A Smart Message-scheduling Scheme for Arbitrary Topology PROFINET IRT Networks Applicable to Shipboard Real-time Communications

Jinbo Sim[1] and Jooyoung Son[2]*

[1]*Zinnos Inc., 1125-18 Dongsam Dong, Yeongdo Gu, Busan, S. Korea*
[2]*Div. of IT Engineering, Korea Maritime University,*
Dongsam Dong, Yeongdo Gu, Busan, S. Korea
[1]*jbsim@zinnos.com,* [2]*mmlab@hhu.ac.kr*

Abstract

A message scheduling that determines a sending order in real-time networked control systems should make the cycle time as short as possible. Minimizing the scheduling time itself in dynamic communication environments is also crucial for improving the performance of the systems. The proposed smart message-scheduling scheme for real-time Ethernet PROFINET IRT networks can obtain the optimal cycle time while substantially reducing the scheduling time when applied to communication networks in vessels with various messages and update times. Unlike previous techniques, the proposed scheme has a novel feature that can be applied to networks with arbitrary topology.

Keywords: *PROFINET IRT, arbitrary topology, real-time shipboard networks, smart message scheduling*

1. Introduction

Communications among devices in networked control systems typically require low latency, high update rate, and high throughput. Moreover, data transfer deadline and time determinism must be satisfied in real-time communications. For a long period of time, the serial and fieldbus technologies have been meeting the communication needs in industrial environments. However, due to problems such as slow data rates and interoperability with conventional data networks, Ethernet-oriented solutions have been sought after. SIO and CAN based networks adopted in the shipboard communication situation has similar problems. IEC61162-2 (NMEA0183) is the standard for the former and IEC61162-3 (NMEA2000) is for the latter. IEC61162-450 is also defined as an Ethernet-based standard for shipboard communications.

Ethernet has begun to be used in industrial fields, but it falls short of satisfying the requirements of real-time communications. The transmission delay becomes longer and unpredictable due to collisions that may occur in the IEEE 802.3 CSMA/CD MAC protocol for Ethernet. To solve this problem, switched Ethernets were adopted to narrow the collision domain down to a single node connected to each port. However, when multiple packets go to the same output port in a switch at the same time, the non-deterministic queue delay (node delay) takes place. The signal collision or switch queue delay in Ethernet makes real-time communication difficult. Real-time Ethernets (RTEs) have been developed to overcome the problem [1]. The RTEs are divided into three protocol classes depending upon the locations where the real-time features are implemented (Figure 1).

The class 1 RTE uses the original Ethernet hardware and MAC, as well as the Internet protocol, TCP/IP. The real-time functionalities are implemented at the application layer.

* Corresponding author: Jooyoung Son

Its advantage is an easy interaction with the Internet, but it has non-deterministic delays [3]. A typical example is Modbus/IDA derived from the de facto standard of fieldbus, MODBUS with the Real-Time Publisher Subscriber protocol.

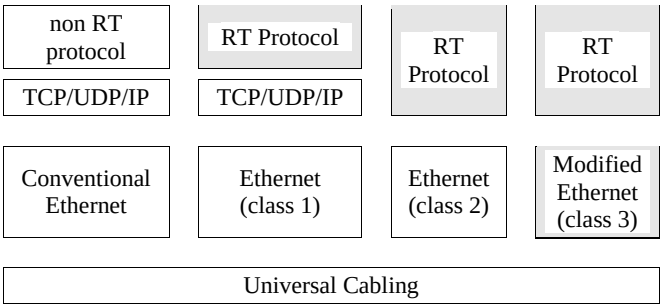

Figure 1. RTE Implementation Structures

The class 2 RTE is also implemented above the existing Ethernet MAC and TCP/IP protocol stack. However, real-time traffics skip the TCP/IP and are passed directly to the application layer. PROFINET RT is an example of this class, which minimizes the queue delay using IEEE 802.1p priority and which in turn increases the temporal determinism.

The class 3 RTE, with the most stringent real-time performance, is implemented in the MAC. PROFINET IRT (Isochronous Real-time) is an example of this class. Here, time is divided into three phases, each of which is assigned to IRT, RT, and non-RT tasks, respectively. In the first phase, PROFINET IRT traffics are scheduled. The frames are switched not by addresses in headers but by the pre-allocated time slot in each device in order to reduce the node delay. In the other two phases, the frames are exchanged in accordance with an address of the header information as usual. PROFINET devices are synchronized by the IEEE 1588 mechanism. In order to reduce the interference caused by the non-RT communication, the higher priority is given to real-time communications.

The SIO and CAN-based networks in ships are implemented based upon IEC61162 (NMEA0183) and IEC61162-3 (NMEA2000) standards, respectively. Ethernet-based communication standard IEC61162-450 was recently enacted, in which, however, no real-time communication technology has been specified yet. In the present paper, the PROFINET network modified to fit various shipboard communication environments is proposed as an alternative.

The communicating-devices in ships generate a number of messages that would vary in terms of length and time depending on situations. The Automatic Identification System (AIS) equipment is one of typical examples [2]. AIS broadcasts messages (i.e. identifier and position information of a ship) periodically to avoid distress and collision on the sea. The periods may dynamically change depending on the speed of ships. Length varies depending on the message type (Table 1). A one-to-one communication between ships is also available.

Table 1. Some AIS Message Types and their Lengths

Type #	Type title	Length (in bits)
01	Position Report Class A	168
05	Static and Voyage Related Data	424
06	Binary Addressed Message	Variable <= 1008
15	Interrogation	Variable 88-160
21	Aid-to-Navigation Report	Variable 272-360
26	Multiple Slot Binary Message	Variable 60-1064
27	Position Report for Long-Range Application	95

Inside shipboard networks in constantly changing environments with various data sizes and data update times, message scheduling tasks are more frequently carried out. In such

cases, scheduling results should be optimal and also the scheduling computation time itself should be minimized. The smart message scheduling scheme is proposed in this paper as a way to satisfy the above conditions.

2. Previous Works

PROFINET IRT networks divide the transmission time into three phases. One of the phases is used only to send isochronous real-time data frames. Controllers in the networks should reduce the cycle time of IRT to a minimum so that more time allowed for real-time and non real-time communications during the other phases. A cycle time is defined as the time spent on one-time data exchange between controller and all connected equipment. In PROFINET IRT networks, messages should be transmitted in the order that the cycle time is to be minimized [3]. The message scheduling problem in obtaining the minimum cycle time for arbitrary topology networks is known as an NP hard problem.

PROFINET IRT equipment from Siemens is believed to use Lauther algorithm which produces schedules close to the optimal results through finding the shortest path in the network graph [3]. The conflict problem in the upstream communication can be tackled by adapting the algorithm for the graph coloring technique [4]. The scheduling problem is mapped to a Job Shop Problem (JSP) [5] and solved by separating the problem into routing and scheduling. The optimal solution can be obtained by formalizing the problem as Resource Constraint Project Scheduling with Temporal Constraints (RCPS/TC) problem [6]. However, RCPS/TC problem does not take into account the fact that no precedence relation exists among messages, so its solution becomes unnecessarily complex. To make matters worse, it takes excessive computation time as RCPS/TC problem only has time-consuming heuristic algorithms, and this is not suitable for dynamic network environments. The dynamic frame packing (DFP) method is proposed in [7], which can be applied to the tree topology and the upstream communication. There is, however, no solution provided for the situations such as when the total size of messages to be packed is larger than the maximum frame size and when the update times are variable. Here, no algorithm for the downstream communication is proposed either.

3. Smart Message Scheduling

The smart message scheduling scheme is proposed in the present research to minimize the cycle time and its computation time for arbitrary topology PROFINET IRT networks. The scheme works in the following two phases.

3.1. Reconfiguration of Topology to MST

The first step reconfigures the network topology into a minimum spanning tree (MST) in order to eliminate the needs of routing. As a matter of course, the controller in the network becomes the root node in the reconfigured tree. The other devices are converted to intermediate nodes or leaf nodes like the example in Figure 2. Compared to the existing MST algorithms, a simpler algorithm can be applied as the root node election process can be omitted. Lastly, the MST information is passed to each node.

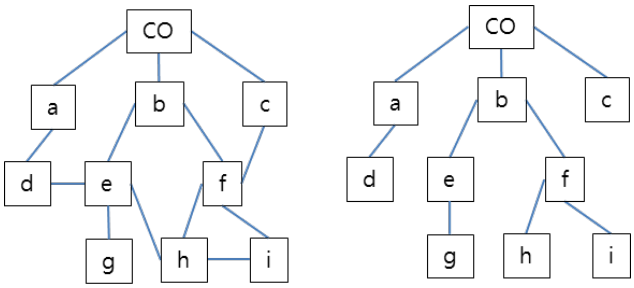

Figure 2. An Example of a Network reconfigured to a MST (CO denotes a controller operating as the root node of the MST)

3.2. Smart Frame Packing

The message transfer unit in PROFINET IRT networks is called a datagram. Each datagram is transmitted in a separate Ethernet frame. Two different schemes for packing multiple datagrams into a frame are proposed in [7] and [8]. The packing schemes can reduce the overhead of the frame header (14 bytes) and trailer (4 bytes) because the size of a datagram is usually smaller than the minimum frame payload size (46bytes). The number of frames to send can also be significantly reduced as a result. The schemes are called frame packing. Figure 3 shows a packed frame structure [8]. However, [7] is limited to the upstream communication, and [8] is only for the line topology. In this paper, Smart Frame Packing algorithm (SFP) is proposed as the second step in the smart message scheduling scheme, which has no restrictions on the topology and the direction of communication.

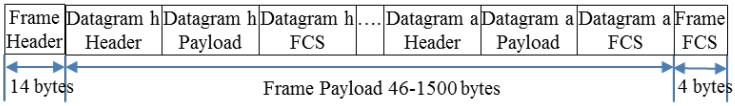

Figure 3. An Example of a Packed Frame Structure Containing Datagram h for Device h, … , and Datagram a for Device a in the Payload Field of the Frame

3.2.1. Downstream Communication

The SFP determines the packing order of datagrams in frames by considering node distances and datagram sizes simultaneously. The node distance is the depth of a device in the reconfigured MST network. The main principle to determine the order is the Farthest Node First method [8]. As a tie-breaker for datagrams of the nodes of the same distance, the Largest Datagram First (LDF) method is developed using the fact that the problem can be directly mapped to a bin packing problem. The LDF method is implemented by adapting a bin packing problem solver, the Non-increasing First Fit (NIFF) algorithm, which is proved as one of best approximation algorithms [9]. This algorithm is much easier and faster than RCPS/TC approach [6] and [7].

```
proc SFPforDownstream_by_Controller(PROFINET)
   if topology of PROFINET is changed then do
      run MST algorithm for the topology of input PROFINET;
      sort the devices in order of nonincreasing depth of devices;
      make the set of devices with the same depth be a group;
      set n_groups as the number of groups;
   end
   make frame frames[]; // sequence of frames to send
   n_frames = 0; // # of frames containing datagrams to send
   for g = 0 to n_groups - 1 do
      gather datagrams to send to devices in group g;
      set n_dgrams as # of datagrams for devices in group g;
      make dgram dgrams[n_dgrams] for datagrams in group g;
      sort dgrams in order of nonincreasing datagram_size;
      NIFFforSFP(dgrams, n_dgrams, frames, &n_frames);
   end
   for f = 0 to n_frames - 1 do
      flood frames[f];
   end
end proc
```

Figure 4. Pseudocode of the SFP for the Downstream Communication Performed by the Controller

Figure 4 shows the procedure in the smart message scheduling performed at the controller for the downstream communication. The dgram denotes a data structure consisting of a datagram size (datagram_size) and the index of destination device for the datagram (datagram_index). The frame contains datagrams packed in a frame, the size of space filled by datagrams, and the number of datagrams. After conversion to a MST topology, devices are sorted in the order of non-increasing depth in the MST. The devices with the same depth are bound into a group. The datagrams in each group are also sorted in the order of non-increasing size. NIFFforSFP() is executed to obtain packed frames for each group. Finally, the packed frames are flooded to devices.

When a packed frame arrives at a device, it is decomposed and the datagram for the device is extracted from it. If at least one of the remaining datagrams in the frame is for one of children of the device in MST, the re-packed frame with the remaining datagrams is flooded to the children. Otherwise, the device discards the re-packed frame.

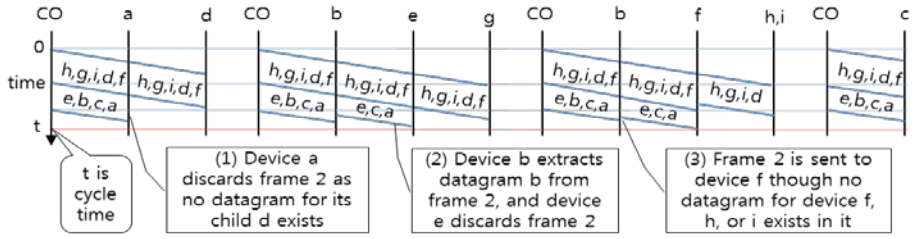

Figure 5. These Space-time Diagrams show an Instance of the Downstream Communication of the Network in Figure 2

Figure 5 shows traffic flow diagrams of the downstream communication for the network in Figure 2. Each diagram depicts the traffic flow happening in each of the branches of the reconfigured MST network in Figure 2. As a result of procedure SFPforDownstream_by_Controller(), the datagrams for devices h, g, i, d, and f are packed into the first frame, and the others are packed into the next frame. As shown in Figure 2, the devices g, h, and i belong to group A, which has the longest distance (3) from the controller (CO). Therefore, the datagrams g, h, and i should go out in the first order. The devices d, e, and f are in group B with the depth of two, and the devices a, b, and c belong

to group C with the depth of one. The controller CO should send the datagrams in the order of A, B, and then C.

The datagrams in each group are sorted in the order of non-increasing size. The sorted order in group A is the datagrams g, h, and i. For group B, the order is the datagrams d, f, and e. For group C, the order is the datagrams b, c, and a. NIFFforSFP() is assumed to derive two frames containing datagrams. In the first frame, the datagrams in group A are packed first, and then the first two of the datagrams in group B are inserted. At this time, it is assumed that the frame 1 cannot accommodate any more datagrams. Then, the remaining datagram e that belongs to group B is packed into the second frame. The datagrams b, c, and a in group C are sequentially packed into frame 2.

The box (1) in Figure 5 explains the following situation: The device a receives the frame 2 which was broadcasted by the controller. The frame 2 has the datagram a, and thus it is extracted from the frame. No datagram for the child d of the device a exists in the remaining datagrams in the frame 2, so the frame 2 is discarded. The box (2) in Figure 5 describes traffic flows occurring in another branch of the MST. The device b extracts the datagram b from the frame 2, and the remaining datagrams are continuously broadcasted to the children (device e and f). After the device e, one of the children of the device b, receives the frame 2, the datagram e is removed from the frame 2. The device e checks whether or not there exist datagrams for the child g in the remaining datagrams in the frame 2. As no datagram exists, the frame 2 is discarded.

The third diagram in Figure 5 shows the phenomenon in which the frame 2 is sent from the device b to the device f even though no datagram exists for the children (device f, h, and i). As mentioned above, frames are broadcasted by nodes including the controller and devices. That makes it possible for nodes to forward frames in a simple manner, and thus node delays can be significantly reduced. Moreover, the broadcasting does not negatively affect the cycle time of the system.

The cycle time of this system is set to the time t at which the transmission of the datagrams (e, c, and a) from the controller CO to the devices (e and f) has just finished.

3.2.2. Adapted NIFF Algorithm for SFP

The procedure NIFFforSFP() in Figure 6 is the adapted NIFF algorithm for the SFP. Each datagram is placed into the first frame that fits it. As datagrams are sorted in the order of non-increasing size, they can be optimally packed in frames. That means that the number of frames to send to devices can be minimized, and so can the cycle times for the system [9]. Much less computation time is needed to get the optimal cycle times as well.

```
proc NIFFforSFP(dgram *dgrams, int n_dgrams,
                frame *frames, int *n_frames)
  for d = 0 to n_dgrams - 1 do // d is index of dgrams[]
    f = 0;
    while (frames[f].filled_space + dgrams[d].datagram_size)
          > MAXFRAME do //look for a frame to fit a datagram
      f++;// frames[f] is insufficient and try the next frame
    end
    frames[f].datagrams[frames[f].n_datagrams++] =
          dgrams[d].datagram_index;
    frames[f].filled_space += dgrams[d].datagram_size;
    if *n_frames < f+1 then do
      *n_frames = f+1; // update the # of filled frames
    end
  end
end proc
```

Figure 6. Pseudocode of the Adapted NIFF Algorithm for the Smart Frame Packing

3.2.3. Upstream Communication

The solution [7] for the conflicts that may occur at branch nodes with two or more children does not consider the case that the total size of conflicted datagrams is larger than the maximum frame size. It does not reorder the conflicted datagrams by their size when they are put into a frame. As a result, the number of packed frames may not be optimal.

The SFP for the upstream communication in Figure 7 solves this problem using NIFFforSFP() as in the downstream communication. In this case, however, the objects to be scheduled are conflicted datagrams in branch devices. Their destination is the same (the controller), which means that the datagrams belong to one and the same group. Hence, NIFFforSFP() is executed only one time in this SFP. As the conflicted datagrams are sorted by size and put into frames in that order, the number of frames to send to the controller can be optimized, and so can the cycle times as well.

```
proc SFPforUpstream_by_Devices(PROFINET)
   gather datagrams from conflicted frames;
   set n_dgrams as # of datagrams gathered;
   make dgram dgrams[n_dgrams] for all datagrams;
   sort dgrams[] in order of nonincreasing datagram_size;
   make frame frames[]; // sequence of frames to send
   n_frames = 0; // # of frames containing datagrams to send
   NIFFforSFP(g_dgrams, n_dgrams, frames, &n_frames);
   for f = 0 to n_frames - 1 do
      send frames[f] to controller;
   end
end proc
```

Figure 7. Pseudocode of the SFP for the Upstream Communication Executed by Branch Devices

4. Conclusions

In the present research, we have proposed an RTE PROFINET IRT protocol adapted for shipboard real-time networks. The smart frame packing (SFP) scheme takes account of arbitrary topology network environments, messages of variable sizes, and various update times. The adapted NIFF algorithm of the bin packing problem proposed herein can be applied in both directions of communications. The NIFF algorithm is very simple and its efficiency has been proven [9]. Thus, the number of frames that are transmitted can be minimized, and consequently, near-optimal cycle times can be derived in much shorter computation time.

When considering future shipboard equipment of high-speed, real-time, and large capacity, shipboard data networks are expected to be replaced by real-time Ethernets. Researches on improving the performance of the PROFINET IRT protocols will continue to cope with these environmental changes.

References

[1] M. Felser, “Real-Time Ethernet – Industrial Prospective”, Proceedings of the IEEE, IEEE Press, New York, vol. 93, no. 6, **(2005)**, pp. 1118-1129.
[2] E. S. Raymond, “AIVDM/AIVDO Protocol Decoding”, http://home.shafe.com/docs/AIVDM.Html, **(2009)**.
[3] J. Jasperneite, M. Schumacher and K. Weber, “Limits of Increasing the Performance of Industrial Ethernet Protocols”, IEEE Conference on Emerging Technologies and Factory Automation, IEEE Press, Greece, **(2007)**, pp. 17-24.
[4] F. Dopatka and R. Wismuller, “A Top-down Approach for Real-time Industrial Ethernet Networks using edge-colouring of conflict-multigraphs”, International Symposium on Power Electronics, Electrical Drives, Automation and Motion, IEEE Press, Taormina, **(2006)**, pp. 883-890.

[5] O. Graeser and O. Niggemann, "Planning of Time Triggered Communication Schedules", Software-intensive Distributed Real-time System, Springer, Boppard, **(2009)**, pp. 21-30.
[6] Z. Hanzalek, P. Burget and P. Sucha, "Profinet IO IRT Message Scheduling With Temporal Constraits", IEEE Transactions on Industrial Informatics, IEEE Press, New York, vol. 6, no. 3, **(2010)**, pp. 369-380.
[7] L. Wisniewski, M. Schumacher and J. Jasperneite, "Fast and Simple Scheduling Algorithm for PROFINET IRT Networks", 9th IEEE International Workshop on Factory Communication Systems, IEEE Press, Lempo, **(2012)**, pp. 141-144.
[8] M. Schumacher, J. Jasperneite and K. Weber, "A New Approach for Increasing the Performance of the Industrial Ethernet System PROFINET", 7th IEEE International Workshop on Factory Communication Systems, IEEE Press, Dresden, **(2008)**, pp. 159-167.
[9] S. Baase, "Computer Algorithms", Addison-Wesley, Philippines, **(1978)**.