table>
EXECUTION TIME ANALYSIS
CONSIDERATIONS ABOUT THE RESULTS

All scenarios successfully executed complied with the requirements

Overall jitter was surprisingly low

Update results show progress
FUTURE WORKS

Software-defined application periods

Integration of TSNsched with in model-based framework for automation systems, such as 4diac

New time-efficient scheduling approaches

General performance improvements
We thank Tiziano Murano and Anand Subramanian for fruitful discussions.