Technical Report: Network Traversal Time (NTT) Analysis of ST Flows in TSN networks

K Pavan Kumar[0009−0007−6169−0462] and Deepak Gangadharan[0000−0001−6630−0012]

IIIT Hyderabad, India
{pavan.kondooru@research,deepak.g}@iiit.ac.in

1 Proof of Theorem 1

Theorem 1. *Let s_i^0 be the slot of interest for the transmission of f_k^p from source switch v_0. Given the arrival jitter $\delta_0, ..., \delta_{k-1}$ for k higher-priority flows, the number of instances of these flows generated prior to s_i^0 is determined by Eq.(1).*

$$hpInst(s_i^0, k, \delta_0, \ldots, \delta_{k-1}) = \sum_{l=0}^{k-1} \left(1 + \left\lfloor \frac{\eta^0(i).T_{sch} + \psi^0(i) - X - \Phi_l - \delta_l}{T_l} \right\rfloor\right) \tag{1}$$

Proof. To determine the higher-priority flow instances, we analyze timing requirements and periodic behavior of the flows. Considering s_i^0, it starts at time instant $\eta^0(i).T_{sch} + \psi^0(i)$.

For a flow instance to get transmitted through s_i^0, it must arrive at the switch and be placed in the transmission queue before the slot begins. The latest possible arrival time to be considered for s_i^0 is $\eta^0(i).T_{sch} + \psi^0(i) - X$, where X represents the propagation and switch-fabric delay.

If a flow instance arrives after this threshold, it will not be queued in time and thus cannot be transmitted through s_i^0. Since lth priority flow generates instances periodically with period T_l after an initial offset Φ_l, the time window during which flow instances could arrive and affect transmission through s_i^0 is $[\Phi_l,\ \eta^0(i).T_{sch} + \psi^0(i) - X]$.

However, the arrival jitter δ_l delays the placement of the flow instance into the queue, effectively shortening the time window by δ_l. Thus, the adjusted region of interest becomes $[\Phi_l,\ \eta^0(i).T_{sch} + \psi^0(i) - X - \delta_l]$. Given this time window, the number of higher priority flow instances is given as:

$$(1 + \lfloor \frac{\eta^0(i) \cdot T_{sch} + \psi^0(i) - X - \Phi_l - \delta_l}{T_l} \rfloor)$$

where 1 is added to account for the initial instance at $t = \Phi_l$. Finally, the total number of k higher priority flow instances is calculated by aggregating l from 0 *to* $k-1$, as described in Eq.(1).

2 Influence of Previous Flow Instances: Earliest and Latest Slot Assignment at Source Switch

While assigning the earliest and latest slots at the source switch, in certain scenarios, previous instances of a flow also influences the slot determination. One such scenario is illustrated in Figure 1. Let f_k^p be the flow instance under consideration whereas f_k^{p-1} is the previously released instance. Using equations (2) and (3), the earliest and latest slots for f_k^{p-1}, denoted as s_i^0 and $s_{i'}^0$ are determined. If s_i^0 is the first slot presented after the release of f_k^p, then s_i^0 and $s_{i'}^0$ are designated as its earliest and latest assigned slots, respectively. This is because the interference from higher priority flows preceding a slot remains same for all instances. Furthermore, since flows are transmitted in a FIFO order, f_k^{p-1} is transmitted prior to f_k^p. If f_k^{p-1} is transmitted via $s_{i'}^0$, then f_k^p must be transmitted through a slot occurring after $s_{i'}^0$. However, such slots fall outside the estimation range, resulting in incomplete slot determination. This situation arises when earliest slot of f_k^p occurs either before or coincides with the latest slot of f_k^{p-1}. In such situations, the earliest slot assignment for f_k^p remains unchanged, while for the latest, $s_{i'+1}^0$ is considered as a potential slot if it satisfies the condition $max\_interference(s_{i'+1}^0, k) = 0$; Otherwise, subsequent slots are evaluated.

$$min\_interference(s_i^0, k) = min\_hpInst(s_i^0, k) - max\_slotfilling(s_i^0, k) \quad (2)$$

$$max\_interference(s_{i+r}^0, k) = max\_hpInst(s_{i+r}^0, k) - min\_slotfilling(s_{i+r}^0, k) \quad (3)$$

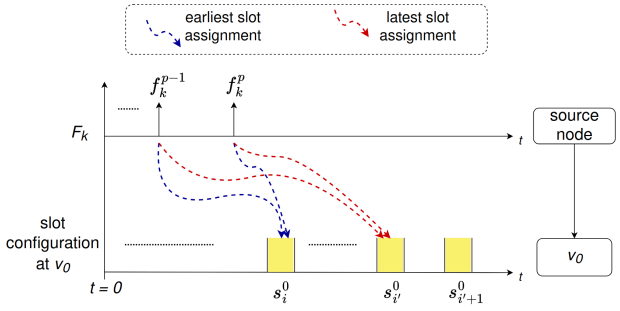

Fig. 1: Scenario illustrating f_k^{p-1}'s and f_k^p's slot estimation

3 Slot Parameters at Source and Intermediate Switches

3.1 At source switch

$$s_i^0.no\_min = \begin{cases} s_{i-1}^0.no\_min, & \text{if } min\_hpInst(s_i^0, k) - s_{i-1}^0.no\_min = 0, \\ s_{i-1}^0.no\_min + 1, & \text{otherwise} \end{cases}$$

$$s_i^0.min\_fill = \begin{cases} False, & \text{if } min\_hpInst(s_i^0, k) - s_{i-1}^0.no\_min = 0, \\ True, & \text{otherwise} \end{cases}$$

$$s_i^0.no\_max = \begin{cases} s_{i-1}^0.no\_max, & \text{if } max\_hpInst(s_i^0, k) - s_{i-1}^0.no\_max = 0, \\ s_{i-1}^0.no\_max + 1, & \text{otherwise} \end{cases}$$

$$s_i^0.max\_fill = \begin{cases} False, & \text{if } max\_hpInst(s_i^0, k) - s_{i-1}^0.no\_max = 0, \\ True, & \text{otherwise} \end{cases}$$

3.2 At intermediate switches

$$s_i^x.no\_min = \begin{cases} s_{i-1}^x.no\_min + 1, & \text{if } s_r^{x-1}.no\_min - s_{i-1}^x.no\_min \text{ != } 0 \\ s_{i-1}^x.no\_min, & \text{if } s_r^{x-1}.no\_min - s_{i-1}^x.no\_min = 0 \end{cases}$$

$$s_i^x.min\_fill = \begin{cases} True, & \text{if } s_r^{x-1}.no\_min - s_{i-1}^x.no\_min \text{ !=} 0 \\ False, & \text{if } s_r^{x-1}.no\_min - s_{i-1}^x.no\_min \text{ =} 0 \end{cases}$$

$$s_i^x.no\_max = \begin{cases} s_{i-1}^x.no\_max + 1, & \text{if } s_r^{x-1}.no\_max - s_{i-1}^x.no\_max \text{ != } 0 \\ s_{i-1}^x.no\_max, & \text{if } s_r^{x-1}.no\_max - s_{i-1}^x.no\_max = 0 \end{cases}$$

$$s_i^x.max\_fill = \begin{cases} True, & \text{if } s_r^{x-1}.no\_max - s_{i-1}^x.no\_max \text{ !=} 0 \\ False, & \text{if } s_r^{x-1}.no\_max - s_{i-1}^x.no\_max \text{ =} 0 \end{cases}$$

4 Detailed Conditions for Best-Case Reachability

Let s_j^{x+1} be the slot of interest to check for the least timed data path. The best case reachable conditions are classified into two scenarios as follows:
Scenario $1 : s_i^x$ is the immediate previous slot for s_j^{x+1}.

To obtain the earliest slot, we consider that the minimum number of higher-priority flows are transmitted up to the occurrence of s_{i-1}^x at v_x, ensuring minimum overhead. Subsequently, the maximum number of flows are assumed to be further transmitted from v_{x+1} up to the occurrence of s_{j-1}^{x+1}. This approach identifies the minimum number of interfering higher-priority flows to consider when determining whether the considered flow can be transmitted via s_j^{x+1} at v_{x+1}.

The corresponding slot reachability condition is satisfied if the net flow to v_{x+1} is less than or equal to zero. This condition is evaluated using the function $bforward()$.

$$bforward(s_i^x \rightarrow s_j^{x+1}) = \begin{cases} True, & \text{if } (s_{i-1}^x.no\_min - \\ & \quad s_{j-1}^{x+1}.no\_max) \leq 0, \\ False, & \text{otherwise} \end{cases}$$

Scenario 2 : s_{i+r}^x is the immediate previous slot for s_j^{x+1} where $r > 0$. This scenario further consists of two cases.

case-1: $s_i^x.min\_fill = True$

Setting $s_i^x.min\_fill$ to true corresponds to the situation where s_i^x is utilized by higher-priority flows that experience the maximum arrival jitter δ^{max}. However, when considering variable arrival jitter δ, it is possible that the higher-priority flows do not utilize the slot, allowing the considered flow to use it instead. Therefore, the minimum number of higher-priority flow transmissions until s_{i+r}^x is given by $s_{i+r}^x.no\_min - 1$. The corresponding condition is considered true if the net flow is less than or equal to zero.

$$bforward(s_i^x \rightarrow s_j^{x+1}) = \begin{cases} True, & \text{if } (s_{i+r}^x.no\_min - 1 - \\ & \quad s_{j-1}^{x+1}.no\_max) \leq 0, \\ False, & \text{otherwise} \end{cases}$$

case-2: $s_i^x.min\_fill = False$

In this case, the minimum number of higher-priority flows transmitted until s_{i+r}^x remains unchanged. Therefore, the net flow is calculated as $s_{i+r}^x.min\_fill - s_{j-1}^{x+1}.min\_fill$. If this value evaluates to zero, the condition is set to true.

$$bforward(s_i^x \rightarrow s_j^{x+1}) = \begin{cases} True, & \text{if } (s_{i+r}^x.no\_min - \\ & \quad s_{j-1}^{x+1}.no\_max) \leq 0, \\ False, & \text{otherwise} \end{cases}$$

It is important to note that there are multiple pairs of s_i^x and s_j^{x+1} that satisfy this condition for transferring the flow. Since we aim to consider the least timed path, the following condition must be met where the first slot satisfying the requirements is considered for best-case reachability.

$$\begin{aligned} best\_reachable(s_i^x \rightarrow s_j^{x+1}) = bforward(s_i^x \rightarrow s_j^{x+1}) \\ \wedge \neg bforward(s_i^x \rightarrow s_w^{x+1}) \\ \forall w \in [0, j-1] \end{aligned} \tag{4}$$

5 Time Complexity of the Proposed Algorithm

Let there be n ST slots in the observation window at every egress port of all TSN switches, and let the network contain M switches.

The function $imm(a, b)$ returns the slot index of the next immediate slot at switch v_{a+1} for the slot s_b^a (with slot index b) at switch v_a.

The overall time complexity is primarily driven by the best- and worst-case slot estimations at all intermediate switches. Other steps, such as slot assignment at the source switch and NTT bound determination at destination switch, take $\mathcal{O}(n)$ time.

5.1 Reachability from Source to First Intermediate Switch

For any slot s_i^0, the largest possible slot range indices at v_1 is given by $[imm(0, i), n]$. So, the number of slots in this range is $n - imm(0, i) + 1$. These estimations are required to compute the largest possible set of reachable slots at v_1.

The total number of computations for all i at v_0 is:

$$\sum_{i=0}^{n-1} [n - imm(0, i) + 1] = n^2 + n - \sum_{i=0}^{n-1} imm(0, i)$$

In the worst-case, slot range estimation covers all possible slots in the next immediate switch v_1.

5.2 Reachability Between Intermediate Switches

The same procedure is repeated for every switch pair $(v_j \rightarrow v_{j+1})$, where $j \in [1, M-2]$.

For each such pair, computations required are:

$$n^2 + n - \sum_{i=0}^{n-1} imm(j, i)$$

Since there are M switches and $M - 1$ reachability switch pairs in total, the computations required are:

$$\sum_{j=0}^{M-2} \left[n^2 + n - \sum_{i=0}^{n-1} imm(j, i) \right]$$

$$(M-1)(n^2 + n) - \sum_{j=0}^{M-2} \sum_{i=0}^{n-1} imm(j, i) \tag{5}$$

Thus, the worst-case time complexity of the proposed algorithm is:

$$\mathcal{O}(M.n^2)$$

As evident from Equation (5), the total worst-case computational cost depends on the ST slot configuration defined by the gate schedule, which is captured by $imm(j, i)$.

6 Extension to the Proposed Work

We provide an extension to analyze the NTT in networks where flows propagate from multiple source switches, converging at a common destination switch. These source switches may also serve as intermediate switches for other flows within the network. The focus is on determining the earliest slot assignment for flows, noting that the principles for the latest slot assignment also follow a similar methodology but are not elaborated here.

Consider a flow F_k, where switch v_y acts as the source switch for both F_k and k higher-priority flows, and also serves as an intermediate switch for k' flows traveling from switch v_x to v_y. The minimum number of k higher-priority instances generated ahead of s_i^y is given by $min\_hpInst(s_i^y, k)$. The minimum number of higher-priority transmissions from v_x is represented by $s_j^x.no\_min$ where s_j^x is the immediate previous slot for s_i^y at v_y. Additionally, the maximum further transmissions from $k + k'$ priority flows at v_y is given by $max\_slotfilling(s_i^y, k + k')$. Thus, the minimum interference to determine the earliest slot assignment ahead of s_i^y at v_y is now expressed as follows:

$$min\_interference(s_i^y, k, k') = min\_hpInst(s_i^y, k) + s_j^x.no\_min - max\_slotfilling(s_i^y, k + k') \quad (6)$$

These concepts can be utilized to determine slot parameters across all ports for all the switches in the network. However, a comprehensive discussion of this generalization is avoided. A further extension could involve adapting it to networks with multiple destination switches.