An SMT-based Tool for Automatic Schedule Generation for Time Sensitive Networking

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

TSN Technology

The composite of standards titled Time-Sensitive Networking (TSN), was created with the purpose of offering reliability and determinism to standard IEEE 802.1 Ethernet networks, allowing its usage in time-critical systems.

Core features and information:
- Determinism, real-time and reliability
- Standards still in development
- Different scheduling mechanisms (*e.g.* Time Aware Shaper)

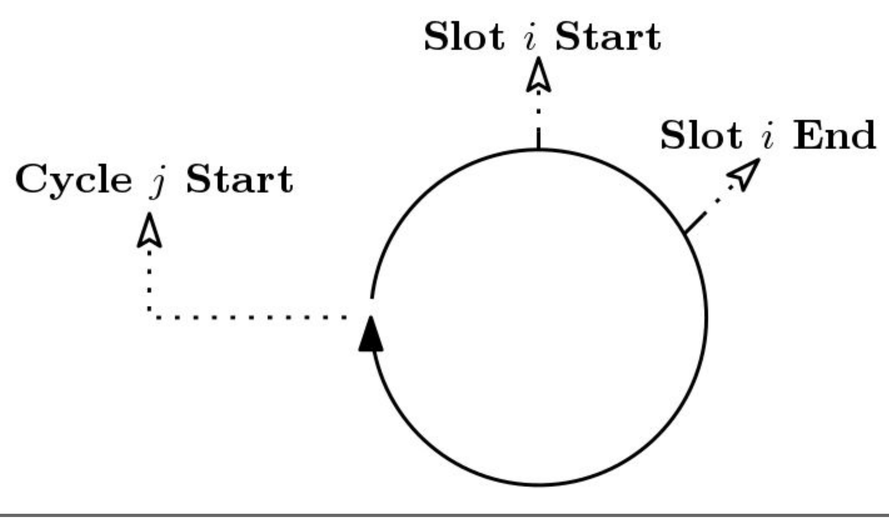
Fields of application:
- Industry automation
- Vehicular applications
- Voltage Sampling
- ...

The Scheduling Problem

From the standards used to implement the TSN technology, the one which builds the basis of the time-triggered communication paradigm is IEEE 802.1Qbv, responsible for the management of queues and traffic in switches present on TSN networks.

Basis of the scheduling problem:
- Uses time scheduling to control the egress queue gates
- VLAN tag encoded priority values
- Priority values assign packets to queues
- Gates stay open in fixed time space of a cycle (called time slots, or slots, for short)
- Is NP-complete

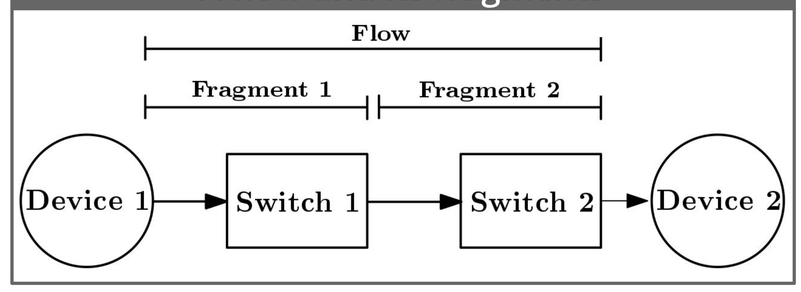
Concept of a cycle



Scheduling Approach

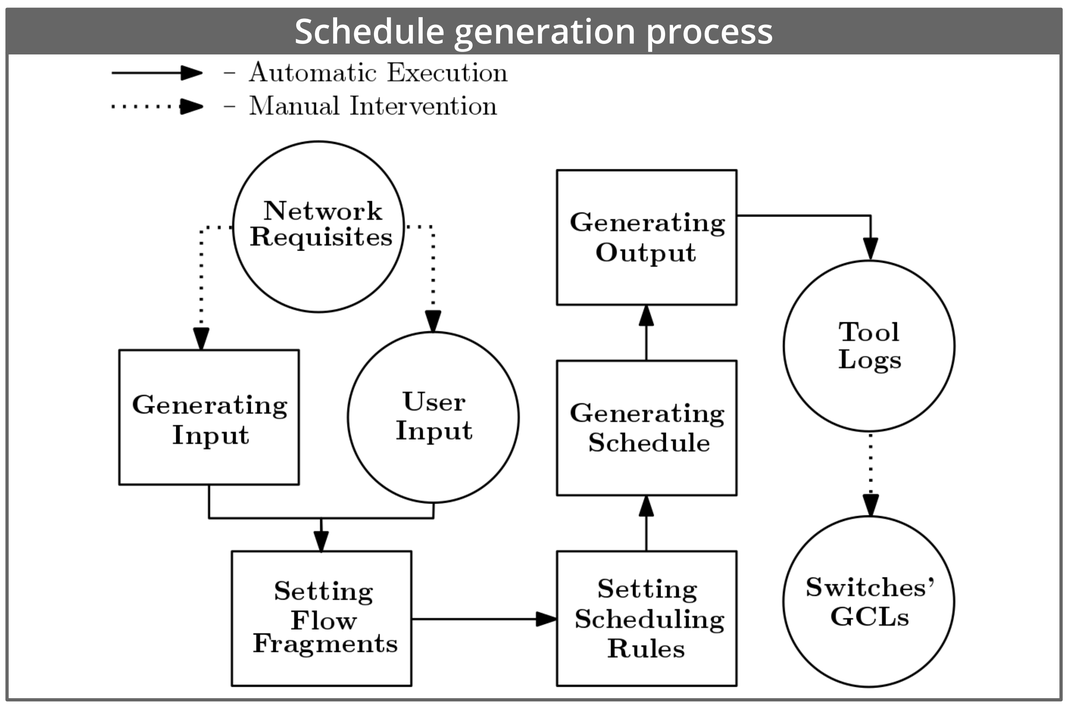
Basis of the scheduling approach:
- Break flows into fragments
- Fragments assigned to ports
- Rules established per port
- Flexible and expressive
- Modular network nodes
- Convergent networks

A flow and its fragments



On the schedule generation process:
- Network is modeled as input
- With the fragments, incognito variables are created
- Generated logs are machine readable

Schedule generation process



Constraints specification:
- Z3 as the solver
- Flattened quantifiers
- Suitable abstraction

Packet Departure constraint

$\forall f \in F;\ i \in Z;\ 0 < i \leq f.FF(1).T.size.$

$f.FF(1).T(i).dt = f.\phi + f.\rho \times (i-1)$

No Overlapping Slots constraint

$\forall S \in SW.\forall p \in S.\forall ff_1, ff_2 \in p.PFF.$

$ff_1.prt \neq ff_1.prt \implies$

$p.c.SS(ff_1.prt) \geq p.c.SS(ff_2.prt) + p.c.SD(ff_2.prt) \vee$

$p.c.SS(ff_1.prt) + p.c.SD(ff_2.prt) \leq p.c.SS(ff_2.prt)$

Transmit Inside a Slot constraint

$\forall S \in SW.\forall p \in S.\forall ff \in p.PFF; i, j \in Z; 0 < i \leq ff.T.size; 0 < j \leq numCycles.$

$ff.T(i).st \geq p.c.s + p.c.d \times (j-1) + p.c.SS(ff.prt) + p.transT \wedge$

$ff.T(i).st \leq p.c.s + p.c.d \times (j-1) + p.c.SS(ff.prt) + p.c.SD(ff.prt)$

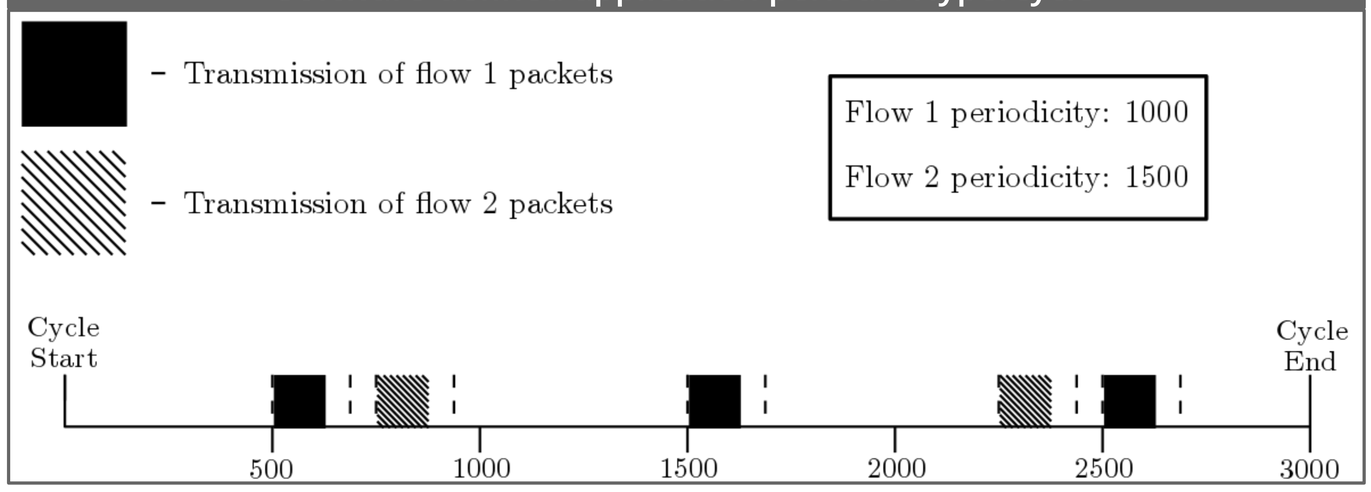
Considerations and Future Work

The modular design created by the usage of flow fragments also provides a simple process of implementation and modification of constraints, resulting in a concise translation from constraints of the formal model on a link level. A consequence of this fact is the ease of implementation of different communication structures such as unicast and multicast flows, aside from the capability of implementation of problem variants, such as the ones explored in works in the state of the art. This flexibility also widens the possibilities for expansion of our work.

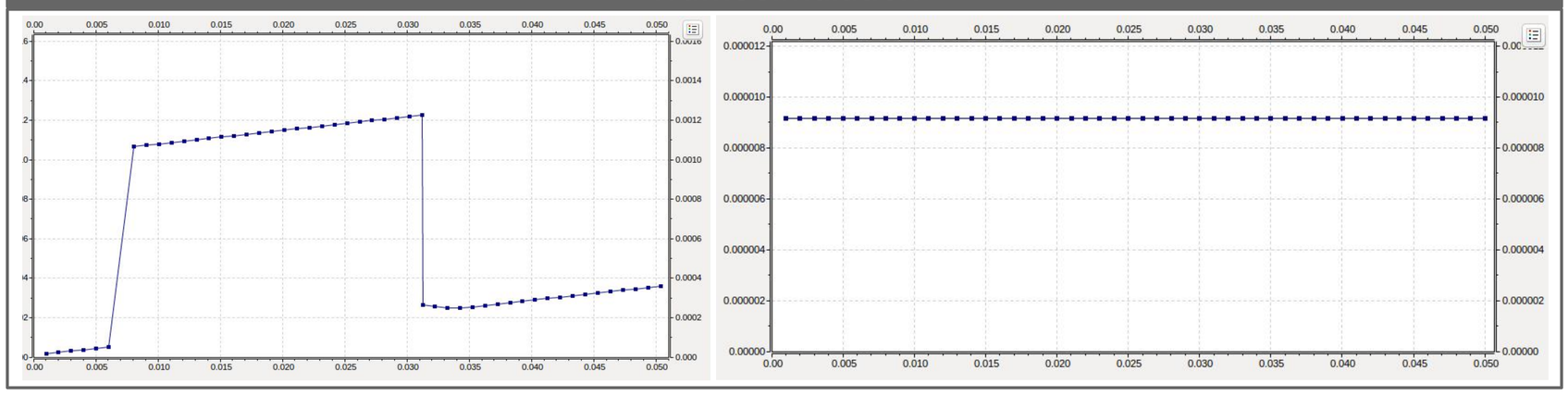
Next steps:
- Integration of TSNsched with 4diac
- Schedule validation and testing
- Implementation of software-defined application periods
- New time-efficient scheduling approaches

Possible expansions:
- Integration of other TSN standards
- Research on symmetry-breaking assertions
- Non-deterministic priority traffic scheduling
- Routing and switching
- Schedule generation during runtime

Software-defined application periods: Hypercycle



Traffic simulation



We thank **Renan Moreira da Silva** and **Iguatemi Eduardo da Fonseca** for the ongoing efforts for the development of TSN traffic simulation tools.
Presented at **FMCAD 2019: Formal Methods in Computer-Aided Design** in San Jose, CA.