Zero-Buffer Data Delivery for Industrial Networks

(Invited Paper)

Kyungmo Koo*, Taejin Ha*, Namwon An*, Kyung-Joon Park†, and Hyuk Lim*
* Gwangju Institute of Science and Technology (GIST)
Gwangju 500-712, Republic of Korea
Email: hlim@gist.ac.kr
† Daegu Gyeongbuk Institute of Science and Technology (DGIST)
Daegu 711-873, Republic of Korea

***Abstract*—We propose a data-delivery scheduling scheme for synchronized multi-hop industrial networks where source nodes periodically transmit their data packets. The proposed scheme adjusts the transmission times of source nodes in the network in order to prevent data buffering of the data-flows at the intermediate switches. The data packets can be delivered to their destinations with the minimum transmission delay without packet losses at the intermediate switches.**

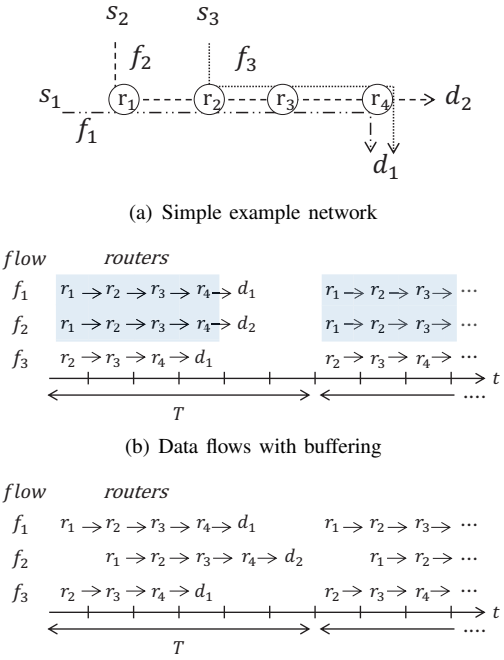
I. Introduction

For an industrial network used for real-time industrial automation applications, it is most critical and important to provide reliable, delay-bounded data delivery among device units such as sensors and actuators [1], [2]. Synchronous networks are more suitable to provide the deterministic characteristics of data delivery in industrial networks [3]. As the scale of industrial networks is usually very large, the data delivery is performed over multiple hops using a number of intermediate switches [4]. In such a multi-hop network topology, if a node transmits its data using its own scheduling decision, some buffer overflows may occur at a switch if the incoming traffic rate exceeds the relay capacity of the switch [5]. To prevent the buffer overflows at the switches, the buffer size can be increased, but the increase of buffer size causes the increase of transmission delay for the data packet delivery [6].

In a synchronous network, if each unit can cooperatively adjusts its transmission timing such that no two packets pass through a same switch simultaneously, the number of packets pending at the queue of intermediate switches is kept to nearly zero; the data can be delivered to the corresponding destination with the minimum transmission delay without packet losses at the intermediate switches. In this paper, we propose a transmission time scheduling scheme that adjusts the transmission times of source nodes in the network to prevent data buffering of the data-flows at the intermediate switches using the path and routing information of data-flows when source nodes periodically transmit the data packets with a delivery deadline constraint.

II. Transmission time scheduling

We consider a multi-hop synchronous networks wherein time is synchronized and divided into slots of equal length. We make several assumptions for the data-flows as follows: 1) Each source node transmits data periodically to the destination nodes with a fixed transmission interval of T slots. The deadline for the data generated by the source nodes coincides with the period T. 2) Each data-flow hops one relay node at one time slot, and data-flows collide with each other when they attempt to pass the same relay node simultaneously because it is assumed that the switches do not have a buffer that can hold data packets. Figure 1(a) shows a simple network topology with three data flows. Figures 1(b) and (c) indicate the periodic data traffic and route information along with the time slots. In Fig. 1(b), all of the data-flows transmit their data simultaneously, such that f_1 and f_2 collide with each other at every transmission-cycle. In contrast, in Fig. 1(c), f_2 transmits its data at the second time slot, such that all of the data-flows transmit their data without a collision or a buffer overflow at every intermediate switch.



(a) Simple example network

(b) Data flows with buffering

(c) Data flows without buffering

Fig. 1. Example of transmission time scheduling

We propose a transmission time scheduling scheme that can prevent data buffering of the data-flows at every inter-

Algorithm 1 Proposed scheduling algorithm

1: Sort the data-flows in the order of long length
2: **for** $u = \max\{L_i\}$ to $\sum_{i=1}^{F} L_i$ **do**
3: **for** $i = 1$ to $(F-1)$ **do**
4: **for** $j = i+1$ to F **do**
5: **if** f_i collides with f_j **then**
6: ret = procedure in Fig. 2
7: **if** ret == 'A' **then**
8: go to line 3 // collision test for all flows
9: **else**
10: go to line 18 // no further candidate
11: **end if**
12: **end if**
13: **end for**
14: **end for**
15: **if** no collisions occur among the data-flows **then**
16: return the current schedule
17: **end if**
18: set transmission times of all data flows as 1
19: **end for**

mediate switch. The path and routing information of data-flows are exploited together with the delivery timing at each transmission-cycle on the synchronous network in order to determine whether any data-flows pass through the same relay node simultaneously. If so, the transmission-time of each data-flow needs to be adjusted such that no two data-flows pass through the same relay node simultaneously.

Let F and u denote the number of data flows and the minimum number of time slots required for all source nodes to deliver their data packets successfully at least once in u time slots, respectively. Note that $\max\{L_i\} \leq u \leq \sum_{i=1}^{F} L_i$, where L_i is the length of the ith flow because u cannot be smaller than the longest length of flows. If the transmissions are sequentially scheduled back-to-back, it corresponds to the worst case, resulting in $u = \sum_{i=1}^{F} L_i$. If a schedule achieves the smallest u and $u \leq T$, it is one of the optimal schedules. Note that the remaining time interval between u and T can be used for serving elastic traffic of non-realtime network applications.

The scheduling problem is formulated as follows:

$$\min_{\mathbf{x} \in \mathcal{F}} \left(\max_{i} (x_i + L_i) \right) \tag{1}$$

where $\mathbf{x} = [x_1, x_2, .., x_F]$ and $\mathcal{F}$ denote the number of data flows and the vector whose elements are the transmission-time-slot indices of the data-flows, and the set of feasible solutions in which there is no collision among flows at any switches, respectively. For a given value of u, the total number of possible candidate $\mathbf{x}$'s is given by $\prod_{i=1}^{F} (u - L_i + 1)$, which is too large to be solved by a brute-force algorithm.

Algorithm 1 illustrates the procedures of our greedy scheduling scheme that can expedite the search of optimal $\mathbf{x}$. In a nutshell, starting from $u = \max\{L_i\}$, the algorithm generates all possible candidates for a valid schedule with

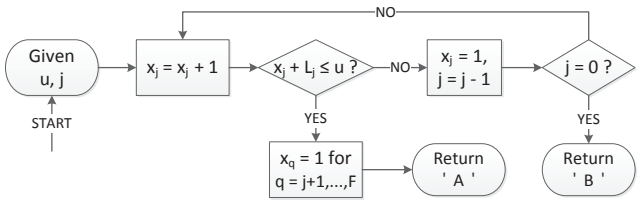
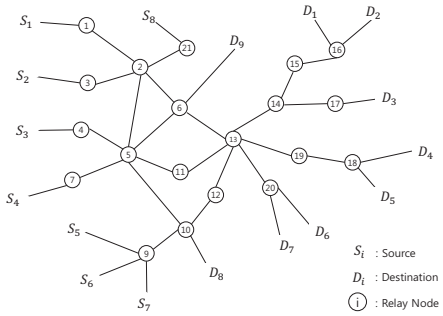
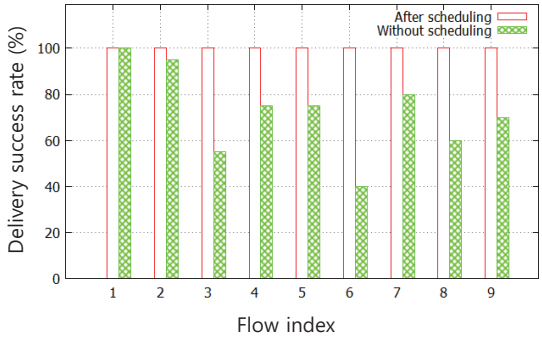


Fig. 2. Procedure for generating next candidate $\mathbf{x}$ that does not exceed u



(a) Multi-hop network topology



(b) Delivery success rate

Fig. 3. Simulation results.

collision-free data-transmission for the given u, and if no valid candidate is found, u is increased by 1. On the lines 3–5, it checks if the two flows f_i and f_j collide in the current schedule. If no collision occurs for all i's and j's, the current schedule is the optimal schedule because the algorithm starts to generate all of the possible candidates from the minimum value of u. If a collision is detected, it generates a new candidate of $\mathbf{x}$ using the procedure in Fig. 2. If the procedure returns 'B', it implies that there is no further possible candidate and u should be increased by 1.

III. Simulation Results

We evaluated our scheduling scheme using NS-2 simulations. In the simulations, the buffer sizes of all nodes were set as 1. In a random access approach with transmission time scheduling, a flow starts its data delivery at a time slot randomly selected from the range between the 1st and $(u - l_i + 1)$th time slots in each cycle. The period of the data-flows is 0.12 s, and the time-slot length is 0.01 s. Figure 3(a) shows one of the network topologies used in the simulations,

and Fig. 3(b) shows the delivery success rates of the proposed scheduling and the random access without scheduling. These results indicate that all the packets are successfully delivered without packet losses for the proposed scheme.

IV. Conclusion

We proposed a scheduling scheme that uses the route information of data-flows to avoid data packet buffering at the intermediate switches. Using the proposed algorithm, the source nodes adjust the transmission times for delivering all of the periodic data without a queuing delay within a minimum period. The simulation results indicate that our scheduling scheme provides more reliable data delivery than the random access approach.

Acknowledgment

This work was supported in part by the NRF funded by MSIP of the Korea government (2014R1A2A2A01006002), and by the ICT R&D program of MSIP and IITP of the Korea government (14-824-09-013, Resilient CPS Research).

References

[1] Industry-4.0, http://www.theengineer.co.uk/manufacturing/automation/industry-40-the-next-industrial-revolution/1016696.article.

[2] W. Zhang, M. S. Branicky, and S. M. Phillips, "Stability of networked control systems," *IEEE Control Systems*, vol. 21, no. 1, pp. 84 – 99, Feb 2001.

[3] L. Hardy and M. Gafen, "A new highly-synchronized wireless mesh network model in use by the electric company to switch to automatic meter reading: Case study," *International Conference on Networked Sensing Systems (INSS)*, pp. 31–34, June 2008.

[4] V. C. Gungor and G. P. Hancke, "Industrial wireless sensor networks: Challenges, design principles, and technical approaches," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 10, pp. 4258 – 4265, Oct 2009.

[5] Z. Fu, P. Zerfos, H. Luo, S. Lu, L. Zhang, and M. Gerla, "The impact of multihop wireless channel on TCP throughput and loss," *IEEE INFOCOM*, pp. 1744–1753, March 2003.

[6] Cheng-Shang Chang, "Stability, queue length, and delay of deterministic and stochastic queueing networks," *IEEE Transactions on Automatic Control*, vol. 39, no. 5, pp. 913 – 931, May 1994.