Distributed Control for Link Failure Based on Tie-Sets in Information Networks

Kiyoshi Nakayama, Student Member, IEEE, Norihiko Shinomiya, Member, IEEE
and Hitoshi Watanabe, Life Fellow, IEEE
Graduate School of Engineering, Soka University, Tokyo, 192-8477, Japan
Email: {kiyoshi.nakayama, shinomi}@iee.org

Abstract—This study proposes distributed control for link failure based on loops in a network to conduct local management. More local and high-speed recovery is considered to be possible if failure is resolved within some tie-sets. This paper introduces the tie-set concept first, and then describes the distributed algorithms for link failure. Experiments to compare against RSTP are also conducted.

I. INTRODUCTION

With the proliferation of the Internet as well as the large scale networks these days, autonomous distributed architectures that can manage networks more locally and flexibly has become required. In modern networks becoming larger and more complicated, such failure, even for a short time, may cause extensive damage to entire network lines. For this reason, high-speed and local restoration of a network takes priority over everything else to keep reliable communication. Today, the general use of Rapid Spanning Tree Protocol or RSTP [1] to restore network failure has its limitations.

Since RSTP sends messages to the entire network systems when a network failure occurs, it delays the restoration of the network itself. High-speed restoration can be achieved by a ring-based restoration system in which a damaged path is quickly shifted to an alternate path on each ring. For instance, Unidirectional Path Switched Ring (UPSR) [2] or Bidirectional Line Switched Ring (BLSR) [3] is used in a Synchronous Optical Network/Synchronous Digital Hierarchy (SONET/SDH) network. Moreover, Ethernet Automatic Protection Switching or EAPS (4) is utilized in local area networks. However, these schemes can only be applied to ring topological network configurations. Hence, it is required to develop a restoration method that can be applied to complex network configurations as well as make full use of high-speed and reliable restoration based on the concept of rings. Generally, existing networks are constituted redundantly using alternate links to cope with failure. A redundantly constituted network corresponds to a bi-connected graph in graph theory and has loops in it. If failure is restored within some loops, adverse effects are considered to be reduced. As a previous work, there has been a study which focuses on loops in a graph, which are called tie-sets [5]. The study shows graph theoretical nature of underlying loops in networks, and indicates possibilities of conducting optimal local network management based on tie-sets. An overview of fault link avoidance based on tie-sets is also suggested in [6]. However, distributed control for network failure based on tie-sets has not yet been proposed. Therefore we introduce graph theory of the tie-set, and then propose distributed control for link failure.

II. TIE-SETS AND STATE INFORMATION

A. Fundamental System of Circuits and Tie-sets

For a given bi-connected and undirected graph \( G = (V, E) \) with a set of vertices \( V = \{v_1, v_2, ..., v_n\} \) and a set of edges \( E = \{e_1, e_2, ..., e_m\} \), let \( L_i = \{e_1', e_2', ..., e_i'\} \) be a set of edges which constitutes a loop in \( G \). The set of edges \( L_i \) is called a “tie-set” [7]. Let \( T \) and \( T \) be a tree and a coterie of \( G \), respectively, where \( T = E - T \). \( \rho = \rho (G) = |T| \) and \( \mu = \mu (G) = |T| \) are called the rank and the nullity, respectively. A tree \( T \) on a graph \( G = (V, E) \) is a ultranet set of edges which does not include any tie-set. In other words, for \( l \in T, T \cup \{l\} \) includes one tie-set. Focusing on a subgraph \( G_T = (V, T) \) of \( G \) and a edge \( l = (a, b) \in T \), there exists only one elementary path \( P_T \) whose origin is \( b \) and terminal is \( a \) in \( G_T \). Then an elementary circuit which consists of the path \( P_T \) and the edge \( l \) is uniquely determined as follows:

\[
L(l) = \begin{cases} (a, l = (a, b), P_T(b, a)) \\ (a, l, \rho = b, t_1, v_1, \cdots, t_k, v_k = a) \end{cases} \quad (1)
\]

In this way, a circuit determined by an edge \( l = (a, b) \in T \) and a path \( P_T(a, b) \) on \( G_T \) is denoted as a "fundamental circuit". A simple circuit can be expressed by a set of edges, so called a tie-set. A tie-set corresponding to a fundamental circuit regarding \( T \) is denoted as a "fundamental tie-set" regarding \( T \). It is known that \( \mu \) fundamental circuits and tie-sets exist in \( G \), and they are called a "fundamental system of circuits" and a "fundamental system of tie-sets", respectively. A fundamental system of tie-sets covers all vertices and edges in \( G \) as shown in Fig.1.
B. State Information in Each Node

In information networks, let us assume that a tree \( T \) on \( G = (V, E) \) corresponds to communication paths, and a cotree \( \overline{T} \) corresponds to non communication paths. In this paper, links representing communication paths are defined as "tree links" which are expressed by thick lines, while links representing non communication paths are defined as "cotree links" which are expressed by thin lines as shown in Fig.1. For example, tree links and cotree links are \( \{e_2, e_3, e_5, e_7, e_9\} \) and \( \{e_1, e_4, e_6, e_8\} \) in Fig.1, respectively. Each node \( n \) mainly has three types of information as state information as follows:

1) Incident Links: Information of links connected to \( n \).
2) Adjacent Nodes: Information of nodes which are connected through incident links of \( n \).
3) Tie-set Information: Information of fundamental tie-sets to which \( n \) belongs. When a fundamental tie-set \( L_i \ni n \), it is defined that \( n \) belongs to \( L_i \) and has information of \( L_i \).

Here is an example of state information of a node \( c \) in Fig.1. The node \( c \) has information of \( \{e_2, e_3, e_5, e_6\} \) as incident links, \( \{a, b, d, e\} \) as adjacent nodes, and tie-set information of \( \{L_1, L_2, L_3\} \).

C. Algorithm for Recognizing Tie-set Information

Each node has information of fundamental tie-sets to which the node belongs so as to solve any problems within some loops. In order to obtain Tie-set Information, each node executes a distributed algorithm to recognize fundamental tie-sets. A Find Tie-set message, which is used to catch information of a fundamental tie-set, includes information as follows:

- **EdgeTable**: A set of links through which a Find Tie-set message passed
- **NodeTable**: A set of nodes through which a Find Tie-set message passed

If Find Tie-set messages are processed according to the rules below, each node can hold information of fundamental tie-sets. First, each node \( n_o \) creates Find Tie-set messages, and then sends those messages to all adjacent nodes of \( n_o \). When sending a Find Tie-set message to an adjacent node \( n_o \), \( n_o \) adds node information of \( n_o \) to NodeTable, and adds information of a link connected to both \( n_o \) and \( n_o \) to EdgeTable. Let \( n_e \) be a node which receives a Find Tie-set message. After receiving a Find Tie-set message, \( n_e \) executes different procedure by the following cases.

**Case 1**: \( n_e \neq n_o \). In this case, if EdgeTable of the Find Tie-set message includes more than one cotree link, \( n_e \) discards the message. If EdgeTable contains no or one cotree link, \( n_e \) copies the Find Tie-set message and sends the copied message to adjacent nodes which are not included in NodeTable. In case that the adjacent node is \( n_o \), \( n_e \) sends the copied message to \( n_o \) even if \( n_o \in NodeTable \). When sending a copied message to an adjacent node \( n_o \), \( n_e \) adds node information of \( n_o \) to NodeTable, and adds information of a link connected to both \( n_o \) and \( n_o \) to EdgeTable.

**Case 2**: \( n_e = n_o \). In this case, the Find Tie-set message has passed through certain loop in a network. If EdgeTable coincides with a fundamental tie-set, the information of EdgeTable and NodeTable included in the Find Tie-set message is stored in \( n_o \).

III. DISTRIBUTED CONTROL FOR LINK FAILURE

A. Procedure for Link Failure

When a link failure occurs in network lines, a node connected to the failed link detects the failure. If two nodes are connected to the failed link, and the both of them detect the failure at the same time, the node which has a smaller address takes the responsibility to restore the failure. Let \( n_f \) be a node which detects a link failure and \( e_f \) be a failed link. The procedure in \( n_f \) is listed as follows:

**Step1** Blocking physical ports connected to \( e_f \) \( n_f \) blocks its physical port connected to \( e_f \), and sends a Close Port message to another node which is connected to \( e_f \). Then the node blocks its physical port connected to \( e_f \).

**Step2** Choosing a tie-set \( L_i \) from Tie-set Information, where \( e_f \in L_i \) \( n_f \) chooses tie-sets which include \( e_f \) from its Tie-set Information.

**Step3** Determining a tie-set \( L_i \) to conduct route switching \( n_f \) chooses one tie-set \( L_r \) in which the communication path is shifted.

**Step4** Opening physical ports connected to the cotree link \( n_f \) sends an Open Port message to the nodes which are connected to the cotree link of \( L_r \) to resume communication.

The behavior of the procedure above is shown in Fig.2. Next procedure after the steps above depends on the kind of link failure. There are mainly two kinds of link failure. One is that link failure can be restored. Another is that link failure cannot be restored permanently or a failed link itself is removed.

1) The Case where Link Failure can be Restored: In this case, if \( n_f \) detects the restoration signal of the failed link \( e_f \), \( n_f \) shifts the communication path from the cotree link of \( L_r \) back to the restored link \( e_f \).
2) The Case where Link Failure cannot be Restored: In this case, the structure of loops should be transformed in order to maintain a fundamental system of tie-sets. For example, the second network in Fig. 2 does not maintain a fundamental system of tie-sets since $L_3$ contains two cotree links. To maintain a fundamental system of tie-sets, $n_f$ conducts the procedure which applies L-transformation [8] as shown in Fig.3.

B. Advertisement after L-transformation

After the procedure for link failure described in III-A, nodes around $n_f$ are still uninformed of the changes about updated communication paths and tie-sets. Therefore, state information of nodes relevant to link failure should be updated. A node relevant to link failure is defined as a node which belongs to a fundamental tie-set including the failed link $e_f$. State information can be updated by executing an advertisement based on message passing on tie-sets. The message passing is realized by sending Update massages around on tie-sets.

IV. SIMULATION AND EXPERIMENTS

A simulator is made to verify the behavior of the recovery method for link failure suggested in this paper, and to compare against RSTP on behalf of existing technologies because of its general use. We did not conduct experiments on EAPS, since EAPS is not applicable to mesh topological networks. A tool which demonstrates RSTP is created in reference to IEEE standards 802.1D [1]. In configuring a network, links are set to be undirected through which data can flow bi-directionally. In addition, network is designed to be redundant, in other words, bi-connected to be able to cope with failure as shown in Fig.4. As node configuration, each node has input ports and output ports, a message buffer, and a processor. Common buffering method is taken in a simulation node, where all messages received through input ports go to the message buffer. The processor takes each message from the message buffer by polling method. After each message is processed in the processor, the message is sent to other nodes through appropriate output ports unless it is received or discarded.

A. Route Switching Points

The distinguished feature of the failure recovery based on tie-sets is that only one route switching is required to restore link failure. Generally, increase in switching points enlarges the scale influenced by link failure and leads to slow recovery. Therefore experiments to measure the number of switching required to restore one point link failure are conducted to compare against RSTP. A tree which represents communication paths before link failure is denoted as $T_o$, and a renewed tree which represents communication paths after link failure is denoted as $T_n$. To measure the number of route switching points, the distance between $T_o$ and $T_n$ is appropriate. The distance is defined as follows:

$$d(T_o, T_n) = |T_o - T_n|$$

(2)

Let $d_i(T_o, T_n)$ be the distance when link failure occurs on a tree link $e_i$ ($i \in T$). Then the average of route switching points $A_s$ is defined as follows:

$$A_s = \frac{\sum_{i=1}^{n_f} d_i(T_o, T_n)}{\rho} (i = 1, 2, \ldots, \rho(= |T|))$$

(3)

For a given bi-connected undirected graph $G = (V, E)$, the number of nodes $|V|$ is 10 and the range of the number of edges $|E|$ is from 10 to 20. A tree is output by giving link costs at random, and executing Spanning Tree Protocol (STP). Case 1 and Case 2 stand for two different trees obtained by executing STP. As shown in Fig.5, RSTP generally requires more than one time switching, while recovery based on tie-sets needs only one time switching.

B. Influenced Nodes

Subsequently, the number of nodes which are influenced by link failure is counted. An influenced node is defined as follows:

a) A node changing physical port states: A state of communication paths on a network is determined by physical port states of network nodes. When a link failure occurs, it is necessary to open alternate ports to resume communication in addition to closing ports which are connected to the failed link. The number of nodes that change their communication paths on a network is determined by physical port states of network nodes.

b) A node changing state information: State information of each node is updated by an advertisement. Until an advertisement is executed, a network stays unstable owning to discrepancy among state information of network nodes.
influenced by an advertisement. A node has the greatest number of incident links. Under these initial conditions, we conducted experiments to count the number of nodes that change their port states when link failure occurs on a tree link. Let $N_p(e_i)$ be the number of nodes that change their physical port states when link failure occurs on a tree link $e_i \in T$. Then the average of nodes changing their port states $A_p$ is defined as follows:

$$A_p = \frac{\sum_{i=1}^{\rho} N_p(e_i)}{\rho}, \quad (i = 1, 2, \ldots, \rho(= |T|))$$  \quad \text{(4)}$$

The conditions are the same as the experiments on route switching points. As shown in Fig.6, RSTP influences a network to a greater degree than the restoration based on tie-sets.

1) Nodes that Change Physical Port States: As mentioned, port states are important in data transfer. Therefore, if port states of network nodes are changed by failure, the change of port states naturally influences communication on a network. Let $N_p(e_i)$ be the number of nodes that change their physical port states when link failure occurs on a tree link $e_i \in T$. Then the average of nodes changing their port states $A_p$ is defined as follows:

$$A_p = \frac{\sum_{i=1}^{\rho} N_p(e_i)}{\rho}, \quad (i = 1, 2, \ldots, \rho(= |T|))$$  \quad \text{(4)}$$

2) Nodes that Change State Information: Failure recovery based on tie-sets executes update procedure of state information when there is a need for conducting an advertisement. The number of nodes influenced by an advertisement varies with tree structures. There are two major methods to output a tree; Breadth First Search (BFS) and Depth First Search (DFS). Focusing on the latter definition of influenced nodes, we conducted experiments to count the number of nodes that change their state information and to determine which tree is better. For a bi-connected undirected graph $G = (V,E)$ which is given at random, the range of the number of nodes $|V|$ is from 5 to 30. A tree is output by using BFS or DFS. In making a tree, a root node is set for a node that has the greatest number of incident links. Under these initial conditions, experiments were conducted to examine the scale influenced by an advertisement. $A_s$ stands for the average of nodes which change their state information. The average $A_s$ is obtained by the same calculation process as $A_p$. As shown in Fig.7, the BFS is more suitable than DFS since BFS can reduce the number of nodes that change their state information in comparison with DFS.

V. CONCLUSION

In this paper, distributed control for link failure in information networks is suggested based on tie-set concept. As a result of experiments, we substantiate that the restoration based on tie-sets can reduce the scale affected by link failure in comparison with RSTP. As a future study, how to cope with concurrent link failure should be discussed. The protocol using tie-set concept easily works with RSTP if port states are enhanced to correspond to those of RSTP. Therefore switch failure can be dealt with by adding some improvement of port states to the proposed protocol. The proposed method is not specified to particular networks since the method is applicable to various networks, whether they are wired or wireless, in which a tree structure is used for setting communication paths. Although one node has limited local information of tie-sets, an entire network is controlled in an orderly fashion due to the graph theoretical basis of tie-sets. Furthermore a series of local information of each node is consistent with the condition of an entire network. That is because a fundamental system of tie-sets is uniquely determined by graph theoretical tree structure that implicitly underlies a network.

REFERENCES