Periodic and Aperiodic Communication Techniques for Responsive Link [*]

Shinpei Kato
Department of Computer Science
The University of Tokyo, Japan
shinpei@il.is.s.u-tokyo.ac.jp

Yuji Fujita and Nobuyuki Yamasaki
Department of Information and Computer Science
Keio University, Japan
{fujita,yamasaki}@ny.ics.keio.ac.jp

Abstract

Responsive Link, an ISO/IEC communication standard, provides many functional capabilities for distributed real-time systems. This paper is focused on periodic and aperiodic communication techniques for Responsive Link. In periodic communication, the priority is assigned to each packet so that the network utilization is improved. A schedulability test for connection establishments is also derived to ensure timing guarantees. In aperiodic communication, meanwhile, the bandwidth is reserved to improve response time as much as possible without periodic timing violations. The effectiveness of the presented techniques is demonstrated through a series of simulations.

1 Introduction

In distributed real-time systems, individual nodes communicate with each other under timing constraints. Transmissions must be therefore delivered to destinations by certain deadlines. Such communication is often referred to as *real-time communication*. In many cases, communication techniques and communication links determine the performance of real-time communication.

Communication techniques are closely relevant to communication standards. Though the most well-known standard is Ethernet, specialized communication schemes [2, 11, 5] are required for hard real-time communication. Ethernet is rather effective to soft real-time communication [6]. Switched Ethernet is often made used of for real-time communication [14, 4, 10], but it does not make Ethernet fully deterministic. For instance, if a burst of messages destined to a single port arrives at the switch in a short time interval, they must be serialized and transmitted one after another.

In fact, we encounter timing problems after all in Ethernet, since it is not designed for real-time communication. ATM is preferred to Ethernet for real-time communication in that the communication bandwidth can be reserved. It is however not suitable for embedded products, e.g. distributed control applications, due to complex implementation. Controller Area Network (CAN) and Process Field Bus (PROFIBUS) are often used in cars and factory automation, as real-time communication standards, though the scalability is problematic: the waiting time to access the network is increased as the number of nodes and the scale of the network is enlarged.

This paper is focused on Responsive Link, which is an ISO/IEC communication standard for distributed real-time systems. To make efficient use of Responsive Link, we consider periodic and aperiodic communication techniques. The goal of periodic communication is to improve the network utilization with timing guarantees, while that of aperiodic one is to reduce the response time as much as possible without causing periodic timing violations.

The rest of this paper is organized as follows. In the next section, Responsive Link and our network model are briefly described. Section 3 and Section 4 present communication techniques for Responsive Link. Those presented techniques are evaluated through simulations in Section 5. This paper is concluded in Section 6.

2 Responsive Link and Network Model

Responsive Link, which was originally developed in [15] and has been standardized as ISO/IEC 24740:2008, provides many functional capabilities for distributed real-time systems. Particularly, the priority-based packet-overtaking function is effective to real-time communication.

The latest specification of Responsive Link has 256-level priorities for each packet. In the Responsive Link switch, every time packets arrive at input ports, they are transmitted to output ports according to their priorities. Specifically, lower-priority packets are stored in an SDRAM buffer, implemented in the Responsive Link switch, to wait for higher-priority packets to be transmitted. If no more than one packet competes for the same output port, a packet is just forwarded from an input port to an output port.

In this paper, we assume that each node is connected by Responsive Link with point-to-point and full-duplex. Each packet may hop several intermediate nodes, namely we assume a multihop network. The transmission mode has two options: cut-through and store-and-forward. Since real-time communication is sensitive in delays, we focus on the cut-through mode. That is, an input packet is transmitted right after its header is read.

[*]This work is supported in part by the fund of New Energy and Industrial Technology Development Organization (NEDO).

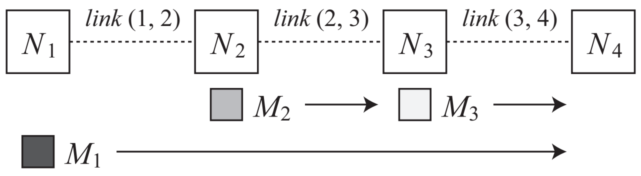


Figure 1. Example of network connections.

Each periodic message M_i is characterized by a set (T_i, D_i, C_i), where T_i is a period, D_i is a relative deadline, and C_i is the size of message per period. Note that (T_i, D_i, C_i) is abbreviated as (T_i, C_i), if T_i is equal to D_i. A set of all the periodic messages is denoted by $\mathcal{M}$. Meanwhile. the system does not know when periodic messages arrive and their size.

Each connection is protected by a real-time channel [3]. That is, a node sends a request message for establishing a real-time channel before the transmissions of packets on the corresponding connection starts.

This paper is not focused on routing problems. For simplicity, we take a static routing algorithm, Shortest Path First. Software implementation problems for the presented techniques are also not the subject of this paper.

3 Periodic Communication Technique

The packet-overtaking function of Responsive Link enables us to consider preemptive packet scheduling. Given that each packet of Responsive Link has a fixed priority, we make use of Deadline Monotonic (DM) [8]. Note that for periods equal to deadlines, DM performs as Rate Monotonic (RM) [9].

3.1 Priority Assignment by Deadlines

DM is known to be an optimal algorithm for CPU scheduling. We notice, however, that DM may perform poorly for packet scheduling, particularly over multihop networks. Let us consider an example with three messages $M_1 = (10, 3)$, $M_2 = (9, 5)$, and $M_3 = (6, 2)$ over four nodes N_1, N_2, N_3, and N_4. Here, M_1 is delivered from N_1 to N_4, M_2 is delivered from N_2 to N_3, and M_3 is delivered from N_3 to N_4, as depicted in Figure 1.

Figure 2 illustrates the DM scheduling of those three messages. M_1 competes with M_2 and M_3, but it is assigned the lowest priority by DM. Since M_1 has to hop three links, it is likely to wait for higher-priority packets to transmit on each middle node. As a result, it may cause timing violations due to the delays. For instance, if deadlines are equal to periods, M_1 misses deadlines at time 10 and time 20.

3.2 New Priority Assignment Policy

In this section, we present a new priority assignment policy, Virtual Deadline Monotonic (VDM). The poor behavior of DM depicted in Figure 2 occurs, because we did not

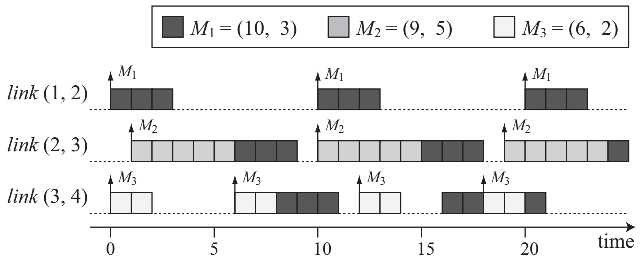


Figure 2. Example of DM packet scheduling.

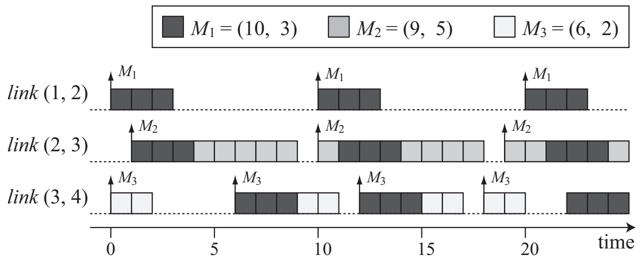


Figure 3. Example of VDM packet scheduling.

concern about multi-hop packets. VDM virtually splits the deadline of each message that hops more than one link. The virtual relative deadline D'_i of M_i that hops H_i of links is expressed by Equation (1).

$$D'_i = \frac{D_i}{H_i} \tag{1}$$

Consider the previous example again. Since M_1 hops three links, its deadline is virtually split to $3/10 = 3.33$. Hence, it is assigned the highest priority according to VDM. Figure 3 then illustrates the VDM packet scheduling of the same three messages. We see the effectiveness of VDM in that M_1 is not blocked by M_2 and M_3 any more.

It is obvious that VDM outperforms DM on Responsive Link multi-hop networks, given that the worst-case arrival rate on each middle node is limited by the virtual deadline in VDM, while that is just limited by the original deadline in DM. With the virtual deadline $D'_i = D_i/H_i$ in the case of H_i hops, VDM guarantees a multi-hop message M_i to be delivered to the destination within $D'_i \times H_i = D_i$, if the schedulability is guaranteed.

We now consider improving VDM. The virtual deadline assignment based on Equation (1) pessimistically assumes that the transmissions of each multi-hop message M_i are not overlapped (separated) among the links. Figure 4 illustrates this assumption. However, as shown in Figure 3, the transmissions can be in fact overlapped, since Responsive Link offers such a network model that transmits a packet arriving one node to the next node at the next time clock.

We take into account this characteristic of Responsive Link. Let C^* be the transmission time of one packet. Since the first packet of a message M_i arriving on one node at time t is always transmitted to the next node at time $t+C^*$ on

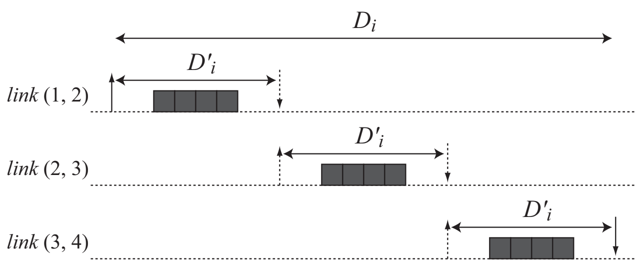


Figure 4. Separated transmission.

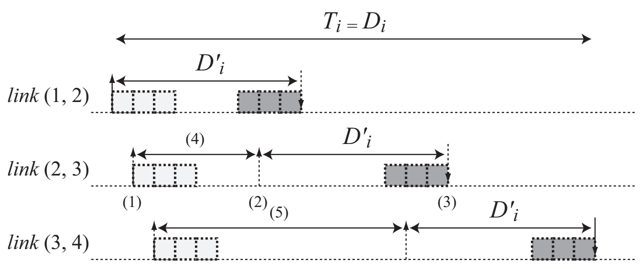


Figure 5. Analysis of jitters.

Responsive Link networks, the remaining transmission time $C_i - C^*$ of M_i can be considered as the available transmission time on the next node. So during H_i hops, the transmission time $(C_i - C^*) \times (H_i - 1)$ can be overlapped for M_i. We should also append this time interval for the available transmission time in addition to D_i. Thus, we renew the definition of the virtual deadline by Equation (2).

$$D'_i = \frac{D_i + (C_i - C^*) \times (H_i - 1)}{H_i} \tag{2}$$

3.3 Scheduling Analysis

The response time analysis (RTA) [1] is used to derive a schedulability test for VDM. The response time of the transmission between two nodes lying side-by-side is the time duration from the arrival of the message to the completion of sending the message to the output node.

We here need to take into account that the arrival rate of a message on a middle node is not equal to its release rate due to different workloads of nodes over multi-hop networks. We first of all assume that all messages are guaranteed to meet both virtual and true deadlines. A message M_i released at time t certainly arrives at each node by $t + D_i - C_i$ at the latest. Such a maximum delay $D_i - C_i$ from the release time is considered as the release jitter problem in [1]. Hence, the worst-case response time $W_i^{x,y}$ of M_i on $link(x, y)$ is obtained by Equation (3), where $hp_{(i)}^{x,y}$ denotes a set of messages whose priorities are higher than M_i.

$$W_i^{x,y} = \sum_{\forall M_j \in hp_{(i)}^{x,y}} \left(\left\lceil \frac{W_i^{x,y} + D_j - C_j}{T_j} \right\rceil \times C_j \right) + C_i \tag{3}$$

We claim that Equation (3) is pessimistic with respect to jitters. Since each link has a virtual deadline, we can more precisely predict the time by which a message is transmitted to the next node. Figure 5 helps to analyze the precise jitters of a message M_i that is delivered from N_1 to N_4 through N_2 and N_3. The white boxes represent the earliest transmissions of M_i, while the gray boxes do the latest ones. Let us focus on $link(2, 3)$. In the figure, (1) and (2) denote the earliest and the latest pseudo-release time of M_i respectively. (3) is the time separated by D'_i from (2). Note that the message transmitted to N_2 at time (2) must be transmitted to N_3 by (3). (4) is then computed by (2) - (1), which is equal to $D'_i - C_i$. By the same token, (5) is expressed by (4) + $D'_i - C_i = 2(D'_i - C'_i)$. Hence, the maximum jitter J_i between the latest pseudo-release and the earliest pseudo-release of M_i on any $link(x, y)$ can be obtained by Equation (4), where $l_i^{x,y}$ means that $link(x, y)$ is the $l_i^{x,y}$th link for M_i.

$$J_i = (l_i^{x,y} - 1) \times (D'_i - C_i) \tag{4}$$

Finally, we can improve Equation (3) to Equation (5) based on the above jitter analysis so that the worst-case response time is more precisely bounded.

$$W_i^{x,y} = \sum_{\forall M_j \in hp_{(i)}^{x,y}} \left(\left\lceil \frac{W_i^{x,y} + J_i}{T_j} \right\rceil \times C_j \right) + C_i \tag{5}$$

The schedulability test is designed so that the connection establishment of each message M_i is accepted, if the worst-case response time $W_i^{x,y}$, obtained by Equation (5), does not exceed its virtual deadline D'_i for all links $link(x, y)$ where M_i is transmitted.

4 Aperiodic Communication Technique

Traditionally, aperiodic requests are often handled by a *server*. A server is a special periodic task whose purpose is to service aperiodic requests as soon as possible. Like a periodic task, a server is characterized by a period and an execution time. The execution time assigned to a server is called *budget*. A server is scheduled by the same algorithm used for periodic tasks, and once active, it serves aperiodic requests in the range of the budget. The budget is replenished at every period.

We make use of server techniques for aperiodic communication over Responsive Link networks. Since the objective of servers is to reduce the response time of aperiodic messages, we consider only the case in which servers are assigned the highest priority. If the budget is exhausted, aperiodic messages are transmitted in background until the budget is replenished.

4.1 Budget Management

We have two options to manage the budget: per-node management and per-connection management. In the per-node management, each node has the budget, and if the budget is remaining when an aperiodic message arrives at the node, the message can be immediately transmitted to the next node. Meanwhile, in the per-connection management, the network bandwidth is reserved for each aperiodic connection, and only the source nodes have the budget. The message can be then delivered to the destination node, only if the budget of the source node is remaining.

In Responsive Link, when one node receives a packet, the packet can be automatically transmitted to the next node by hardware according to the routing table. Though the per-node management has such advantage that can deal with the packets of *any* aperiodic messages as long as the per-node budget is remaining, it is not suitable for Responsive Link in that each node has to take in each packet before it is transmitted to the next node so as to verify if the budget is remaining, which leads to the increase of communication delays. In addition, the packet-overtaking function is never exploited, that is, there is no advantage to use Responsive Link. Therefore, we adopt the per-connection management. Note that we must know in advance which paths are used by aperiodic messages, and we must keep connections for aperiodic messages, once they are established. Otherwise, we need to establish a connection every time one aperiodic message is generated.

Henceforth, S_k stands for the server of the kth connection established for aperiodic messages. U_k^{srv}, C_k^{srv} and T_k^{srv} then denote the bandwidth, the budget and the period of S_k respectively. An issue of concern here is how to reserve the server bandwidth.

4.2 Server Bandwidth

We assume that all periodic messages are guaranteed to meet deadlines. That is, $W_i^{x,y} \leq D_i'$ is satisfied for any M_i and $link(x, y)$ that M_i uses. Let $B_{x,y}$ be the bandwidth available for the servers on $link(x, y)$. Since the servers should not violate the timing constraints of periodic messages, $B_{x,y}$ is computed by Equation (6), where $\mathcal{M}_{x,y}$ is a set of periodic messages whose connections go through $link(x, y)$.

$$B_{x,y} = \min\left\{ \frac{D_i' - W_i^{x,y}}{T_i} \;\middle|\; M_i \in \mathcal{M}_{x,y} \right\} \tag{6}$$

If multiple connections established for aperiodic messages go through the same link, the network bandwidth should be shared among the servers of those connections. However, each node cannot verify by itself if the network bandwidth of the link is used by some connections for aperiodic communication, because the transmission is controlled by only the source nodes in the per-connection management. It is therefore difficult to reserve the network bandwidth exclusively for the server of each connection.

The approach here is that we divide the network bandwidth by the number of connections, a part of which has been established on the link so that each connection can reserve its own network bandwidth. Dividing the network bandwidth enables each connection to assign a server without the interference from the other connections competing for the same link. Here, since we adopt the per-connection management, the server bandwidth of one connection must be the same on each link through which the connection is established. To this end, any server bandwidth U_k^{srv} is obtained by Equation (7), where $\mathcal{L}_k$ is a set of links through which the connection managed by S_k is established, and $n_{x,y}$ is the number of connections going through $link(x, y)$.

$$U_k^{srv} = \min\left(\frac{B_{x,y}}{n_{x,y}} \;\middle|\; link(x, y) \in \mathcal{L}_k \right) \tag{7}$$

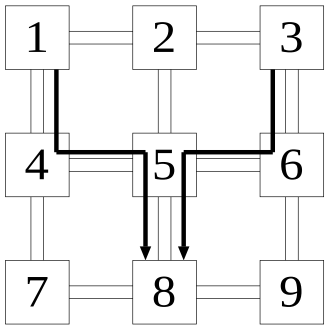
Figure 6 shows an illustrative example of assigning the network bandwidth to the server. The network is a mesh topology composed of nine nodes. We assume that the system wants to reduce the response time of two aperiodic messages, as shown in Figure 6(a). One is delivered from N_1 to N_8 via $N4$ and N_5. The other is meanwhile delivered from N_3 to N_8 via N_6 and N_5. First, we compute the network bandwidth available for the server for each link. Let us assume that the resulting bandwidth is $2/5$ for every link. We then count the number of the connections that go through each link, as shown in Figure 6(b). The available server bandwidth for each link is computed such that the value of the network bandwidth divided by the number of the connections, as shown in Figure 6(c). In the case of the connection established from N_1 to N_8, we obtain the available server bandwidths, $2/5$, $2/5$ and $1/5$. Finally, the server bandwidth for this connection is decided to be the minimum of them, which is thus $1/5$. In the case of the one established from N_3 to N_8, we also obtain the same result.

4.3 Server Algorithms

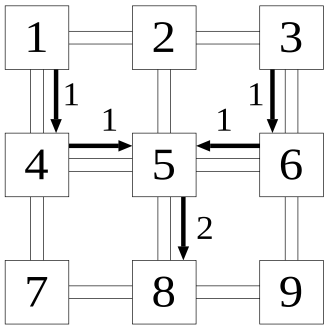
In this paper, we take Polling Server [7, 12], Deferrable Server [7, 13], and Periodic Server (a hybrid of Polling Server and Deferrable Server), as server algorithms for aperiodic communication over Responsive Link networks. Polling Server released at time t can transmit only the aperiodic messages that arrive before t within the budget. Deferrable Server is similar to Polling Server but differs in that any aperiodic messages can be transmitted within the budget anytime, and the budget is consumed only when it is used for transmitting aperiodic messages. Periodic Server is a hybrid of them: namely any aperiodic message can be transmitted within the budget anytime, but the budget is always consumed as the time goes by regardless of whether it is used for transmitting aperiodic messages or not.

For all the algorithms, the period T_s of the server is set by the minimum virtual deadlines of periodic messages, i.e. $T_s = \min(D_i' \mid \forall M_i)$, so that the server is assigned the highest priority according to VDM. Aperiodic messages are transmitted in background when the server is not active (the budget is exhausted).

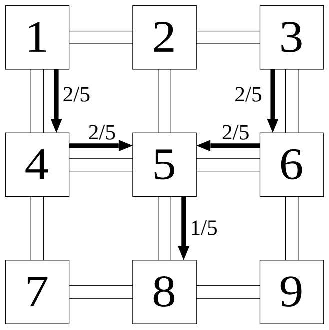
The budget C_k^{srv} of any server S_k must be determined so that it never violates the timing constraints of periodic messages. By the packet-overtaking function of Responsive Link, packets of aperiodic messages transmitted by the highest priority are always transmitted to the next node



(a) Connections using servers.



(b) The number of the connections.



(c) Available bandwidth per connection.

Figure 6. Illustrative example of assigning network bandwidth to server.

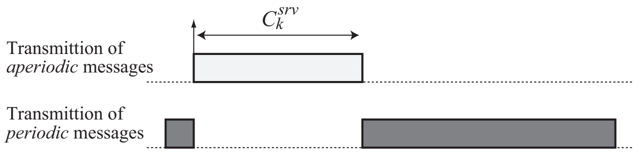

C_k^{srv}
Transmittion of *aperiodic* messages
Transmittion of *periodic* messages

Figure 7. Blocking by Polling Server.

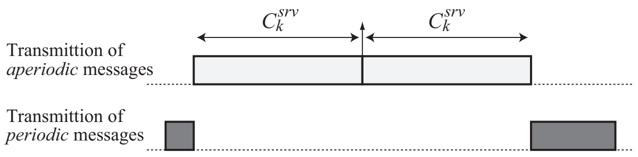

C_k^{srv}
C_k^{srv}
Transmittion of *aperiodic* messages
Transmittion of *periodic* messages

Figure 8. Blocking by Deferrable Server.

without any interference from lower-priority transmissions. Thus, the server interferes periodic messages the most when the budget is fully utilized by aperiodic messages.

According to [13], periodic messages are blocked by at most one instance in Polling Server as shown in Figure 7, and by at most two instances in Deferrable Server as shown in Figure 8. It is clear that Periodic Server may have the same situation as Polling Server when the budget is fully utilized. Finally, the budget of any server S_k in Polling Server or Periodic Server is derived by Equation (8), and that in Deferrable Server is derived by Equation (9).

$$C_k^{PS} = U_k^{srv} \times T_k^{srv} \tag{8}$$

$$C_k^{DS} = U_k^{srv} \times T_k^{srv} \times \frac{1}{2} \tag{9}$$

5 Evaluation

We evaluate the presented communication techniques for Responsive Link through a series of simulations. The performance metric for periodic communication is the acceptance ratio of the connection establishment, while that for aperiodic communication is the mean response time of the aperiodic messages, with respect to the network loads.

The simulations use a 15-node tree structure depicted in Figure 9 as a network topology for simulations. The network utilization of periodic messages is defined by $U_p = \sum_{\forall M_i \in \mathcal{M}} C_i/T_i \times 1/L$, where L is the number of unidirectional links between two nodes lying side-by-side.

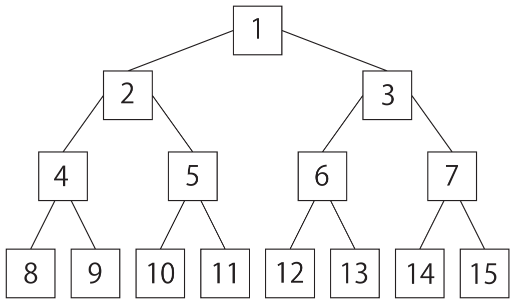


Figure 9. Network topology for simulations.

Table 1. Parameters of periodic messages.

	setup 1	setup 2
period	[100, 1000]	[1000, 2000]
message size	[1, 10]	[100, 500]

5.1 Evaluation of Periodic Communication

The periods and message sizes of periodic messages are uniformly distributed. As for the range of the distributions, we prepare two setups as indicated in Table 1. The first setup generates messages with short periods and small message sizes, while the second one generates messages with relatively long periods and large message sizes. The source and destination nodes are randomly determined. The acceptance ratio is measured in such a way that each connection requires for establishing a real-time channel, and the requirement is accepted if the worst-case response time on

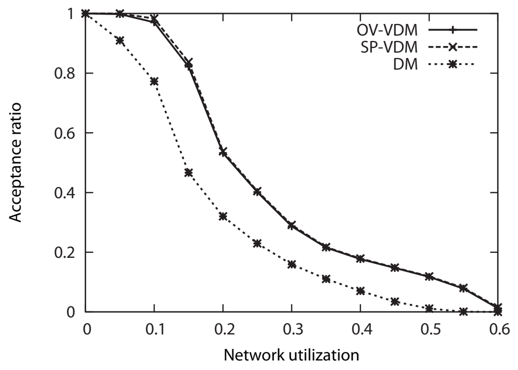


Figure 10. Comparison of priority assignment policies ($T_i = [100, 1000]$ and $C_i = [1, 10]$).

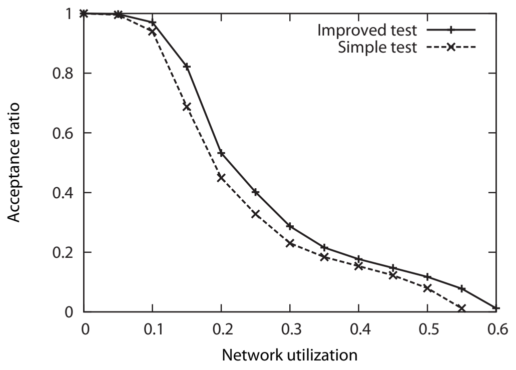


Figure 12. Comparison of schedulability tests ($T_i = [100, 1000]$ and $C_i = [1, 10]$).

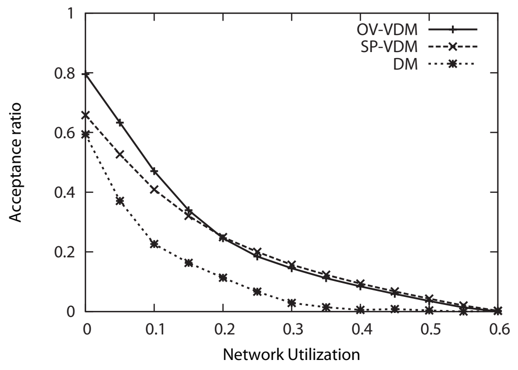


Figure 11. Comparison of priority assignment policies ($T_i = [1000, 2000]$ and $C_i = [100, 500]$).

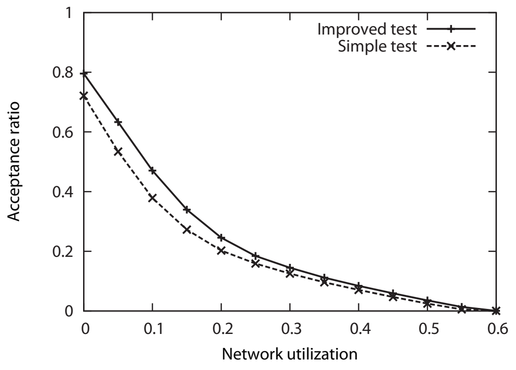


Figure 13. Comparison of schedulability tests ($T_i = [1000, 2000]$ and $C_i = [100, 500]$).

each link does not exceed the deadline. The worst-case response time is obtained by the presented analysis, that is, by Equation (5). The deadline of a message M_i is D'_i for VDM and is D_i for DM.

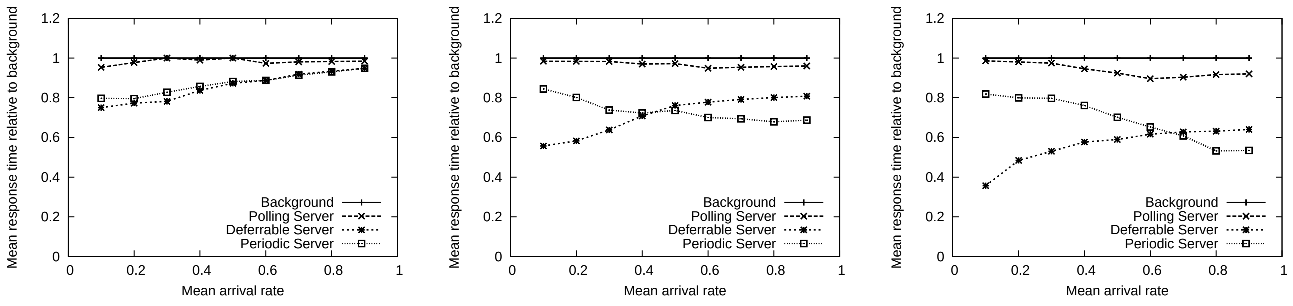
Figure 10 and Figure 11 show performance comparisons of priority assignment policies. Here, "SP-VDM" represents VDM with separated virtual deadlines based on Equation (1), and and "OV-VDM" represents the one with overlapped virtual deadlines based on Equation (2). For both setup 1 and setup 2, SP-VDM and OV-VDM outperform DM. This result is not surprising, since dividing deadlines with respect to the number of hops clearly improves schedulability, as we explained in Section 3. It is however remarkable that OV-VDM does not outperform SP-VDM for the first setup. This is mainly because overlapping virtual deadlines does not help to increase the transmission time per link, if message sizes are small. Thus, OV-VDM is more effective for such networks that generate large-size messages.

Figure 12 and Figure 13 show performance comparisons of schedulability tests. The schedulability tests are conducted on OV-VDM, which is the best performer in the simulated priority assignment policies. Here, "Simple test" represents the test using Equation (3) for the worst-case response time analysis, while "Improved test" does the one using Equation (5). Since the improved test takes into account the jitters more than the simple one, the acceptance ratio is improved for every network utilization. From those simulations, the effectiveness of the presented worst-case response time analysis can be observed.

5.2 Evaluation of Aperiodic Communication

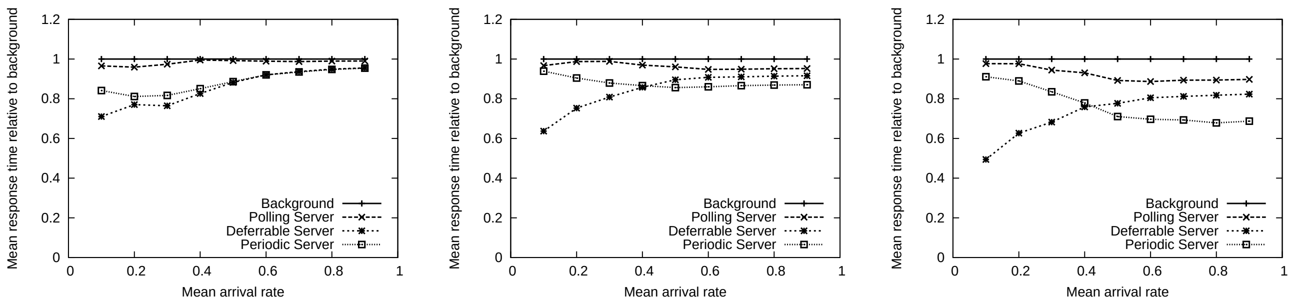
We next evaluate the server algorithms with the presented per-connection management scheme for aperiodic communication. Periodic messages are scheduled by OV-VDM. The mean response time is measured in such a way that we randomly generate aperiodic messages over the network where periodic messages have already established their connections. We use setup 2 in Table 1 for the parameters of periodic messages. We then assume that aperiodic messages are generated based on the poison arrival model. Due to limitation of space, we generate only three types of aperiodic messages: (i) ones delivered from N_8 to N_9 in 2 hops, (ii) ones delivered from N_8 to N_{11} in 4 hops, and (iii) ones delivered from N_8 to N_{15} in 6 hops.

Figure 14 shows the mean response time of aperiodic messages whose mean service rates are $\mu = 0.05$, when the network utilization of periodic messages is $U_p = 0.1$. All results are normalized based on the results of background service. For those aperiodic messages who hop 2 links, the



(a) 2-hop case. (b) 4-hop case. (c) 6-hop case.

Figure 14. Comparison of server algorithms ($\mu = 0.05$ and $U_p = 0.1$).



(a) 2-hop case. (b) 4-hop case. (c) 6-hop case.

Figure 15. Comparison of server algorithms ($\mu = 0.05$ and $U_p = 0.2$).

response time is not much reduced by the servers, as compared to those who hop 4 or 6 links. Since the servers have more chance to reduce the response time when there are more links to hop, they are effective to multi-hop networks.

Periodic Server and Deferrable Server outperform Polling Server, since they can transmit aperiodic messages anytime as long as the budget is remaining. Periodic Server and Deferrable Server are superior to each other, depending upon the circumstances. Mostly, Deferrable Server is superior to Periodic Server when the load of aperiodic communication is small, but is inferior when the load is high. We consider that is because the budget is consumed only when it is used for transmitting aperiodic messages in Deferrable Server, whereas it is consumed as the time goes by in Periodic Server. If aperiodic messages do not arrive frequently, the budget is just wasted in Periodic Server. However, if they arrive frequently, Periodic Server can be likely to fully utilize the budget, and its budget is assigned twice as much as Deferrable Server, as expressed by Equation (8) and Equation (9).

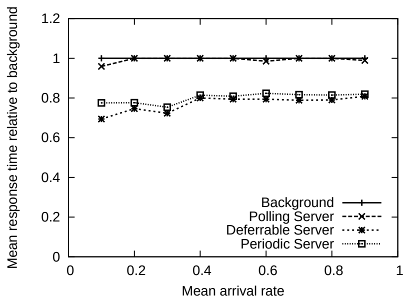
Figure 15 shows the results of the simulations in which the network utilization of periodic messages is increased to $U_p = 0.2$ from $U_p = 0.1$. The relative order of the servers in terms of the mean response time is not changed, but their performance improvements over background service are smaller than the case of $U_p = 0.1$. It is obvious that the response time of aperiodic messages is increased, as the network utilization of periodic messages is increased, and so the benefits of using servers are not much obtained.

Figure 16 and Figure 17 depict the mean response time of aperiodic messages with mean service rates $\mu = 0.1$. As the previous two cases, Periodic Server and Deferrable Server outperform Polling Server, but Deferrable Server is more often superior to Periodic Server. We consider that is because greater service rates lead to smaller message sizes, and Deferrable Server receives more benefit, since it can be more likely to complete transmissions of messages, though the budget is half as much as Periodic Server.

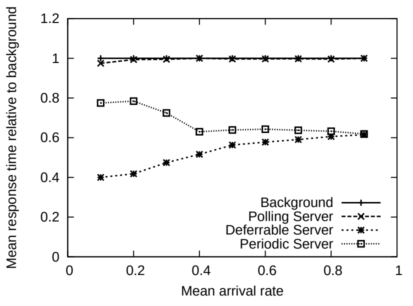
6 Conclusion

We studied real-time communication over Responsive Link networks. For periodic communication, the priority assignment policy, called VDM, is proposed. A schedulability test is also derived. For aperiodic communication, we considered the per-connection management scheme for the server algorithms. To the best of our knowledge, this is the first challenge to consider both periodic and aperiodic communication techniques for Responsive Link.

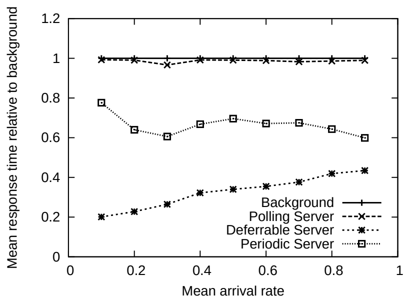
According to the simulation results, VDM improved the acceptance ratio of connections of periodic messages, as compared to the simple DM policy. We also demonstrated that the tightness of the schedulability test was improved by the presented worst-case response time analysis. In addition, we showed that the server algorithms work effectively to reduce the response time of aperiodic messages over Responsive Link networks, using together the presented per-connection management scheme.



(a) 2-hop case.

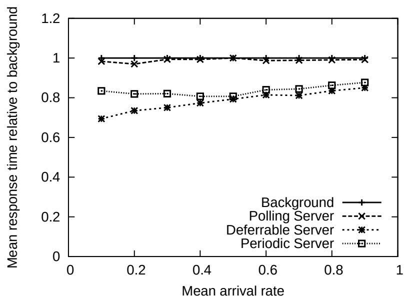


(b) 4-hop case.

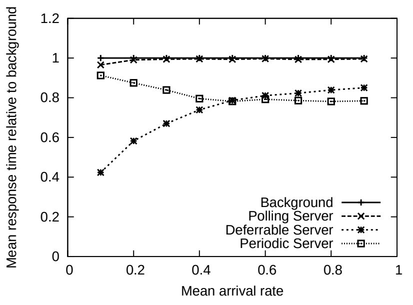


(c) 6-hop case.

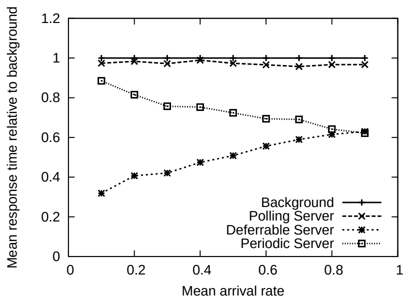
Figure 16. Comparison of server algorithms ($\mu = 0.1$ and $U_p = 0.1$).



(a) 2-hop case.



(b) 4-hop case.



(c) 6-hop case.

Figure 17. Comparison of server algorithms ($\mu = 0.1$ and $U_p = 0.2$).

References

[1] N. Audsley, A. Burns, M. Richardson, K. Tindell, and A.J. Wellings. Applying new scheduling theory to static priority pre-emptive scheduli ng. *Software Engineering Journal*, 8(5):284–292, 1993.

[2] J. Chen, Z. Wang, and Y. Sun. Real-time capability analysis for switch industrial ethernet traffic priority-based. In *Proc. of the IEEE International Conference on Control Applications*, 2002.

[3] D. Ferrari and D.C. Verma. A scheme for real-time channel establishment in wide-area networks. *IEEE Transactions on Selected Areas in Communications*, 1990.

[4] H. Hoang, M. Jonsson, U. Hagstrom, and A. Kallerdahl. Switched real-time ethernet with earliest deadline first scheduling protocols and traffic handling. In *Proc. of the IEEE International Parallel and Distributed Processing Symposium*, pages 94–99, 2002.

[5] F. Kanehiro, Y. Ishiwata, H. Saito, K. Akachi, G. Miyamori, T. Isozumi, K. Kaneko, and H. Hirukawa. Distributed control system of humanoid robots based on real-time ethernet. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006.

[6] S.-K Kweon, M.-G Cho, and K.G. Shin. Soft real-time communication over ethernet with adaptive traffic smoothing. *IEEE Transactions on Parallel and Distributed Systems*, 15(10):946–959, 2004.

[7] J.P. Lehoczky, L. Sha, and J.K. Strosnider. Enhanced aperiodic responsiveness in hard real-time environments. In *Proc. of the IEEE Real-Time Systems Symposium*, pages 261–270, 1987.

[8] J. Leung and J. Whitehead. On the complexity of fixed-priority scheduling of periodic real-time tasks. *Performance Evaluation, Elsevier Science*, 22:237–250, 1982.

[9] C.L. Liu and J.W. Layland. Scheduling algorithms for multiprogramming in a hard-real-time environ ment. *Journal of the ACM*, 20(1):46–61, 1973.

[10] J. Loeser and H. Haertig. Low-latency hard real-time communication over switched ethernet. In *Proc. of the Euromicro Conference on Real-Time Systems*, pages 13–22, 2004.

[11] P. Pedreiras, P. Gai, L. Almeida, and G.C. Buttazzo. Ftt-ethernet: A flexible real-time communication protocol that supports dynamic qos management on ethernet-based systems. *IEEE Transactions on Industrial Informatics*, 1(3):162–172, 2005.

[12] B. Sprunt, L. Sha, and J.P. Lehoczky. Aperiodic task scheduling for hard real-time systems. *Real-Time Systems*, 1(1):27–60, 1989.

[13] J.K. Strosnider, J.P. Lehoczky, and L. Sha. The deferrable server algorithm for enhanced aperiodic responsiveness in hard real-time environments. *IEEE Transactions on Computers*, 44(1):73–91, 1995.

[14] S. Varadarajan and T. Chiueh. Ethereal: A host-transparent real-time fast ethernet. In *Proc. of the IEEE International Conference on Network Protocols*, pages 12–21, 1998.

[15] N. Yamasaki. Responsive processor for parallel/distributed real-time control. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1238–1244, 2001.