On Routing Schemes for Switch-Based In-Vehicle Networks

Shuhui Yang, Wei Li, and Wei Zhao
Department of Computer Science
Rensselaer Polytechnic Institute
Troy, NY 12180
Emails: {yangs6, liw7, zhaow3}@rpi.edu

Abstract— **In this paper, we study in-vehicle digital communication networks. We propose switch-based network architecture and analyze routing schemes in terms of their performance in this new kind of in-vehicle network. We are able to derive conditions under which a network with a proper routing scheme can meet criteria of delivery guarantee, loop-less routing, link-capability limitation, and real-time constraints. We show that these conditions can be easily satisfied in existing systems. Hence, network solutions we discover can be readily utilized in a wide range of practical automotive systems.**

Keywords— bus network, in-vehicle network, routing scheme, switch network.

I. INTRODUCTION

In this paper, we address issues related to in-vehicle digital network systems. A vehicle is inherently a distributed information system, as the operation of a vehicle depends on the collaborative operations of functional components in different locations [Cho95] [KWMH96] [Kop99] [LH02a]. Before the era of automotive electronic systems, communication in the system was performed by mechanical or hydraulic means. A modern vehicle consists of mechanical systems, (i.e., engines, drive train, and steering), hardware, (i.e., electronic control units, sensors and actuators, and vehicle networks), and software, (i.e., embedded operating systems, middleware, and applications). Digital networking technology has been exploited for in-vehicle communication due to its cost-efficiency, space efficiency, and flexibility [AFH03] [CKMNP07] [CVV05] [Koo02] [LHD99] [NHB05] [NSSW05]. A variety of functions, including lights, wipers, doors/windows, and motor control, have realized digitized control. Network protocols have been developed accordingly, such as LIN [ABD99], CAN [Bos91], TTP/C [TTP99], FlexRay [BE01], and MOST [MOST02]. Our goal is to develop and analyze network architectures and related protocols to support scalable, flexible, and real-time communications in automotive systems.

The "control-by-wire" feature [WNSS04] [Tre02] is being realized in the design of the next generation vehicle thoroughly, for not only general functionalities (e.g. light-by-wire), but also for mission critical ones (e.g. break-by-wire, engine-by-wire). An integrated "x-by-wire" system [WNSS04] is expected to take charge of the communication in the vehicle in a real-time, reliable, fault-tolerant, and scalable manner. The rapid development of vehicle capabilities introduces new challenges and demands for automotive network systems [CVV05] [KHM04] [LHD99] [NHB05] [NSSW05]. For example, the amount of transmitted information will increase significantly with more electronic and digital components deployed in automotive systems (e.g., use of audio and video). In addition, applications such as *Collision Warning and/or Avoidance* and *Intelligent Transportation Systems* (ITS) [MM99] also require the support of novel network [HA04] [KHM03] [Koo02].

Protocols used for current in-vehicle digital systems (e.g., CAN), are mostly based on bus-based networks. The event-triggered mechanism [Alb04] [ASV04] [BB03] [Kop01] is used in protocols such as CAN to provide priority-driven communications. The time-triggered mechanism [AFF99] [Alb04] [Ban99] [FMAPN00] [GN05] [KG94] [KHE00] [Kop01] [LH02b] [OPK05] is also widely used, in which a system-wide time-base is established in order to transmit messages in a synchronous manner. The bus-based network architecture lacks scalability when more electrical components are added to the system. Real-time gateways [Dod01] [TY03] are designed to solve the problem by constructing a heterogeneous in-vehicle network to make the system scale easily, as well as to integrate different protocols. However, the single-node gateway has the inherent drawback of the capability bottleneck that may harm the system performance.

The major contributions of this paper are as follows: First, we propose novel switch-based network architecture. Note that the concept of switch-based networks is adopted for the purpose of flexible and scalable communication. In our system, a network is partitioned into sub-networks. Each of these can be a bus-based network, as in current in-vehicle systems. These bus-based sub-networks are then connected by a backbone that consists of a number of switches. The several unique characteristics of our switch-based network architecture distinguish it from other proposals: Taking advantage of a switch-based backbone, our network can be easily scaled up in terms of bandwidth capabilities and may eliminate a single point of failure if properly configured. As we connect bus-based sub-networks to the backbone, the system is still compatible with existing vehicle control units that usually connect to bus-based networks. The cost of the network can be kept low as the switches in the system require very low hardware complexity.

Then, based on this architecture, we focus our study on message routing schemes that are particularly suitable for these new in-vehicle network systems. While routing schemes have been extensively studied for switch-based networks, we have new challenges here. The messages of in-vehicle networks are usually not addressed by their source and destination addresses, rather by functional types. Each type of messages may have multiple senders and multiple receivers. We evaluate routing schemes in terms of their efficiency and effectiveness for this

type of many-to-many cast messages. We are able to derive conditions under which a network with a proper routing scheme can meet criteria of delivery guarantee, loop-less routing, link-capability limitation, and real-time constraints. We show that these conditions can be easily met in the in-vehicle system design. Hence, the solutions we discover can be readily utilized in real automotive systems.

The remainder of the paper is organized as follows: Section II discusses related work; Section III presents the architecture of the proposed system, the design of the switch, and the network model. Section IV provides detailed results on evaluation of routing schemes. Section V summarizes the paper with a discussion of future work.

II. Related Work

A number of digital in-vehicle network systems such as CAN, LIN, FlexRay and TTP/C have been developed. They all adopt a bus-based architecture but are different in terms of functions, performance and application scope. These networks can be classified into two categories, the event-triggered and the time-triggered networks. CAN and TTP/C are the representatives for each category, respectively. The event-triggered architecture (such as CAN) is designed for asynchronous communications. In CAN, each frame has an identifier standing for its priority. It also uses the method of binary countdown to select the frame with the highest priority for transmission. On the other hand, time-triggered architecture (such as TTP/C, FlexRay) is designed to support synchronous communications. For example, TTP/C is designed to support real-time communications by adopting Time Division Multiple Access (TDMA) scheme. In TDMA scheme, each node has been allocated a fixed time period to utilize the network resources in a round-robin, instead of preemptive manner in event-triggered networks. [Alb04] compares these two types of architecture and details their advantages and drawbacks.

As automotive systems evolve with increasing capabilities, more electronic components are integrated into vehicle electronic systems. This brings remarkable increasing demand on communications in terms of bandwidth, real time, and reliability. Existing in-vehicle networks become the bottleneck of bandwidth and performance due to their bus-based architecture. Work has been done to upgrade the bus-based network architecture. The basic idea has been to partition a network into sub-networks (each of them is still a bus for the sake of compatibility) and then to interconnect the sub-networks. Two types of approaches have been proposed for the interconnection. We briefly discuss them below.

The first is to use a backbone network for interconnection, called the *backbone approach*. That is, the backbone network interconnects all functional sub-networks. Information exchanged among different sub networks is performed by this network as well as message format conversion. In [RSG05], IEEE 1394 networks are used as backbones to link all other networks such as CAN and TTP/C. In [HPZ07] [SJW08], a FlexRay network is used as a backbone network to interconnect several sub systems, which are networked by MOST, CAN and FlexRay as well. [SBF05] uses a bus Ethernet to interconnect several CAN bus.

The second one is called *gateway approach*. In [Dod01] [TY03], it uses a dedicated computer node to interconnect sub-networks. The role of the gateway computer is to transmit messages from one sub-network to another and perform message format conversion, if necessary. The gateways can be implemented at a different layer of OSI model. At the DLL and MAC layer, a gateway is then called a bridge. In [EKP96] and [EKP97] design and performance of CAN-to-CAN bridges have been discussed. In [AST03], [CSS00], [KB02], [MSK07], [PEP04], [PEP05], [PF00], [SB07], [SBF06], [SHL06], and [SLH06], different types of gateways which interconnect different in-vehicle network protocols have been introduced.

The gateway approach is efficient when the number of sub-networks in the system is small. However, it is prone to single point failure and may become a performance bottleneck, and hence suffers scalability. The backbone approach overcomes these problems but needs to be properly configured in order to provide required communication services. We adopt this approach and address how to develop effective routing schemes.

III. Switch-Based Network Architecture

In this section, we introduce a switch-based architecture for in-vehicle networks. We present the motivation, discuss its characteristics, and consider the routing problem.

A. Proposed Network Architecture

As mentioned in the previous section, existing in-vehicle network systems are diverse and correspond to different protocols. However, they typically share a common characteristic, i.e., bus-based topology. Most of existing networks adopt bus topology, implying a broadcast mode for message transmission.

In a bus-based in-vehicle network, nodes are usually called *Electronic Control Units (ECUs)*. An ECU is composed of both hardware and software. Usually, ECUs are directly connected to sensors and/or actuators. According to their different functions, ECUs send or receive different types of messages. In early generations of in-vehicle networks, only ECUs on the same bus communicate to each other.

However, with increasing demand for onboard automotive electronic systems, sub-systems may adopt different in-vehicle networks and information needs to be exchanged among them.

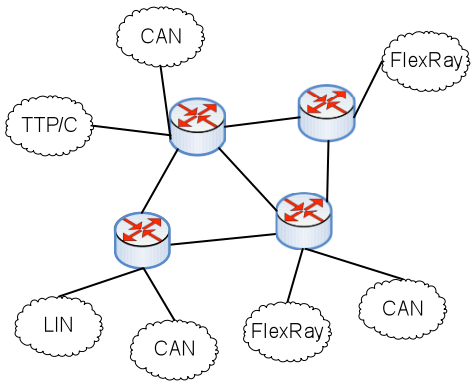


Figure 1. Switch-based Network Architecture

How to connect sub-networks is a critical issue on which we will focus in this paper.

Figure 1 shows the architecture of our proposed network. In Figure 1, there are a number of sub-networks. They may be

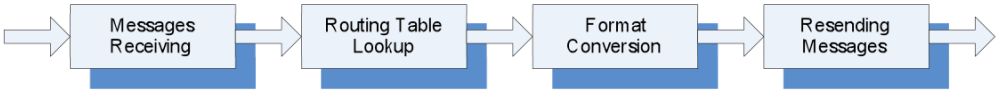


Figure 2. Workflow of a Switch

CAN, FlexRay, TTP/C or other bus-based sub-networks. Each sub-network may contain one or more ECUs (not shown in Figure 1). Sub-networks are then connected by a switch-based backbone. This backbone relays messages among these sub-networks and performs format conversion if necessary.

In comparison with other approaches that we discussed in Section II, this network architecture has the following advantages:

1) Scalability and adaptability: Switch-based networks can be easily adjusted to meet the increased demands and to adapt to different situations. This is a clear advantage over the gateway approach.

2) Reliability: Redundancy can be easily introduced into our network in order to eliminate the single point of failure, hence increasing reliability. In [RSG05], [HPZ07], [SJW08], and [SBF05], a bus is used as a backbone which limits flexibility of topology and hence is difficult to introduce redundancy.

3) Low cost: With proper design, both manufacturing and maintenance costs can be kept at a minimum. We will discuss this issue further in the latter portion of this paper.

B. Design of Switch

Figure 2 presents the main workflow of a switch, which includes message receiving and buffering, routing table look-up, protocol/format conversion[1], and message relay. Based on this workflow model, Figure 3 provides a conceptual design of a switch. A switch consists of the following major components:

1) Computing resources. Computing resources include basic hardware such as processors, memories, storage, etc. These resources must meet the constraints on cost, reliability, and real time.

2) Network Interface Controllers. For different sub-networks, the switches need to use different types of network interface controllers such as CAN controller or FlexRay controllers. The ports of a switch are interfaces to sub-networks or other switches. Messages are received from some ports and are relayed to others. Usually, a port contains two message buffers, i.e. a memory space of the FIFO for incoming and outgoing messages.

3) Routing table. Different from the routing tables in Internet routers, we will consider routing schemes that use static routing tables, which means that routing information will not be changed at runtime and is stored in solid memories, such as EPROM. Section IV will provide detailed discussion on this subject.

[1] An alternative approach is to implement protocol/format conversion at ECUs. For example, in [DF06], [Bev01], [OPK05], [O07], [KBM04], and [DBK03], middleware is used to convert all messages into a standard format via application programming interface (API) for all ECUs. Each approach has its own advantages and problems. Where to implement conversion is not relevant to the results of this paper and hence we will not discuss this issue further.

C. Challenges of Routing Schemes

As in any switch-based network, routing scheme is critical for system efficiency and effectiveness. For efficiency, we mean that a routing scheme should be simple and that it uses the minimum amount of computing and communication resources. For effectiveness, we require that messages be correctly routed to their destinations, meet their deadlines, and avoid endless loops.

A particular challenge in routing messages in our in-vehicle network is in the addressing mode of messages. Messages

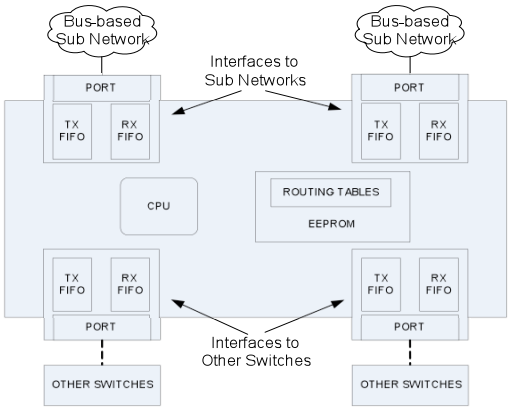


Figure 3. Conceptual Design of a Switch

within in-vehicle networks are not uni-cast, any-cast, or multi-cast, but are, in fact, many-to-many cast. That is, messages are grouped into types. For a type of message, there are typically multiple senders and multiple receivers [TB94]. A message from any sender of a given type must be delivered to all the receivers designated by this type.

For these kinds of many-to-many cast messages, routing schemes utilized in the traditional Internet may not be adoptable to in-vehicle networks due to (the lack of) efficiency, effectiveness, or both.

It is the objective of this paper to explore routing schemes that are suitable for our newly proposed switch-based in-vehicle networks. Section IV will report details of our investigation results.

IV. Analysis of Routing Schemes

A. Overview

In a network, it is its routing scheme to generate routing tables that decide how a message is transmitted from its sender to its receiver. In a switch-based network, routing is usually realized by routing tables embedded in switches. That is, each switch has a routing table. A routing table has a number of entries. Each entry identifies a message and an output port. Once the message arrives with that particular identification, the message is delivered to the output port indicated in the entry for transmission.

As networks become more complicated, entries of a routing table carry more semantics. An entry does not only have to identify an individual message, rather a group of messages. For example, messages may be grouped by a particular destination address and/or a particular sender address. In our case, naturally, messages may be grouped by their types. Furthermore, a message may not have to just be transmitted to one output port. For example, in the case of multi-cast, a message may need to be transmitted over a group of output ports. The same is true for many-to-many cast messages in our in-vehicle networks. Figure 4 shows a general form of routing table.

Group ID	Output Ports
a	1, 4
b	2
…	…
z	2, 3

Figure 4. General Form of Routing Table

Depending on when routing table entries are generated, we can classify routing strategies as follows:

- *Dynamic generation of routing table entries*. That is, the routing table entries are generated in an on-demand fashion at run time. A switch usually caches a number of routing table entries. When a message arrives at the switch, if its entry does not exist, the switch invokes a routing algorithm to dynamically generate the entry. Obviously, this strategy works well for situations in which the network is large and may be dynamically changing. However, this approach requires that switches have powerful computation and communication capacity to coordinate with each other and generate routing table entries in real time.
- *Static generation of routing table entries*. That is, the entries are generated off line and uploaded to the switch at the time when the network is installed. This approach has the advantage of no longer requiring any computation and communication capability for run-time entry generation. It works well in the situations when the complete knowledge of messages is available before network installation. Fortunately, this is the case for our in-vehicle networks. Hence, we adopt this approach in our study.

Now, let us consider group identifiers used in routing table entries. Obviously, if group identifiers contain more information, better decisions can be made on how to route the messages. Hence the system becomes more effective. On the other hand, as more information is contained in the identifiers, the space overhead of the switch will be increased. Thus, the design issue here is to discover routing schemes that can be effective with the minimum amount of information in group identifiers. From this point of view, we will consider three cases of routing table structure:

- Case 1. In our system, the simplest way to define group identifier is to use message type. That is, each type of messages has an entry in the routing table in a switch. Figure 5 (a) shows this kind of routing table. The length of the table is *m* where *m*, is the number of message types.
- Case 2. A better way to define the group identifier of a message is to use both message type and input port ID. Note that our messages are many-to-many cast. Hence, messages of the same type may come to a switch from different input ports. Figure 5 (b) shows this kind of routing table. Thus, the length of a routing table is now in the order of $O(mn)$, where n is the number of input ports a switch may have. We note that n is usually between 4 and 8. Hence, the increase of the length is not too significant.
- Case 3. A more complicated method is to define group identifiers by message type, input port ID, and sender's ID. Figure 5 (c) shows this kind of routing table. The length of a routing table is now in the order of $O(nmk)$,

Group ID	Output Ports
Type 1	1, 3
Type 2	2, 3
…	…
Type *m*	1, 2

(a) Case 1

Group ID		Output Ports
Message Type	Input Port	
Type 1	1	3
Type 1	2	3, 4
Type 1	3	2, 4
Type 2	1	2
Type 2	2	3, 4
Type 2	3	1, 2, 4
…	…	…
...	…	…
Type *m*	1	2
Type *m*	2	1, 3
Type *m*	3	2, 4

(b) Case 2

Group ID			Output Ports
Message Type	Input Port	Sender	
Type 1	1	15	3, 4
Type 1	1	19	2
Type 1	2	11	1,3
Type 1	2	23	4
Type 1	2	24	3, 4
Type 1	3	12	1, 2
Type 1	3	18	1, 4
…	…	…	…
….	…	…	…
Type *m*	1	17	2, 4
Type *m*	1	24	2, 3
Type *m*	1	35	3
Type *m*	2	12	3
Type *m*	2	32	2
Type *m*	3	19	2, 3

(c) Case 3

Figure 5. Routing Tables with Different Group Identifiers

where k is the total ECUs in the system. Thus, the table length here is typically at least one or two orders bigger than that of Cases 1 and 2. Thus, the length increase is very significant.

Our goal in this paper is not to promote a particular routing strategy, rather to evaluate (in terms of efficiency and effectiveness) different routing schemes that use different amounts of information in the routing table. We hope that the results of our investigation will provide guidelines for design and implementation of routing schemes used in practice.

As we are taking a static approach for routing table generation, we no longer need to worry about the run time overhead of computation and communication. Thus, the efficiency of a routing scheme should be measured by its space complexity of routing table.

In order to be considered effective, a routing scheme must meet the following four criteria:

- Delivery guarantee. The routing scheme must guarantee that a message is delivered to all its designated destinations.
- Loop-less routing. In-vehicle networks are mission critical and hence messages should be routed in a way that would never let them be looped in the network.
- Link capacity constraint. Messages transmitted over a link should be no more than the link capacity allowed.
- Real-time constraint. Transmission of messages must meet their deadlines in order to achieve the mission objectives.

Much of the work has been done to verify if a given routing scheme (i.e., routing tables it generated) can meet these criteria for general switch-based network. In Sections IV.B and IV.C, we will develop specific verification methodologies for our in-vehicle networks.

B. *Verification of Delivery-Guarantee and Loop-Less Criteria*

In this subsection, we would like to study routing schemes and analyze under what conditions, they will meet delivery guarantee and loop-less criteria as outlined in Section IV.A.

First, we introduce a concept of optimal scheme.

Definition 1. An optimal routing scheme in terms of criterion X (where X can be delivery guarantee, loop less, or both) is the one that for any network topology and message specification, if there exists a routing scheme that satisfies criterion X, the optimal one also satisfies the criterion.

Next, we consider the three cases of routing table structure discussed in Section IV.A.

Case 1.

Recall that, in this case, the routing table is the simplest and has the shortest length. Only message types are used to identify messages in routing. We have the following results.

Theorem 1. Assume that in a network, routing table structure of Case 1 is adopted. Then, the problem of determining whether there exist routing tables that meet both criteria of delivery guarantee and loop less is NP-complete.

Reader is referred to Appendix for a complete proof. From Theorem 1 and general complexity theory, it is unlikely to develop an optimal routing scheme that has polynomial time complexity for this case. Thus, we will no longer investigate optimal routing schemes for this case. Rather, we will consider some heuristic schemes.

A common heuristic scheme is the *shortest path protocol (SPP)*. With SPP, a path with shortest length is chosen to deliver a message from its source to its destination. SPP is expected to achieve better performance in terms of message delay and resource conservation. The pseudo code of the algorithm is given in Algorithm 1.

Algorithm 1. Shortest Path Protocol (SPP)

```
1. For each message type i,
2.     For each sender node s,
3.         For each receiver node r,
4.             Find the shortest path from s to r,
5.             Fill in the routing table entry on the path
6. End.
```

Obviously, the shortest path protocol meets the criterion of delivery guarantee. Figure 6 shows simulation results of the performance of SPP in terms of meeting the criterion of loop less. Figure 6 reports the data for randomly generated systems that have a configuration as shown in Figure 1. In a simulated system, each subnet contains 10 ECUs. There are 70 types of messages. For each type, two ECUs are randomly chosen as senders and x ECUs are randomly chosen as receivers. x varies from 1 to 70. For each value of x, total 100 systems are generated. We have also simulated other situations with other simulation parameters. The results are similar and will not be reported here due to the space limitation.

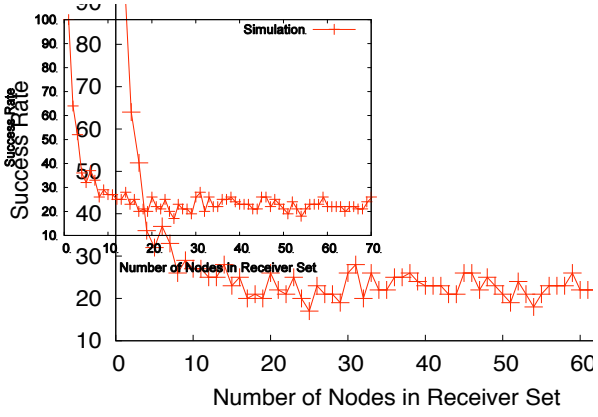
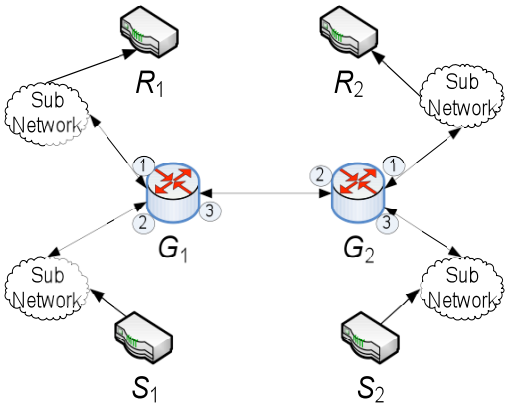


Figure 6. Performance of the Shortest Path Protocol

We measure the performance by *Success Rate*, which is defined as the ratio of the number of generated systems that meet the loop-less criterion vs. the total number of systems generated. From Figure 6, we observe that when the number of receivers increases, the success rate decreases. Especially, when x, the number of receivers for a type of messages, exceeds 10, the success rate reduces to between 10% and 30%. We believe that this kind of performance is unsatisfactory and would not recommend the shortest path scheme for this case.

An obvious reason that SPP fails is that SPP is simple and has no ability to detect and break cycles that may occur due to the routing tables it generates. Fundamentally, we also believe that the failure in Case 1 is due to the fact that too little information is used in routing (i.e., by the group identifiers) and hence any routing scheme would have difficulty to meet the loop-less criterion. Consider the following example. It shows that for a simple network like the one shown in Figure 7, there does not exist any routing scheme that can meet the loop-less criterion in Case 1.



(a) Network Configuration

Group ID	Output Ports
Type 1	1, 3

(b) Routing Table at G_1

Group ID	Output Ports
Type 1	1, 2

(c) Routing Table at G_2

Figure 7. An Example for Case 1

Example 1. Figure 7 (a) shows a network with two switches i.e., G_1 and G_2. There is one type of messages that has senders of S_1 and S_2 and receivers of R_1 and R_2. In order to route messages from S_1 to R_1 and R_2, the routing table in G_1 must be in the form shown in Figure 7 (b). Similarly, in order for messages generated by S_2 to be routed to R_1 and R_2, the routing table in G_2 must have a form as shown in Figure 7 (c). That is, it is necessary to configure the routing tables as shown in Figures 7 (b) and (c) in order to meet delivery guarantee criterion.

Now, when a message generated by S_1 arrives at G_1, it is routed to R_1 and G_2. Then, when the message arrives at G_2, it will be routed to R_2 and G_1. Thus, the message will endlessly loop between G_1 and G_2, failing to meet the loop-less criterion.

In brief, while systems in Case 1 will have the least space complexity, it may be difficult to develop good heuristic routing schemes due to the insufficient information in the routing table. This motivates us to investigate other cases where more information is used.

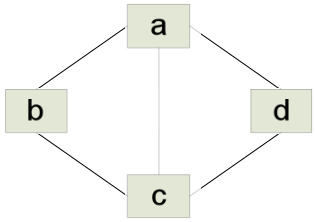
Case 2.

Recall that, in this case, the routing table uses more information than that of Case 1. Both message types and input port IDs are used to identify messages in routing. We have the following results.

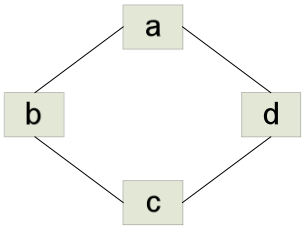
Theorem 2. Assume that in a network, routing table structure of Case 2 is adopted. Then, the problem of determining whether there exist routing tables that meet both criteria of delivery guarantee and loop less is NP-complete.

Reader is referred to Appendix for a complete proof. Once again, from this theorem and general complexity theory, it is unlikely to develop an optimal routing scheme that has polynomial time complexity for this case. Thus, we should consider some heuristic schemes.

Again, the candidate of heuristic scheme is the shortest path protocol (SPP). Though this scheme meets the delivery guarantee criterion easily, as analyzed in Case 1, it may not be able to meet the loop-less criterion due to its simplicity. Our goal here is to identify certain topological condition under which SPP can always meet the loop-less criterion. It is our hope that the topological condition is not too constrained and can be easily met in practice. We introduce some notations first.



(a) Switch Graph of the System in Figure 1



(b) Another Example of Switch Graph

Figure 8. Switch Graphs

Definition 2. *Switch graph* $G = (V, E)$ of an in-vehicle network system is a graph containing only switch nodes and links between switches, where V is the set of all nodes and E is the set of all links.

Figure 8 (a) shows the switch graph of the system in Figure 1. Figure 8 (b) shows another example of switch graph. Note that the switch graph corresponds to the system backbone.

For any two nodes v_i and $v_j \in V$, let V_{ij} denote the set of nodes on the shortest path between v_i and v_j in the switch graph.

Definition 3. A subgraph $G'=(V', E')$ of a switch graph G is *shortest path completed* if for any two nodes v_i and $v_j \in V'$,

1) $V_{ij} \subseteq V'$, and

2) if $(v_i, v_j) \in E$, $(v_i, v_j) \in E'$.

Definition 4. A SPP-routing *ring* of G is a shortest path completed subgraph of G. The *SPP-routing ring* has a ring topology. The number of the nodes on the ring is its *ring length*.

In the switch graph shown in Figure 8 (b), the entire switch graph is a SPP-routing ring with length of 4.

In Figure 8 (a), {*a*, *b*, *d*} and edges among them form a SPP-routing ring with length of 3. Also, {*b*, *d*, *c*} and edges among them form another SPP-routing ring with length 3.

However, in Figure 8 (a), {*a*, *b*, *c*, *d*} with edges {(*a*, *b*), (*b*, *c*), (*c*, *d*), (*d*, *a*)} do not form a SPP-routing ring since the shortest path from *b* to *d* is not included in it and it is not shortest path completed. Thus, our readers should not be confused between a ring and a SPP-routing ring. A SPP-routing ring is a ring sub-graph in a graph, but a ring sub-graph may not necessarily be a SPP-routing ring.

Theorem 3. Assume that in a network, routing table structure of Case 2 is adopted. If the maximum length of its SPP-routing rings of the switch graph is no more than 3, the shortest path protocol is optimal in terms of meeting both delivery guarantee and loop-less criteria.

Reader is referred to Appendix for the complete proof. Thus, if in a system, its maximum length of SPP-routing rings is no more than 3, the shortest path scheme will be truly effective. The question is if this condition can be easily met. We argue this is the case in practice. This is because in-vehicle networks are usually expected to have a small number (say, less than 6) of switches and they are usually well connected. Consequently, their switch graphs are unlikely to contain a large SPP-routing ring. For example, it can be easily proved that the switch graph in Figure 8 (a) meets this condition.

Case 3.

Recall that, in this case, the routing table uses more information than that of both Cases 1 and 2. Message types, input port IDs, and source node identifications are all used to identify messages in routing. We have the following result.

Theorem 4. Assume that in a network, routing table structure of Case 3 is adopted. Then the shortest path routing scheme is optimal in terms of meeting both criteria of delivery guarantee and loop-less criteria for any network topology.

Reader is referred to Appendix for the complete proof. Thus, for the systems of this case, the shortest path routing scheme is recommended.

We now like to make some remarks to conclude Section IV.B. We analyze the performance of routing schemes in terms of delivery guarantee and loop-less routing in the three cases of routing table structures. Generally speaking, the criterion of delivery guarantee is easy to realize, and the loop-less criterion has been a focus for the discussion here.

As discussed in the above three cases, more information contained in the group identifiers is used for routing, more effective a routing scheme can be.

- The length of the routing tables in Case 1 is the shortest. However, as we have shown, the general problem is NP hard and the shortest path protocol is virtually ineffective.
- The length of the routing tables in Case 2 is the second shortest. While the general problem is still NP hard, the shortest path protocol is optimal as long as network topology meets certain condition. We argue that in in-vehicle networks, this condition can be easily met. Thus, the shortest path scheme is effective in the domain of in-vehicle networks for this case.
- The length of the routing tables in Case 3 is the longest. The problem is no longer NP hard and the shortest path protocol is the optimal one. The problem is that the routing tables used here may be significantly larger than that in Cases 1 and 2, making it costly to be utilized in practical system. Note that in automobile industry, cost is a sensitive issue. While a router in the Internet may cost thousands of dollars, switches in the in-vehicle networks we consider here are expected to have a cost in the order of ten dollars or less.

The three different routing table structures present the different tradeoffs between hardware/software cost and routing performance and provide with designer multiple options.

C. Verification of Link-Capability and Real-Time Criteria

To verify if the real-time requirement can be met is a challenging topic and usually requires a study of different methodology from routing schemes. However, recently a new method called *utilization-based schedulability testing* [LL73] [LWF96] [SAA04] [WLZ05] has been developed that significantly reduces the complexity of the problem. With this method, one should model network, traffic and real-time requirement and derive a link utilization bound (say, α). Then, the verification of real-time requirement will become simple: if the utilization of payload traffic on a link is less than α percent of the link capacity, real-time requirement is met. In other words, if we replace link capacity B by $B' = \alpha B$, then the verification of meeting real-time requirement is actually reduced to verification of link capacity (in terms of B'). Thus, once α can be properly derived, we no longer need to develop an independent verification method for meeting real-time requirement. We would like to refer readers to [ACZD92], [ACZD94], [CXLBZ00], and [WXBZ04] for methodologies of deriving α. In the rest of this subsection, we will focus on verification methods of link capacity.

Let us assume that the link capacity of the system is B.[2] There are m message types in the system, and for message type i, its bandwidth requirement is b_i. Let d be the longest path of any message in the system. The following theorem provides a sufficient condition under which the link capacity criterion will be satisfied.

Theorem 5. The link capacity criterion is met if

$$\sum_{i=1}^{m} d \times b_i \le B \qquad (1)$$

Proof: As the length of the longest path is no more than d, there are at most $d+1$ switches along any path, denoted them as $\{v_1, v_2, \ldots, v_{d+1}\}$. In the worst case, every switch along the path except v_{d+1} relays messages of type i towards v_{d+1}. Then the cumulated bandwidth on the last link, v_d to v_{d+1}, is d x b_i. In the worst case, all the m types of messages behave like this. Therefore, the summation of bandwidth requirement on the last link is $\sum_{i=1}^{m} d \times b_i$ which must be no more than B in order to meet the link capacity criterion. ■

[2] As discussed above, B should be normalized by α if real-time requirement is of concern.

With this theorem, we can derive a simpler verification condition for the shortest path scheme.

Corollary 1. Assume that the shortest path scheme is used and that d^* is the diameter of the switch graph. Then, the link capacity criterion is met if

$$\sum_{i=1}^{m} d^* \times b_i \le B \qquad (2)$$

The proof of this corollary is straightforward. Depending on the routing scheme, (1) or (2) can be used to verify if the link capacity criterion can be met.

V. FINAL REMARKS

In this paper, we propose the switch-based network architecture for an in-vehicle digital communication system. This network architecture is compatible with most of the existing in-vehicle systems and aims at providing flexibility, scalability, reliability, and cost efficiency. Due to the performance requirements and the specific communication mode of the in-vehicle systems, the major challenge of the proposed solution is the development of routing schemes. We analyze routing schemes with different routing table structures. We propose four criteria of delivery guarantee, loop less, link-capacity and real time and use them to evaluate routing schemes in the proposed network. In particular, we discover that the shortest path method will perform well in a wide range of networks used for in-vehicle communication. In general, the results of our work clearly reveal performance tradeoffs among different routing schemes and provide guidelines for network design in practice.

The work reported in this paper is fundamental but preliminary, and many extensions are possible. In our proposed switch-based in-vehicle system, the switch design in both hardware and software is vital to the effectiveness of the entire system. One of our ongoing efforts is to design and prototype hardware and software that realize the proposed low-cost switches. Another interesting extension is to explore the capability of fault tolerance for the switch-based networks. With the introduction of fault tolerance criteria, more design spaces need to be explored. For example, one may choose to provide routing schemes with built-in fault tolerance capability (i.e., transmitting redundant copies of a message in the network) or to consider schemes with dynamic fault detection and reconfiguration (of routing tables). These alternative approaches are expected to offer various performance tradeoffs and are certainly worth a comprehensive investigation.

REFERENCES

[ABD99] Audi AG, BMW AG, Daimler Chrysler AG, Motorola Inc. Volcano Communication Technologies AB, Volkswagen AG, and Volvo Car Corporation, "LIN specification and LIN press announcement," SAE World Congress Detroit, http://www.lin-subbus.org, 1999.

[ACZD92] G. Agrawal, B. Chen, W. Zhao, and S. Davari, "Guaranteeing Synchronous Message Deadlines with the Timed Token Protocol," in Proceedings of IEEE International Conference on Distributed Computing Systems, June 1992.

[ACZD94] G. Agrawal, B. Chen, W. Zhao, and S. Davari, "Guaranteeing Synchronous Message Deadlines with the Timed Token Medium Access Control Protocol," in IEEE Transactions on Computers, vol. 43, no. 3, pp. 327-339, March 1994.

[AFF99] L. Almeida, J.A. Fonseca, and P. Fonseca, "A Flexible Time-Triggered Communication System Based on the Controller Area Network: Experimental Results," in Proceedings of International Symposium on Fieldbus Technology, 1999.

[AFH03] J. Axelsson, J. Fro¨berg, H. Hansson, C. Norstro¨m, K. Sandstro¨m, and B. Villing, "A comparative case study of distributed network architectures for different automotive applications," Technical Report, MRTC, 2003.

[Alb04] A. Albert, "Comparison of event-triggered and time-triggered concepts with regards to distributed control systems," in Proceedings of Embedded World Conference, pp. 235–252, 2004.

[AST03] A. Albert, R. Strasser, and A. Tr¨achtler, "Migration from CAN to TTCAN for a Distributed Control System," in Proceedings of 9th international CAN in Automation Conference, vol.5, pp. 9-16, 2003.

[ASV04] F. Ataide, M.M. Santos, and F. Vasques, "A comparison of the communication impact in CAN and TTP/C networks when supporting steer-by-wire systems," in Proceedings of IEEE International Conference on Industrial Technology, vol.2, pp. 1078-1083, Dec. 2004.

[Ban99] R. Bannatyne, "Time Triggered Protocol: TTP/C," Embedded Systems Programming, pp. 76-86, Mar. 1999.

[BB03] I. Broster and A. Burns, "An analysable bus-guardian for event-triggered communication," in Proceedings of 24th IEEE Real-Time Systems Symposium, pp. 410-419, Dec. 2003.

[BE01] J. Berwanger, C. Ebner, and et al., "FlexRay--The Communication System for Advanced Automotive Control Systems," in SAE World Congress, Society of Automotive Engineers Press, no. 2001-01-0676, Apr. 2001.

[Bev01] M. Beveridge, "Jini on the control area network (can): a case study in portability failure," Master thesis, Carnegie Mellon University, 2001.

[Bos91] Robert Bosch, CAN specification 2.0, Parts A and B, Sep. 1991.

[Cho95] E. G. Chowanietz, "Automobile Electronics in the 1990s. Part 1: Powertrain Electronics," in Journal of Electronics and Communication Engineering, vol. 7, no. 1, pp. 23- 36, 1995.

[CKMNP07] J.A. Cook, I.V. Kolmanovsky, D. McNamara, E.C. Nelson, and K.V. Prasad, "Control, Computing and Communications: Technologies for the Twenty-First Century Model T," in Proceedings of the IEEE , vol.95, no.2, pp.334-355, Feb. 2007.

[CSS00] P. Castelpietra, F. Simonot-Lion, Y.-Q. Song, and M. Attia, "Performance Evaluation of a Multiple Networked in-Vehicle Embedded Architecture," in Proceedings of WFCS'2000, 2000.

[CVV05] G. Cena, A. Valenzano, and S. Vitturi, "Advances in automotive digital communications," in Computer Standards & Interfaces, vol. 27, no. 6, pp. 665-678, June 2005.

[CXLBZ00] B. Choi, D. Xuan, C. Li, R. Bettati, and W. Zhao, "Scalable QoS Guaranteed Communication Services for Real-Time Applications," in Proceedings of IEEE International Conference on Distributed Computing Systems, Apr. 2000.

[DBK03] M. Ditze, R. Bernhardi, G. Kamper, and P.Altenbernd, "Porting the Internet Protocol to the Controller Area Network," in RTLIA, 2003.

[DF06] M. Dinkel and D. Fengler, "Unified Communication in Heterogeneous Automotive Control Systems," in Proceedings of the 3rd International Workshop on Intelligent Transportation, Mar. 2006.

[Dod01] D.S. Dodge, "Gateways - 101," in the Proceedings of IEEE Military Communications Conference, vol.1, pp. 532-538, 2001.

[EKP96] H. Ekiz, A. Kutlu, E.T. Powner, "Design and implementation of a CAN/CAN bridge," in Proceedings of Second International Symposium on Parallel Architectures, Algorithms, and Networks, pp.507-513, 12-14 Jun. 1996.

[EKP97] H. Ekiz, A. Kutlu, and E.T. Powner, "Implementation of CAN/CAN bridges in distributed environments and performance analysis of bridged CAN systems using SAE benchmark," in Proceedings of IEEE Southeastcon '97, pp.185-187, Apr. 1997.

[GN05] B. Gaujal and N. Navet, "Maximizing the Robustness of TDMA Networks with Applications to TTP/C," in Real-Time System, vol. 31, no. 1-3, pp. 5-31, Dec. 2005.

[FMAPN00] J. Fonseca, E. Martins, L. Almeida, P. Pedreiras, and P. Neves, "Flexible Time-Triggered Protocol for CAN – New Scheduling and Dispatching Solutions," in Proceedings of 7th International CAN Conference, Oct. 2000.

[HA04] P. F. Hokayem and C. T. Abdallah, "Inherent Issues in Networked Control Systems: A Survey," in Proceedings of 2004 American Control Conference, pp. 4897-4902, 2004.

[HPZ07] M. Horauer, O. Praprotnik, M. Zauner, R. Holler, and P. Milbredt, "A Test Tool for FlexRay-based Embedded Systems," in International Symposium on Industrial Embedded Systems, pp. 349-352, Jul. 2007.

[Kan92] V. Kann, “On the Approximability of NP-complete Optimization Problems,” PhD thesis, Department of Numerical Analysis and Computing Science, Royal Institute of Technology, 1992.

[KB02] J. Kaiser and C. Brudna, “A publisher/subscriber architecture supporting interoperability of the CAN-Bus and the Internet,” in Proceedings of 4th IEEE International Workshop on Factory Communication Systems, pp. 215-222, 2002.

[KBM03] J. Kaiser, C. Brudna, C. Mitidieri, and C. Pereira, “COSMIC: A Middleware for Event-Based Interaction on CAN,” in Proceedings of 9th IEEE International Conference on Emerging Technologies and Factory Automation, 2003.

[KG94] H. Kopetz and G. Grünsteidl, “TTP-A Protocol for Fault-Tolerant Real-Time Systems,” in Computer, vol. 27, no. 1, pp.14-23, Jan 1994.

[KHE00] H. Kopetz, M. Holzmann, and W. Elmenreich, “A Universal Smart Transducer Interface: TTP/A,” In Proceedings of 3rd IEEE International Symposium On Object-oriented Real-time Distributed Computing, Mar. 2000.

[KHM03] N. Kandasamy, J. P. Hayes, and B.T. Murray, “Dependable Communication Synthesis for Distributed Embedded Systems,” in Proceedings of Computer Safety, Reliability and Security Conference, pp. 275–288, 2003.

[KHM04] N. Kandasamy, J.P. Hayes, and B.T. Murray, “Dependable communication synthesis for distributed embedded systems,” in Proceedings of International Conference on Computer Safety, Reliability and Security, Sep. 2003.

[Koo02] P. Koopman, “Critical embedded automotive networks,” in IEEE Micro, vol. 22, no. 4, pp. 14-18, Jul./Aug. 2002.

[Kop01] H. Kopetz, “A Comparison of TTP/C and FlexRay,” Research Report 10/2001, TU Wien, May 2001.

[Kop99] H. Kopetz, “Automotive electronics,” in Proceedings of 11th Euromicro Conference on Real-Time Systems, pp.132-140, 1999.

[KWMH96] J. G. Kassakian, H. Wolf, J. M. Miller, and C. J. Hurton, “Automotive electrical systems circa 2005,” in IEEE Spectrum, vol. 33, no. 8, pp. 22-27, Aug. 1996.

[LAE08] P. Lindgren, S. Aittamaa, and J. Eriksson, “IP over CAN, Transparent Vehicular to Infrastructure Access,” in Proceedings of 5th IEEE Consumer Communications and Networking Conference, pp. 758-759, 2008.

[LH02a] G. Leen and D. Heffernan, “Expanding automotive electronic systems,” in Computer, vol.35, no.1, pp.88-93, Jan. 2002.

[LH02b] G. Leen and D. Heffernan, “TTCAN: A New Time-Triggered Controller Area Network,” in Microprocessors and Microsystems, vol. 26, no. 2, pp. 77–94, 2002.

[LHD99] G. Leen, D. Heffernan, and A. Dunne, “Digital networks in the automotive vehicle,” in Computing & Control Engineering Journal , vol.10, no.6, pp.257-266, Dec. 1999.

[LL73] C. L. Liu and J. W. Layland, “Scheduling algorithms formultiprogramming in a hard-real-time environment,” in Journal of ACM, vol. 20, no. 1, pp. 46–61, Jan. 1973.

[LWF96] J. Liebeherr, D. E. Wrege, and D. Ferrari, “Exact admission control for networks with a bounded delay service,” in IEEE/ACM Transaction on Networking, vol. 4, no. 6, pp. 885–901, 1996.

[MM99] J. McQueen and B. McQueen, *Intelligent Transportation Systems Architectures*, Boston, USA, Artech House Books, 1999.

[MOST] MOST Cooperation, MOST Specification, Nov. 2002.

[MSK07] T. Moon, S. Seo, J. Kim, S. Hwang, and J. Jeon, “Gateway system with diagnostic function for LIN, CAN and FlexRay,” in Proceedings of International Conference on Control, Automation and Systems, pp.2844-2849, Oct. 2007.

[NHB05] T. Nolte, H. Hansson, and L. Bello, “Automotive Communications - Past, Current and Future,” in Proceedings of 10th IEEE International Conference on Emerging Technologies and Factory Automation, 2005.

[NSSW05] N. Navet, Y. Song, F. Simonot-Lion, and C. Wilwert, “Trends in Automotive Communication Systems,” in Proceedings of IEEE, vol.93, no.6, pp.1204-1223, Jun. 2005.

[O07] R. Obermaisser, “A Model-Driven Framework for the Generation of Gateways in Distributed Real-Time Systems,” in Proceedings of 28th IEEE international Real-Time Systems Symposium, 2007.

[OPK05a] R. Obermaisser, P. Peti, and H. Kopetz, “Virtual Gateways in the DECOS Integrated Architecture,” in Proceedings of 19th IEEE International Parallel and Distributed Processing Symposium, Apr. 2005.

[OPK05b] R. Obermaisser, P. Peti, and H. Kopetz, “Virtual networks in an integrated time-triggered architecture,” in Proceedings of 10th IEEE International Workshop on Object-oriented Real-time Dependable Systems, 2005.

[PEP05] P. Pop, P. Eles, and Z. Peng, “Analysis and optimisation of heterogeneous real-time embedded systems,” in IEE Proceedings on Computers and Digital Techniques, vol. 152, no. 2, pp. 130-147, Mar. 2005.

[PEP04]P. Pop, P. Eles, Z. Peng, V. Izosimov, M. Hellring, and O. Bridal, “Design optimization of multi-cluster embedded systems for real-time applications,” in Proceedings on Design, Automation and Test in Europe Conference and Exhibition, vol. 2, no. 16-20, pp. 1028-1033, Feb. 2004.

[RSG05] M. Rabel, A. Schmeiser, H.P. Grobmann, “Ad-hoc in-car networking concept,” in Proceedings of IEEE Intelligent Transportation Systems, pp. 363-368, 13-15 Sept. 2005.

[SAA04]L. Sha, T. Abdelzaher, K. Arzen, A. Cervin, T.Baker, A. Burns, G. Buttazzo, M. Caccamo, J. Lehoczky, and A. K. Mok, "Real Time Scheduling Theory: A Historical Perspective," in Journal of Real-time Systems, col. 28, no. 2-3, 2004.

[SAE93] “Class C Application Requirement Considerations”, SAE Technical Report J2056/1, Jun. 1993.

[SB07] J. Sommer and R. Blind, “Optimized Resource Dimensioning in an embedded CAN-CAN Gateway,” in Proceedings of International Symposium on Industrial Embedded Systems, pp. 55-62, Jul. 2007.

[SBF05] J.-L. Scharbarg, M. Boyer, and C. Fraboul, “CAN-Ethernet architectures for real-time applications,” in Proceedings of 10th IEEE Conference on Emerging Technologies and Factory Automation, vol. 2, pp. 8, 19-22 Sept. 2005.

[SBF06] J. Sommer, L. Burgstahler, and V. Feil, “An Analysis of Automotive Multi-Domain CAN Systems,” in Proceedings of 12th EUNICE Open European Summer School, 2006.

[SHL06] S. Shaheen, D. Heffernan, and G. Leen, “A gateway for time-triggered control networks,” in Microprocessors and Microsystem, vol. 31, no. 1, pp. 38-50, Feb. 2007.

[SJW08] G. Sung, C. Juan, C. Wang, “Bus Guardian Design for automobile networking ecu nodes compliant with FlexRay standards,” in Proceedings of IEEE International Symposium on Consumer Electronics, pp. 1-4, Apr. 2008.

[SLH06] S. Seo, S. Lee, S. Hwang, and J. Jeon, “Development of Network Gateway Between CAN and FlexRay Protocols For ECU Embedded Systems,” in Proceedings of International Joint Conference on SICE-ICASE, pp. 2256-2261, Oct. 2006.

[TB94] K. Tindell and A. Burns, “Guaranteed Message laten-cies for Distributed Safety-Critical Hard Real Time Con-trol Networks,” Technical Report YCS229, Department of Computer Science, University of York, May 1994.

[Tre02] N.R. Trevett, “X-by-Wire, New Technologies for 42V Bus Automobile of the Future,” PhD thesis, South Carolina Honors College, 2002.

[TTP99] TTTech Computertechnik AG, Specification of the TTP/C Protocol, 1999.

[TY03] M. Tokunaga and S. Yoshida, “Advanced Functionality of Vehicle Gateways,” SEI Technical Review, no.55, pp. 55-59, Jan. 2003.

[WNSS04] C. Wilwert, N. Navet, Y.-Q. Song, and F. Simonot-Lion, “Design of automotive X-by-Wire systems,” *The Industrial Communication Technology Handbook*, R. Zurawski, Ed. Boca Raton, FL: CRC, 2004.

[WLZ05] J. Wu, J. Liu, and W. Zhao, “On Schedulability Bounds of Static Priority Schedulers,” in Proceedings of IEEE Real-Time and Embedded Technology and Applications Symposium, Mar. 2005.

[WLZ07] J. Wu, J. Liu, and W. Zhao, “Utilization-Bound Based Schedulability Analysis of Weighted Round Robin Schedulers,” in Proceedings of 28th IEEE Real-Time Systems Symposium, Dec. 2007.

[WXBZ04] S. Wang, D. Xuan, R. Bettati, and W. Zhao, “Providing Absolute Differentiated Services for Real-Time Applications in Static-Priority Scheduling Networks,” in IEEE/ACM Transactions on Networking, vol. 12, no. 2, pp. 326-339, Apr. 2004.

[YLZ08] S. Yang, W. Li, and W. Zhao, “Analysis of Routing Schemes for Switch-Based In-Vehicle Networks,” Technical Report, Department of Computer Science, Rensselaer Polytechnic Institute, Aug. 2008.

Appendix

In this section, we will present the proofs of theorems, and related definitions and lemmas. For proving Theorem 1, we need two lemmas: the first one is to show that the computation complexity of determining whether there exists a feasible message path set is equvelant to that of determining whether there exists a feasible generated DAG; the second one is to show that determination of the existence of a feasible generated DAG is NP-complete. Then, we result in Theorem 1.

For proving Theorem 2, we need three lemmas: Lemma 3 and Lemma 4 are used to prove the set of switch-based graphs and the set of port-based graphs have a one-to-one mapping, and this mapping function is efficiently computable. That is, there exists a polynomial time complexity algorithm for the mapping function; Lemma 5 is to show if a port-based graph has circles then its mapped switch-based graph will also have circles, and vice versa. From these lemmas, we can get the result of Theorem 2.

For Theorem 3, we first show that the worst-case message set is the one with all nodes being senders/receivers in Lemma 6. Then we prove the necessary and sufficient condition for SPP to be feasible in Case 2.

For Theorem 4, we prove the SPP scheme is optimal in Case 3.

Next, we will present the detailed proofs.

Definition 5. A switch-based in-vehicle system is modeled as a directed graph $G_S = (V_S, A)$, where each subnet or switch is a node in V_s, and the connection between switches, and switch and its subnets are directed links in A. An arc $a = (x, y)$ is considered to be directed from x to y. The arc $a' = (y, x)$ is called the arc $a = (x, y)$ inverted. The number of nodes in G_S is denoted as $|V_S|$. The number of arcs in G_S is denoted as $|A|$.

Definition 6. The messages in the system is denoted as $M = \{m_i | m_i = (i, S_i, R_i, b_i), 1 \le i \le m\}$, where i is the message type number, S_i is the sender set, R_i is the receiver set, and b_i is the bandwidth requirement of this message. There are M types of messages in the system.

Note that to make the notation simple, when we study on a certain message type in the following, we use S and R to denote its sender/receiver set.

Definition 7. A directed acyclic graph is a directed graph $G_A = (V_A, A_A)$ with no directed cycles, that is, for any vertex v, there is no nonempty directed path that starts and ends on v. If G_A is a sub graph of G_S and G_A is directed acyclic, we say G_A is a *generated DAG of* G_S.

Definition 8. Given a generated DAG G_{FA} of G_S, we say it is *a feasible generated DAG* if in G_{FA}:

Condition I. For any $v_i \in S$ and $v_j \in R$, there exists a path $p = (v_i, ..., v_j)$.

Definition 9. For a given directed path $p = \{v_1, v_2, \ldots v_n\}$, if $v_1 \in S$ and $v_n \in R$, we call p as *a message path*. If any node in this path will appear only once, we call p as *a feasible message path*.

It is obvious that the path defined in **Condition I**. must be a feasible message path from v_i to v_j.

Definition 10. Given that two paths p_i and p_j have the same node set V and different arc set A_x and A_y. For any two nodes $v_x \in V$ and $v_y \in V$, which are connected by an arc $a_h = (v_x, v_y)$ and $a_h \in A_x$, if for any a_h, there is an arc $a_k = (v_y, v_x) \in A_y$ and $a_k \in A_y$, i.e., $a_h = a_k'$, we say p_j is *an inverted path* of p_i (and vice versa), and we denote the inverted path of p_i as $\overleftarrow{p_i}$.

Definition 11. For a given directed graph G_S, we call a path set P_F as *a feasible message path set* if P_F satisfies: For any $v_i \in S$ and $v_j \in R$, their feasible message path exists and is in P_F.

Definition 12. If a path p_i is a sub graph of path p_j, i.e., $p_i \subseteq p_j$, we say p_i is the sub path of p_j.

Definition 13. Given any two paths p_i and p_j, if there exist sub paths $p_x \subseteq p_i$, $p_y \subseteq p_j$, and p_x, p_y have same start node and end node, then we say p_i and p_j have *an alternative route*.

Definition 14. If a path set $P_{MF} \subseteq P_F$, we call P_{MF} as *an acyclic feasible message path set* if P_{MF} satisfies: For any two feasible message paths p_i and p_j in P_{MF}, p_i and $\overleftarrow{p_j}$ has no alternative route.

Lemma 1. For a given system G_S, for any message type with S and R, if and only if a feasible generated DAG exists, an acyclic feasible message path set for this message exists.

Proof of Lemma 1.

First we prove the necessity of the condition. That is, for a given topology G_S, if an acyclic feasible message path set exists, then a feasible generated DAG must exist.

We assume that graph G_S has an acyclic feasible message path set but has no feasible generated DAG, which means:

1) For directed graph G_S, no generated DAG exist, or

2) All possible generated DAG of G_S do not satisfy **Condition I**.

For the case 1), it is obviously that for any bidirectional graph G_S, there exists at least one generated DAG. So, case 1) does not hold.

Case 2) means a generated DAG that contains at least one path from any sender to any receiver does not exist. According to the definition, feasible message path is a sub graph of G_s. So we first use union operation to joint all paths in P_{MF} together, and we denote the resulting graph as G_S'. It is easy to know that $G_s' \subseteq G_s$, and there are no cycles in G_S'. This also means G_S' is a generated DAG of G_S. Therefore there exists a feasible generated DAG that contains at least one path from any sender to any receiver.

So, in both cases the assumptions are not true, and the necessity of the condition holds.

Next we prove the sufficiency of the condition. That is, for a given topology, if a feasible generated DAG exists, then an acyclic feasible message path set must exist.

We assume that graph G_S has a feasible generated DAG but no acyclic feasible message path set. That is, there is a graph that satisfies the following conditions:

1) There exists node $v_i \in S$ and $v_j \in R$ with no feasible message path between them, or

2) Feasible message path set exists, but no acyclic feasible message path set.

Case 1) means, for node $v_i \in S$ and $v_j \in R$, a) no message path between them exists or, b) their message paths exist, but all these paths have loops. For a), if it holds, then v_i and v_j will not connect to each other. This will conflict to the definition of feasible generated DAG, which requires that any sender and any receiver must have a path. For b), since all possible paths between v_i and v_j have loops, none of these paths will appear in any feasible generated DAG.

Case 2) means there exist two feasible message paths $p_i = (v_0, ..., v_x)$ and $p_j = (v_0', ..., v_y')$, $v_0, v_0' \in S$, $v_x, v'_y \in R$. p_i and p_j satisfy: 1) p_i and $\overleftarrow{p_j}$ has an alternative route, 2) there does not exist another feasible message path p_i' from v_0 to v_x that makes p_i' and $\overleftarrow{p_j}$ has no alternative route (otherwise p_i and p_j will not be chosen to construct acyclic feasible message path set). This means for any feasible message path p_i'' from v_0 to v_x and any feasible message path p_j'' from v'_0 to v'_y, p_i'' and $\overleftarrow{p_j''}$ will have an alternative route. However, from the definition of feasible generated DAG, we know that there must exit a path p_i''' from v_0 to v_x and a path p_j''' from v'_0 to v'_y, which make p_i''' and $\overleftarrow{p_j'''}$ have no alternative route.

Therefore, in both cases the assumption can not hold, and the sufficiency of conditions is proved. ■

Definition 15. A *feedback arc set* (FAS) is a set of edges which, when removed from the graph, leave a DAG. In other words, it is a set containing at least one edge of every cycle in the graph [Kan92].

Lemma 2. Given a directed graph G_S, finding a feasible generated DAG is NP-complete.

Proof of Lemma 2.

Next we convert this problem to *feedback arc set* problem which is a NP-complete problem.

Feedback arc set problem: Given a directed graph $G_S = (V_S, A)$ and a positive integer *k*, whether there exits a subset $A' \subseteq A$ with $|A'| \le k$ such that *A'* contains at least one arc from every directed cycle in G_S. [Kan92]

We assume that there is a polynomial algorithm Λ which can find a feasible generated DAG.

This means that, given G_S, *S* and *R*, we can find a feasible generated DAG in polynomial time with Λ. Since the specification of *S* and *R* is independent to the configuration of G_S, we can use a modified algorithm Λ' to find a feasible generated DAG of any given directed graph G_S by:

1) labeling nodes in G_S with sets *S* and *R*, and then

2) using algorithm Λ to find out a feasible generated DAG.

Because step 1) can be finished in polynomial time, algorithm Λ' can be finished in polynomial time. Then we get:

Given any directed graph G_S, there exists an algorithm Λ', which can find a feasible generated DAG in polynomial time.

According to the definition of feasible generated DAG, there is no cycle in any feasible generated DAG. This means, algorithm Λ' can find out all possible cycles in G_S and choose correct directions of each arcs to avoid cycles. That is, for any possible cycles in G_S, algorithm Λ' must remove one or more arcs in this cycle to break the cycle. This also means that, using algorithm Λ' we can construct an arc set, by removing which the graph G_S will have no cycles. This arc set is the feedback arc set problem defined. However, as we know,

feedback arc set problem has been proved as a NP-complete problem, so our assumption here will not hold. ■

Theorem 1. Assume that in a network, routing table structure of Case 1 is adopted. Then, the problem of determining whether there exist a routing table that meets both criteria of delivery guarantee and loop less is NP-complete.

Proof of Theorem 1.

We only have to prove that determining whether there exists a routing table that meets the criteria of loop less is NP-complete. That is, for any given topology defined in Definition 5 and message set defined in Definition 6, determining if there exists an acyclic feasible message path set is NP-complete.

We assume that we can find an acyclic feasible message path set in polynomial time. That is, for a given system with topology G_S, and an arbitrary message with *S* and *R*, there exists a polynomial algorithm $\overline{\Lambda}$ which can find an acyclic feasible message path set for the system. If we find out an acyclic feasible message path set by $\overline{\Lambda}$, then we would know that the system must have a feasible generated DAG. We can determine whether G_S has a feasible generated DAG in polynomial time. Obviously, this conflicts to the conclusion of Lemma 2. Therefore, the assumption we made is not true and Theorem 1 holds. ■

Definition 16. For a given directed graph $G_I = (V_I, A_I)$ and $|V_I| \geq 3$, if for any two nodes $v_i \in V_I$, $v_j \in V_I$, there is an arc from v_i to v_j, then we call this graph an *internal port graph*. It is obvious that an internal port graph G_I is a complete graph.

Definition 17. Given two internal port graphs G_I and G_I', for any two nodes $v_i \in V_I$ and $v_j \in V_I'$, the arc a_{ij} from v_i to v_j is called *link arc* of G_I and G_I'.

Definition 18. Given a set of internal port graphs $G_{I_0}, G_{I_1}, ..., G_{I_k}$ and a set of link arcs $a_0, a_1, ..., a_l$, G_P is union of the internal port graph set and link arc set, if G_P satisfies followings we say G_P is a *port-based graph*:

1) For any internal port graph, it has at least one link arc with any other internal port graph,

2) For any two internal port graphs, they have only one link arc if they have link arcs.

Definition 19. All link arcs of G_P construct a set A_L, and all nodes of G_P construct a set V_P.

Definition 20. All possible switch-based graphs construct a set G_S^*.

Definition 21. All possible port-based graphs construct a set G_P^*.

From Theorem 1, we know that for a given system with topology G_S, and an arbitrary message with *S* and *R*, there are no polynomial algorithms to find an acyclic feasible message path set for the system.

Then our problem to be solved in Theorem 2 can be equally changed to, for a given system with any topology of G_P, whether there exists a polynomial algorithm to find an acyclic feasible message path set for the system.

To prove it, we first prove that port-based graph set has a one-to-one relation to switch-based graph set.

Definition 22. If there exists a bijection function $f: G_P^* \to G_S^*$, we say f is a *port-based graph mapping function* from G_P^* to G_S^*.

Lemma 3. There exists a bijection function $f: G_P^* \to G_S^*$.

Proof of Lemma 3.

We first construct a f as following:

We use a node mapping function $g: V_P \to V_S$. That is, given a port-based graph G_P, for any of its internal port graph G_I, g will mapping each internal node set V_{I_i} to one and only one node in V_S.

We use a link mapping function $h: A_L \to A_S$, which will mapping each arc in A_L to one and only one arc in A_S. That is, for each link arc a_{L_i}, $h(a_{L_k}) = arc(g(V_I), g(V_I')) = a_{S_k}$, $a_{L_k} = arc(v_i, v_j)$, $v_i \in V_I$ and $v_j \in V_I'$, $a_{S_k} \in A_S$.

Then we use these two functions to construct a new graph based on a port-based graph.

$$f(G_P) = f(V_P, A_P) = \begin{cases} V_S, \forall v_i \in V_S, \exists g(V_{I_i}) = v_i \\ A_S, \forall a_{S_j} \in A_S, \exists h(a_{L_j}) = a_{S_j} \end{cases}$$

Obviously, (V_S, A_S) will construct a graph and it is a switch-based graph defined before.

From the definition of f, we know that given a port-based graph we can generate a unique switch-based graph.

Next, we will give the definition of f^{-1} , i.e., $f^{-1}: G_S{}^* \to G_P{}^*$.

$$f^{-1}(G_S) = f^{-1}(V_S, A_S) = \begin{cases} V_P, \forall V_{I_i} \in V_I{}^*, \exists g^{-1}(v_i) = V_{I_i} \\ A_P, \forall a_{L_j} \in A_L, \exists h^{-1}(a_{S_j}) = a_{L_j} \end{cases}$$

The meaning of function g^{-1} is to mapping a switch to its port set, and all these ports are fully connected. This also means given a switch node, g^{-1} will mapping one and only one port set, i.e., one and only one internal port graph.

The meaning of function h^{-1} is that, for a given arc a_{ij} in A_S , $v_i \in V_{I_i}$, $v_j \in V_{I_j}$ we will replace v_i with a port in $g^{-1}(v_i) = V_{I_i}$ and replace v_j with a port in $g^{-1}(v_j) = V_{I_j}$. This also means, given an arc in switch-based graph, function h^{-1} will mapping one and only one link arc in port-based graph. Then for any given switch-based graph, f^{-1} will mapping it to one and only one port-based graph.

So, for any port-based graph, there exists a bijection function which mapping it to a switch-based graph. ■

Lemma 4. The bijection functions f and f^{-1} are efficiently computable.

Proof of Lemma 4.

An efficiently computable mapping $f: G_P{}^* \to G_S{}^*$ is equal to there exists an algorithm with polynomial complexity which can determine if for any $G_P \in G_P{}^*$ there is one and only one $G_S \in G_S{}^*$ that satisfy $f(G_P) = G_S$.

First we will analyze the complexity of function link mapping function *h*. In function *h*, we need to search all possible internal port graphs, whose complexity will not exceed $O(|G_I{}^*|) \le O(|V_P|)$.

Next, for the node mapping function *g*, for each link arc, we need to find the internal port graphs connected to this arc, whose complexity will be $O(|G_I{}^*| * |V_P|) \le O(|V_P|^2)$.

Therefore the complexity of function f will not exceed $O(|V_P|) + O(|V_P|^2) \le O(|V_P|^2)$, and then function f is efficiently computable. That is, there exists a polynomial algorithm that can transform a port-based graph to one and only one switch-based graph. Similarly, we can prove f^{-1} is also efficiently computable. ■

Lemma 5. A switch-based graph G_S has a circle if and only if $f^{-1}(G_S) = G_P$ has a circle which contains link arcs.

Proof of Lemma 5.

We first prove the sufficiency. That is, if $f^{-1}(G_S) = G_P$ has a circle which contains link arcs, then G_S will contain a circle. According to the definition of f and f^{-1} , if G_P has a circle which contain link arcs, the circle will contain two switches connected. That is, the circle will pass through these two switches infinitely. Therefore, in a switch-based graph, these two switches will be on a circle.

Next, we will prove the necessity. That is, if G_S has a circle, $f^{-1}(G_S) = G_P$ will have a circle which contain link arcs. This means that the circle in G_S includes multiple switches and arcs. According to the definition of f and f^{-1} , in G_P there exists a circle that pass through multiple internal graphs no matter how ports are connected. In addition, because there are at least two switches in this circle, at least one link arc will be contained in this circle. ■

Theorem 2. Assume that in a network, routing table structure of Case 2 is adopted. Then, the problem of determining whether there exist routing tables that meet both criteria of delivery guarantee and loop less is NP-complete.

Proof of Theorem 2.

Similar to the proof of Theorem 1, here we will only have prove that the problem of determining whether there exist routing tables that meet the criteria of loop less is NP-complete. That is, we need to prove that for any given port-based graph, it is NP-complete to determine if there exists an acyclic feasible message path set.

We assume that there is a polynomial complexity algorithm Γ that can determine if a port-based graph has circles and for each circle if it contains link arcs. Then for any switch-based graph G_S , according to Lemma 4, we can use an algorithm with polynomial complexity to generate a port-based graph $f^{-1}(G_S) = G_P$. So, by algorithm Γ we can determine whether G_P has circles and if these circles have link arcs. Therefore, according to Lemma 5, we can also determine whether G_S has circles. That is, we can use polynomial complexity algorithm to determine if a switch-based graph has a circle.

However, according to the result of Theorem 1, finding a circle in switch-based graphs is NP-complete. Our assumption does not hold. Therefore, it is NP-complete to determine whether a port-based graph has a circle. ■

Definition 23. From a given message type *i* with sender set *S* and receiver set *R*, sender switch set $V_{SD} \subseteq V$ is a set including all switch nodes whose attached subnets have senders; receiver switch set $V_{RV} \subseteq V$, is a set including all switch nodes whose attached subnets have receivers.

Definition 24. For a given message type i, and a switch graph G, $NET_i = (V, E, V_{SD}, V_{RV})$. When a routing protocol (RP) satisfies the delivery guarantee and loop-less criteria on G for this message type, we denote as $\boldsymbol{f}(NET_i, RP)=1$; otherwise, $\boldsymbol{f}(NET_i, RP)=0$.

Definition 25. $NET^* = \{NET_i | NET_i = (V, E, V_{SD}, V_{RV})\}$ is the set of NETs corresponding to all possible message types.

Definition 26. A NET_w is the worst case for a switch graph G, if for any NET_i in NET^*, when $f(NET_i, RP)=0$, $f(NET_w, RP)=0$.

Lemma 6. For a switch graph G, $NET_w = (V, E, V_{SD}, V_{RV})$ is the worst case for any RP, which guarantees the delivery.

Proof of Lemma 6.

We use contradiction to prove. If the lemma is not true, then there exists a $NET_i = (V, E, V_{SD}, V_{RV})$, where $V_{SD} \subset V, V_{RV} \subseteq V$ or $V_{SD} \subseteq V, V_{RV} \subset V$. $f(NET_i, RP)=0$, $f(NET_w, RP)=1$. RP is a routing protocol.

$f(NET_w, RP)=1$, and $v_i \in V$ and $v_j \in V$, using RP the message is routed correctly from v_i to v_j in NET_w.

$f(NET_i, RP)=0$. Therefore, there exists a $v_i \in V_{SD}$ and a $v_j \in V_{RV}$, when using RP to route the message from v_i to v_j, since RP guarantees the delivery, the message route forms a routing loop.

Since the sender/receiver switch set of NET_i is a subset of that of NET_w, the output ports of each entry of the routing tables of NET_i is a subset of that of NET_w. Since RP is deterministic, the message generated from v_i to v_j forms a routing loop based on the tables in NET_w. Therefore, $f(NET_w, RP)=0$.

This is contradicted by the assumption. Therefore, NET_w is the worst case. ■

In the following proof, we assume the worst-case message specification, that is, the sender/receiver switch sets are both V.

Theorem 3. Assume that in a network, routing table structure of Case 2 is adopted. If the maximum length of its SPP-routing rings of the switch graph, if exists, is 3, the shortest path protocol (SPP) is optimal in terms of meeting both delivery guarantee and loop-less criteria.

Proof of Theorem 3.

i) First to prove the condition that l is 3 is the necessary condition for SPP to be feasible. We prove if there exists a SPP-routing ring of G that has length larger than 3, SPP can not guarantee loop less.

We use $\{v_1, v_2, \ldots, v_l\}$ to represent this SPP-routing ring, $l>3$. Other than the ports connecting to its own subnets, each switch node, v_i, has two ports connecting to two other switches, v_{i-1} and v_{i+1}. As this is a ring, $v_{l+1} = v_1$.

Prove this by induction. Let $l = 4$, we show the theorem holds.

Without loss of generality, we can assume that the subnet of switch v_1 generates a message. Note that we consider here all switches in both the sender and receiver switch sets (the worst case). Using SPP, v_1 relays the message to v_2 since this is the shortest path from v_1 to v_2. v_2 will then relay the message to v_3 since it is on the shortest path from v_1 to v_3 in this SPP-routing ring.

Consequently, the message will be routed to v_4 and then v_1. When v_1 receives this message from v_4, it will be routed to v_2 again. Because v_1 is on the shortest path from v_4 to v_2.

Hence, message is relayed in the SPP-routing ring with no halt in the sequences of $\{v_1, v_2, v_3, v_4\}$. This is a routing loop and SPP does not satisfy loop less.

Now, we assume that the theorem holds for $l = k$ and show that it will hold for $l = k+1$, (k>=4).

If when $l = k$, the SPP-routing ring forms a routing loop, when $l = k+1$, the message generated from v_1, as in the $l = k$ SPP-routing ring, will be relayed via $v_2, v_3, \ldots,$ to v_k.

Since v_k is on the shortest path from v_{k-1} to v_{k+1} in the SPP-routing ring of $l = k+1$, v_k relays the message to v_{k+1}. v_{k+1} then relays it to v_1 again since it is on the shortest path from v_k to v_1.

Hence, message is relayed in the SPP-routing ring with no halt in the sequences of $\{v_1, v_2, \ldots, v_{k+1}\}$. This is a routing loop and SPP does not satisfy loop less.

ii) Then to prove that l is 3 is the sufficient condition for SPP to satisfy loop less. This is, if SPP does not satisfy the loop less for G, l must be larger than 3.

When SPP does not guarantee the loop less, there exists a routing loop. We use $\{v_1, v_2, \ldots, v_x\}$ to represent the nodes that are included in the routing loop. Next we first prove that $x > 3$.

Case a) When x is 2, v_1 does not relay message from v_2 back to v_2 since it is not on the shortest path from v_2 to v_2. v_2 does not relay message from v_1 back to v_1. Therefore, $\{v_1, v_2\}$ does not form a routing loop.

Case b) When x is 3, v_1 does not relay message from v_2 to v_3 since it is not on the shortest path from v_2 to v_3. v_2 and v_3 are the same. Therefore, $\{v_1, v_2, v_3\}$ does not form a routing loop.

Therefore, we have that $x > 3$.

We now to prove that with a SPP routing loop $\{v_1, v_2, \ldots, v_x\}$ and x is lager than 3 (according to previous result), there exists SPP routing ring with length larger than 3.

We use induction in the following proof.

x is 4, we show that $\{v_1, v_2, \ldots, v_x\}$ forms a SPP-routing ring.

In the routing loop of $\{v_1, v_2, v_3, v_4\}$, v_i relays message from v_{i-1} to v_{i+1}. Using SPP, it means v_i is on the shortest path from v_{i-1} to v_{i+1}. Therefore, there is no direct connection between v_{i-1} and v_{i+1}.

Therefore, there is no direct connection between v_1 and v_3, or between v_2 and v_4. $\{v_1, v_2, v_3, v_4\}$ forms a SPP-routing ring.

Now we assume that when x is k, $\{v_1, v_2, \ldots, v_x\}$ forms a SPP-routing ring. Then we prove that when x is $k+1$, there is a SPP-routing ring.

Since when x is k, $\{v_1, v_2, \ldots, v_k\}$ forms a SPP-routing ring. With v_{k+1} included in the routing loop, v_{k+1} must be on the

shortest path from v_k to v_1. That is to say, there is no direct connection between v_k and v_1.

Since v_1 needs to be on the shortest path from v_{k+1} to v_2, there is no direct connection between v_{k+1} and v_2. This is the same with v_{k+1} and v_{k-1}.

If v_{k+1} has no connection to nodes in $\{v_3, v_2, \ldots, v_{k-2}\}$, $\{v_1, v_2, \ldots, v_{k+1}\}$ forms a SPP routing ring.

Otherwise, we assume that there is a connection between v_{k+1} and v_i, $2<i<k-1$. When v_i gets the message from v_{i-1}, since it is on the shortest path from v_{i-1} to v_{k+1}, it will forward the message to v_{k+1}, too. Then $\{v_1, v_2, \ldots, v_i, v_{k+1}\}$ is also a routing loop, and since x is no larger than k and no smaller than 4 in this case, $\{v_1, v_2, \ldots, v_i, v_{k+1}\}$ forms a SPP-routing ring.

That is to say, when x is $k+1$, either $\{v_1, v_2, \ldots, v_i, v_{k+1}\}$ forms a SPP-routing ring or there exists a SPP-routing ring.

Therefore, there exists a SPP-routing ring. This is contradicted by the assumption that all SPP-routing rings in G have numbers smaller equal to 3.

When a given switch graph has no SPP-routing ring, that is, either there is only one node in G, or G forms a tree structure, it is obvious that SPP satisfies the loop-less criteria. ■

Theorem 4. Assume that in a network, routing table structure of Case 3 is adopted. Then the shortest path routing scheme is optimal in terms of meeting both criteria of delivery guarantee and loop-less criteria for any network topology.

Proof of Theorem 4.

Using SPP, given a message type, for each sender node, there is a subgraph of the switch graph that forms a tree structure rooted at the sender node and with all the receiver nodes as leaves. The routing tables are established based on these tree structures.

When there is only one node in the sender set of the message, one routing tree exists. Obviously, the switches only need to relay message from their parents to the children on the tree. Therefore, the delivery guarantee and loop less are satisfied.

When there is more than one senders of the message, correspondingly, several trees are established in the routing tables. However, since source information is embedded in the message, the switches can distinguish which tree to use to relay the message. Therefore, it is the same situation with one sender node routing. Therefore, delivery is guaranteed and there is no routing loop. ■