This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

An Autonomous Distributed Control Method for Link Failure Based on Tie-Set Graph Theory

Kiyoshi Nakayama, *Student Member, IEEE*, Norihiko Shinomiya, *Member, IEEE*, and Hitoshi Watanabe, *Life Fellow, IEEE*

***Abstract*—This study proposes an autonomous distributed control method for single link failure based on loops in a network. This method focuses on the concept of tie-sets defined by graph theory in order to divide a network into a string of logical loops. A tie-set denotes a set of links that constitutes a loop. Based on theoretical rationale of graph theory, a string of tie-sets that cover all the nodes and links can be created by using a tree, even in an intricately-intertwined mesh network. If tie-sets are used as local management units, high-speed and stable fail-over can be realized by taking full advantage of ring-based restoration. This paper first introduces the notion of tie-sets, and then describes the distributed algorithms for link failure. Experiments are conducted against Rapid Spanning Tree Protocol (RSTP), which is generally used for fault recovery in mesh topological networks. Experimental results comparing the proposed method with RSTP suggest that our method alleviates the adverse effects of link failure with a modest increase in state information of a node.**

***Index Terms*—Distributed control, fault tolerance, graph theory, link failure, loop, tie-set.**

NOTATION AND DEFINITIONS

$G=(V,E)$	Bi-connected and undirected graph as networks are assumed redundant and links are bidirectional.
$V=\{v_1,\ldots,v_n\}$	The set of vertices of G.
$E=\{e_1,\ldots,e_m\}$	The set of edges of G.
L_i	Tie-Set, a set of all the edges $\{e_1^i,e_2^i,\ldots,e_k^i\}$ in a loop of G.
T	A spanning tree within G.
$\overline{T}$	Cotree of G, where $\overline{T}=E-T$.
ρ	The rank of G, where $\rho=\rho(G)=\lvert T\rvert$.
μ	The nullity of G, where $\mu=\mu(G)=\lvert\overline{T}\rvert$.
l	An edge of $\overline{T}$, where $l=(a,b)\in\overline{T}$.
G_T	A subgraph $G_T=(V,T)$, where T represents a tree in G.
P_T	One elementary path whose origin is b and terminal is a of $l=(a,b)$ in G_T.
$L(l)$	Fundamental Circuit, a circuit determined by an edge $l=(a,b)\in\overline{T}$ and a path $P_T(a,b)$ on G_T.
v_i	An arbitrary vertex (node) in a network.
v_a	An adjacent node of an arbitrary node.
v_o	A node that creates *Find Tie-set* messages.
v_r	A node that receives a *Find Tie-set* message.
v_f	A node that detects a link failure.
e_i	An arbitrary edge (link) in a network.
e_f	A failed link in a network.
$\boldsymbol{L}^f$	A class of tie-sets that contain a failed link e_f.
L_i^f	A tie-set included in $\boldsymbol{L}^f$.
L_r	A tie-set in which route switching is conducted.
D	The diameter of a graph G.
T_o	A tree that represents communication paths before link failure.
T_n	A renewed tree that represents communication paths after link failure.
$d(T_o,T_n)$	The distance between T_o and T_n, where $d(T_o,T_n)=\lvert T_o-T_n\rvert$.
$d_i(T_o,T_n)$	The distance when link failure occurs on a tree link $e_i(\in T)$.
A_s	The average of the number of route switching points.
$N_h(e_i)$	The number of hops from a failed point to a restored point when link failure occurs on a tree link $e_i(\in T)$.
A_h	The average number of hops defined by $N_h(e_i)$.

Manuscript received September 15, 2011; revised December 23, 2011; accepted January 27, 2012. This work was performed when the authors were at the Graduate School of Engineering, Soka University, Tokyo 192-8577, Japan. This paper was recommended by Associate Editor J. Lu.

K. Nakayama is with the Department of Computer Science, University of California, Irvine, CA 92697 USA (e-mail: kiyoshi.nakayama@ieee.org).

N. Shinomiya and H. Watanabe are with the Department of Information Systems Science, Soka University, Tokyo 192-8577, Japan (e-mail: shinomi@soka.ac.jp, hitoshiw@mtf.biglobe.ne.jp).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSI.2012.2196109

$N_p(e_i)$	The number of nodes that change their physical port states when link failure occurs on a tree link $e_i(\in T)$.
A_p	The average number of nodes changing their port states.
$N_a(e_i)$	The number of nodes that change their state information when link failure occurs on a tree link $e_i(\in T)$.
A_a	The average number of nodes changing their state information.

I. Introduction

As THE Internet continues to grow in size and complexity, it is essential to manage information networks more locally and flexibly with autonomous distributed control architectures. In modern networks becoming larger and more complicated, even a short time failure may cause extensive damage to entire network lines. For this reason, high-speed and reliable restoration for network failures becomes especially important.

It is known that ring-based restoration can realize high-speed and stable fail-over because of the availability of exactly one backup path between any two nodes, leading to simple automatic protection switching mechanisms. For instance, Unidirectional Path Switched Ring (UPSR) [1] or Bidirectional Line Switched Ring (BLSR) [2] is used in a Synchronous Optical Network/Synchronous Digital Hierarchy (SONET/SDH) network. Moreover, Ethernet Automatic Protection Switching or EAPS [3] is utilized in local area networks. In ring restoration, in case of link failure, the end-nodes of the link switch to the backup path joining the two end-nodes. In path protection, all affected connections are notified of the link failure, and they switch to the backup paths. However, the route switching technique in rings requires the reservation of half of the total capacity for protection purposes. Aside from failure management, ring-structured networks are getting more attention in various fields, such as genetic networks [4], smart grid networks [5], [6], etc.

More recently, attention has focused on mesh networks partly because of the increased flexibility they provide in routing connections, and partly because the natural evolution of network topologies leads to a mesh-type topology. While failure recovery in mesh networks can potentially be more efficient, it is more complex as well because of the multiplicity of routes which can be used for recovery. If logical loops virtually existed in mesh networks are efficiently utilized, autonomous distributed control that takes full advantage of ring protection becomes feasible. Ring protection schemes such as UPSR or BLSR are still applied by overlaying logical rings on physical mesh networks because of their simplicity and reliability. This is due to the fact that covering mesh topologies with appropriate rings provides the ring-based restoration functionality on the mesh networks [7]. From the viewpoint of graph theory [8], [9], this problem is equivalent to finding the cycles cover of a graph. There are a variety of conditions to cover a network with cycles. For example, cycles covering every node in a graph [10], rings covering every link in a network [11], and the smallest total length of cycles covering every edges in a graph [12], etc. In the works [13], [14], the relations between spanning trees and fundamental cycle sets of a graph are investigated. The work [15] proposes mesh algorithms for finding a good separating cycle and the triconnected components of a planar graph, and for solving the single function coarsest partitioning problem. The work [16] also proposes how to construct the smallest ring over interconnected systems. Those studies are effective for finding cycles in a sequential manner.

Many studies focusing on cycles and single backup paths for protection and fast recovery have been conducted, especially in the optical network community. Path protection methods based on "p-cycles" that are developed over the past decade have lead to sophisticated techniques and solutions [17], [18]. Besides those finding methods of cycles, autonomous distributed configuration of those cycles as local state information of nodes still remains as a significant issue of fundamental network management. Although ring management has its great merit in path protection, its application to mesh networks still has its difficulty, especially to large-scale and intricately-intertwined nonplanar networks. Artificially embedding physical rings to mesh networks cannot cope with the natural evolution of network topologies.

Today, Rapid Spanning Tree Protocol (RSTP) [19] is generally used for mesh networks to solve the problem of traffic loops and broadcast storms. RSTP is an evolution of the Spanning Tree Protocol (STP), and introduced to provide faster spanning tree convergence after a topology change to reduce recovery times. A major issue of RSTP is that the additional complexity of meshed topology causes fail-over times to increase in the vicinity of a root bridge [20]. There are researches where RSTP is applied to ring topologies in order to conduct fast recovery [20], [21]. Those studies examine how RSTP can be deployed in ring configurations in industrial networks to meet the fault recovery times required by a large number of automation applications.

As discussed, there are a lot of attempts to utilize or embed rings in mesh networks, and many sequential algorithms are proposed. However, those researches beg the question of how distributed architecture should configure state information based on the ring structure. In fact, it is quite difficult for a network node to recognize appropriate information of loops with limited peripheral information of adjacent nodes and incident links. Furthermore, segmentation by rings backed up by mathematical basis as well as orderly distributed control on those units are fraught with complications.

This topic becomes a significant issue in this paper. The proposed method focuses on the concept of tie-sets defined by graph theory in order to divide a network into a set of "logical" loops, not a set of "physical" rings. A tie-set denotes a set of links that constitutes a loop. Based on theoretical rationale of graph theory, a string of tie-sets that cover all nodes and links can be created by using a tree in a distributed manner even in a nonplanar mesh network. By combining the spanning tree algorithm and the notion of tie-sets, we succeed in configuring state information of logical loops that is consistent with entire network state in every node.

There have been previous works that focus on tie-sets in bi-connected graph [22], [23]. Those studies show graph theoretical nature of underlying loops in a network, 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 [24], and distributed algorithms based on tie-sets and simple simulations are also introduced in our previous works [25], [26].

The goal of our study is to establish distributed control architecture based on tie-sets in information mesh networks, and this paper deals with link failure as an important and fundamental issue of network management. Therefore, all algorithms presented in this paper are distributed algorithms, not sequential algorithms. The most significant feature of our method is that mathematically independent loops are created by graph theoretical basis, and autonomous distributed control is realized on the logical and virtual loops defined by tie-sets.

With a modest increase of tie-set information of a node, each node collaboratively behaves for link failure with other nodes in its local unit. Particularly based on the notion of a "fundamental tie-set," route switching can instantly be conducted in any case of link failure only by shifting a failed path to a noncommunication path, which exists just one in a fundamental tie-set. Thereby, ring protection is virtually realized on a logical loop defined by a tie-set. Naturally, adverse effects against communications caused by route switching are greatly lessened compared with STP or RSTP because of local control based on tie-sets. In this paper, we try to analyze strengths of the proposed method from a perspective of distributed algorithms [27] as well as simulation experiments.

The rest of the paper is organized as follows. The graph theory on tie-sets is given in Section II-A. State information of a node is defined in Section II-B, and how a node recognizes tie-set information is described in Section II-C. Algorithms for link failure are given in Section III. Experimental results comparing the proposed method with RSTP are presented in Section IV, and the paper is concluded in Section V.

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 $V = \{v_1, v_2, \ldots, v_n\}$ and a set of edges $E = \{e_1, e_2, \ldots, e_m\}$, let $L_i = \{e_1^i, e_2^i, \ldots, e_k^i\}$ be a set of edges which constitutes a loop in G. The set of edges L_i is called a "tie-set" [9]. Let T and $\overline{T}$ be a spanning tree and a cotree of G, respectively, where $\overline{T} = E - T$. $\rho = \rho(G) = |T|$ and $\mu = \mu(G) = |\overline{T}|$ are called the $rank$ and the $nullity$, respectively. A tree T on a graph $G = (V, E)$ is a maximal set of edges which does not include any tie-set. In other words, for $l \in \overline{T}$, $T \cup \{l\}$ includes one tie-set. Focusing on a subgraph $G_T = (V, T)$ of G and an edge $l = (a, b) \in \overline{T}$, 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:

$$L(l) = (a, l = (a, b), P_T(b, a))$$

$$= (a, l, v_0 = b, t_1, v_1, \ldots, t_h, v_h = a). \quad (1)$$

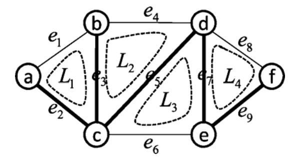

Fig. 1. An example of a fundamental system of tie-sets.

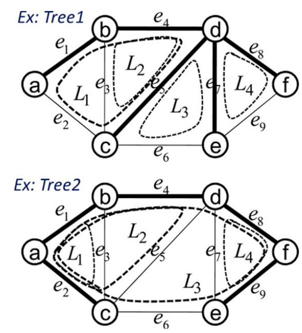

Fig. 2. Different fundamental systems of tie-sets by tree topologies.

In this way, a circuit determined by an edge $l = (a, b) \in \overline{T}$ and a path $P_T(a, b)$ on G_T is denoted as a "fundamental circuit." A simple circuit[1] 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 μ 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. In network segmentation by tie-sets, a topology of tree T becomes important. In other words, tie-set distribution differs by a tree as seen in Fig. 2.

B. State Information of a Node

Each node v_i mainly has three types of information as state information as follows:

1) *Incident Links:* Information of links connected to v_i.

2) *Adjacent Nodes:* Information of nodes which are connected through incident links of v_i.

3) *Tie-Set Information:* Information of fundamental tie-sets to which v_i belongs. When a fundamental tie-set L_i contains e_i that includes v_i in its two vertices, it is defined that v_i 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 Configuring Tie-Set Information

As described in Section II-B, 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

[1]If a path is a simple path with no repeated vertices or edges other than the starting and ending vertices, it is called a simple circuit, cycle, circle, or polygon.

fundamental tie-sets. By automatically configuring state information of tie-sets, a burden for initialization of network managers is greatly lessened especially in large-scale networks.

In information networks, let us assume that a tree T in $G = (V, E)$ corresponds to communication paths, and a cotree $\overline{T}$ corresponds to non communication paths. In this paper, "tree links" and "cotree links" are defined as follows:

- *tree links:* links representing communication paths;
- *cotree links:* links representing non-communication paths.

Here, tree links are expressed by thick lines, while cotree links 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.

A tree T is easily constructed by executing the spanning tree algorithm (STA), which is one of the basic distributed algorithms.

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 v_o creates *Find Tie-set* messages, and then sends those messages to all adjacent nodes of v_o. When sending a *Find Tie-set* message to an adjacent node v_a, v_o adds node information of v_o to *NodeTable*, and adds information of a link connected to both v_o and v_a to *EdgeTable*. Let v_r be a node that receives a *Find Tie-set* message. After receiving a *Find Tie-set* message, v_r executes different procedure by the following cases.

Case 1: $v_r \neq v_o$ In this case, if *EdgeTable* of the *Find Tie-set* message includes more than one cotree link, v_r discards the message. If *EdgeTable* contains no or one cotree link, v_r 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 v_o, v_r sends the copied message to v_o even if $v_o \in$ *NodeTable*. When sending a copied message to an adjacent node v_a, v_r adds node information of v_r to *NodeTable*, and adds information of a link connected to both v_r and v_a to *EdgeTable*.

Case 2: $v_r = v_o$ 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 v_o.

In this algorithm, the number of messages casted on a network, so-called Communication Complexity, can be analyzed by focusing on a format of the *Find Tie-set* message. The communication complexity is determined to be $O(n^4)$ from a perspective of distributed algorithm, where $n = |V|$. However, the number of physical ports of a node is actually limited. In this case, the Communication Complexity is determined to be $O(n^3)$. As for execution time, a message passes through on tree links and one cotree link. Therefore, time complexity is $O(D)$ where D is defined as a diameter of a graph G.

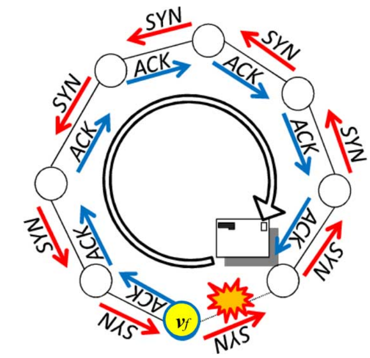

Fig. 3. Failure detection in a tie-set.

III. Distributed Control for Link Failure

A. Tie-Set Agent and Failure Detection

1) Tie-Set Agent: A tie-set agent is defined as a message that constantly circulates on a fundamental tie-set in a certain direction, for instance, in a clockwise fashion on a tie-set. A tie-set agent in a tie-set L_i recognizes state information of all the nodes that belong to L_i. Therefore, μ tie-set agents exist in a network. If some information changes at a certain point in a tie-set L_i, its tie-set agent understands the change and notifies the up-to-date information to other nodes that belong to L_i.

2) Failure Detection: In case that failure occurs in a tie-set L_i, its tie-set agent message drops on a failed link. Thereby, all the nodes in L_i notice that some kind of failure occurs at certain point in L_i. Then each node v_i in L_i sends a *SYN* message to an adjacent node v_a in a tie-set L_i in the opposite direction from the direction of its tie-set agent as shown in Fig. 3. After sending a *SYN* message, v_i executes different procedure by the following cases.

Case 1: v_i receives a signal from v_a

In this case, the adjacent node v_a of v_i and the link between v_i and v_a is undamaged, since an ACK message comes back from v_a.

Case 2: v_i does not receive a signal from v_a

In this case, an ACK message does not come back from v_a. v_i understands that failure happens in v_a or on the link between v_i and v_a. Then v_i and v_a negotiates to decide which node conducts failure recovery. Naturally, two nodes are connected to the failed link. In case that both of them detect the failure at the same time, certain criterion must be defined beforehand. In this method, the node that has a smaller address takes the responsibility to restore the failure.

B. Procedure of a Node in Failure Detection

1) Distributed Algorithm for Link Failure: Let v_f be a node that detects a link failure and e_f be a failed link. The procedure in v_f is listed as follows:

Step 1) *Blocking physical ports connected to* e_f
v_f blocks its physical port connected to e_f, and sends a *Close Port* message to another node that is connected to e_f. Then the node blocks its physical port connected to e_f.

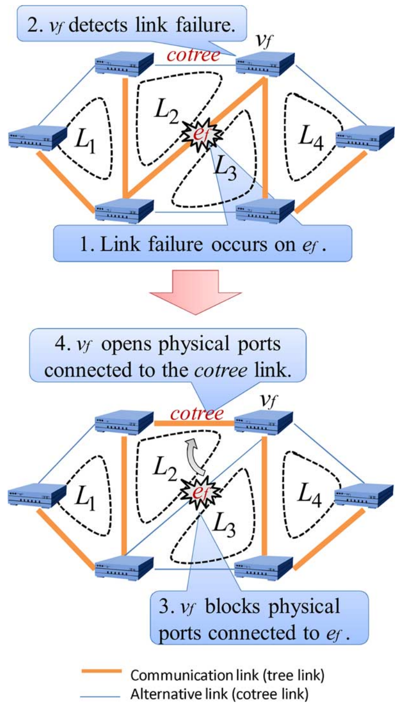

Fig. 4. Overall behavior in changing a communication path.

Step 2) *Selecting tie-sets $\boldsymbol{L}^f$ that include e_f from Tie-set Information*
v_f selects tie-sets $\boldsymbol{L}^f = \{L_i^f\}$ from Tie-set Information, where $e_f \in L_i^f$.

Step 3) *Determining a tie-set L_r to conduct route switching*
When several tie-sets including e_f exist in v_f, v_f chooses one tie-set L_r in which the communication path is shifted from $\boldsymbol{L}^f$.

Step 4) *Opening physical ports connected to the cotree link*
v_f sends an *Open Port* message to the nodes which are connected to the cotree link of L_r to resume communication.

In Step 3 above, there are several criteria to decide a tie-set in which route switching is conducted. Criteria are, for instance, the number of hops to a cotree link, the total value of link weights of a tie-set, the size of a tie-set, etc. It depends on the characteristics of a network to select a tie-set that is appropriate to shift a communication path. If there is not any unique feature in a network, the number of hops becomes a proper criterion. The behavior of the procedure above is shown in Fig. 4.

2) Procedure After Link Failure: 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.

Case 1: Link Failure can be Restored

In this case, v_f detects the restoration signal of the failed link e_f. Then v_f shifts the communication path from the cotree link of L_r back to the restored link e_f as seen in case 1 of Fig. 5.

Case 2: 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. 4 does not maintain a fundamental system of tie-sets since L_3 contains two cotree links. To maintain a fundamental system of tie-sets, v_f conducts the procedure which applies L-transformation [23] as shown in case 2 of Fig. 5.

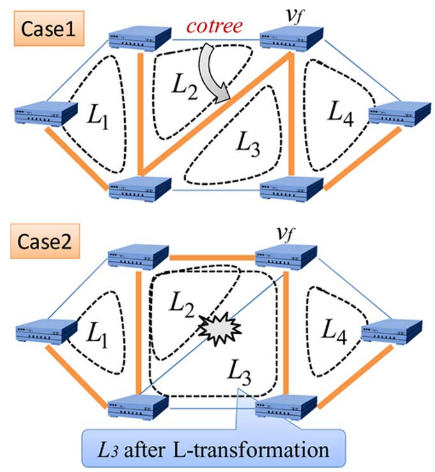

Fig. 5. Procedure after link failure.

3) L-Transformation: In this paper, L-transformation is defined as transformation of a fundamental system of tie-sets. If a formation of tree changes in a network, its fundamental system of tie-sets also changes in response to the transformed tree. Let $\boldsymbol{L}^f$ be a class of tie-sets that contains a failed link, and L_r be a tie-set in which route switching is conducted. For each tie-set L_i^f where $(L_i^f \in \boldsymbol{L}^f) \wedge (L_i^f \neq L_r)$,[2] v_f executes $L_i^f \leftarrow L_i^f \oplus L_r$.[3] Then v_r notifies the updated information by L-transformation to other nodes by means of advertisement based on tie-sets.

C. Advertisement After L-Transformation

After the procedure for link failure described in Section III-B, nodes around v_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 that includes the failed link e_f. State information can be updated by executing an advertisement based on massage passing on tie-sets. The massage passing is realized by sending *Update* messages around on tie-sets as shown in Fig. 6. Time complexity of advertisement based on tie-sets is $O(D)$, where D is a diameter of a graph. The number of messages is equivalent to the number of tie-sets that contains a failed link. However, considering the worst case, communication complexity becomes $O(|E|)$.

IV. Simulation Experiments and Analysis

A simulator is made by Java 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

[2]A conjunction $\wedge$ is a compound statement formed by joining two statements with the connector AND. The conjunction "P and Q" is symbolized by $P \wedge Q$. A conjunction is true when both of its combined parts are true; otherwise it is false.

[3]The definition of $\oplus$ for a set A and a set B is defined as follows: $A \oplus B = (A - B) \cup (B - A) = (A \cup B) - (A \cap B)$.

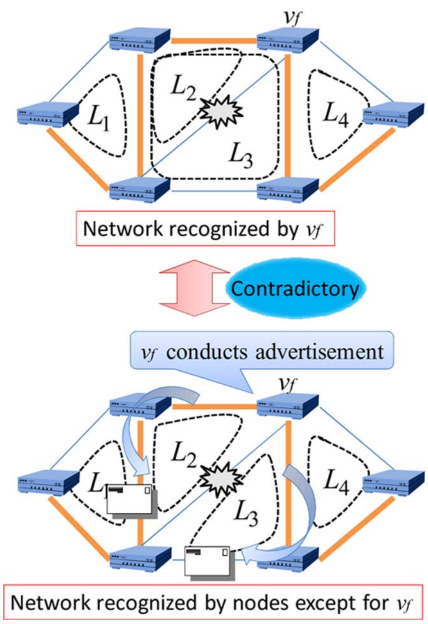

Fig. 6. Advertisement based on tie-sets after failure recovery.

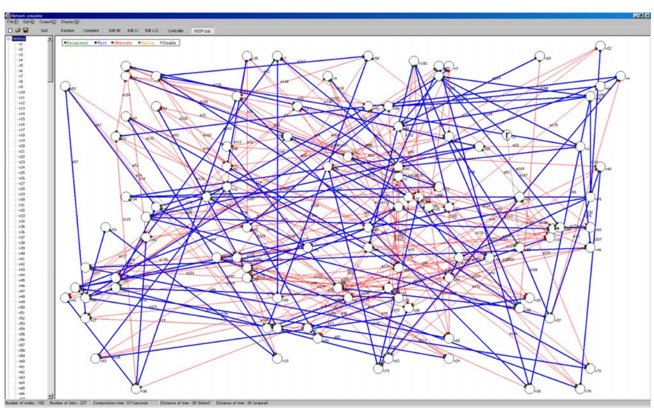

Fig. 7. Network configuration consisting of 100 nodes created at random.

EAPS, since EAPS is not applicable to mesh topological networks such as a network shown in Fig. 7. A tool which demonstrates RSTP is created using Delphi in reference to IEEE standards 802.1D [19]. 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. 7. 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 route switching points leads to the factors as follows:

- throughput degradation;

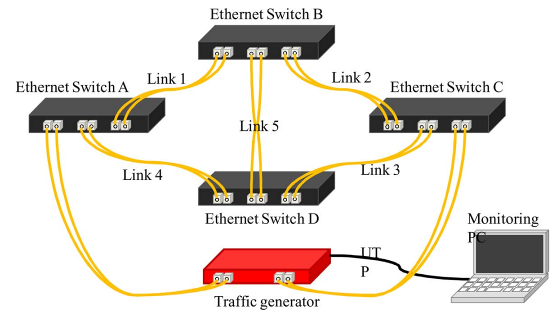

Fig. 8. Experimental environment of the evaluation experiment for scalability.

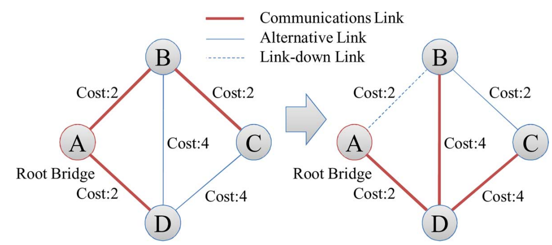

Fig. 9. Configuration of RSTP where route switching occurs 2 times.

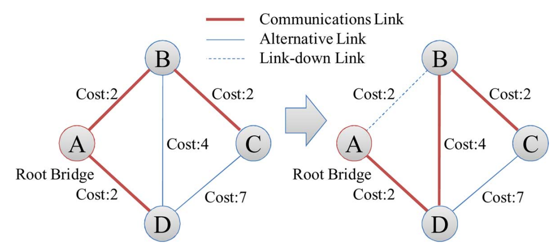

Fig. 10. Configuration of RSTP where route switching occurs 1 time.

- slow recovery.

We first conducted experiments to verify the points listed above using Ethernet switches. Experimental environment is shown in Fig. 8. In experiments, we set up 4 Ethernet switches, and connected fibers as shown in Fig. 8. A switch to cause link failure is set on link 1, and a Next Stream is connected to switch A and B. By changing a cost on link 3, the number of times of route switching is controlled. Setup 1 is configuration of RSTP where route switching occurs 2 times as shown in Fig. 9. Setup 2 is configuration of RSTP where route switching occurs 1 time as shown in Fig. 10. In setup 1 shown in Fig. 9, the total link cost of the path A $\rightarrow$ D $\rightarrow$ B $\rightarrow$ C surpasses the total link cost of the path A $\rightarrow$ D $\rightarrow$ C. Accordingly, the first route switching occurs from link 1 to link 5, and second route switching occurs from link 2 to link 3, requiring 2 times of switching. In setup 2 shown in Fig. 10, the total link cost of the path A $\rightarrow$ D $\rightarrow$ B $\rightarrow$ C is smaller than the total link cost of the path A $\rightarrow$ D $\rightarrow$ C. Therefore, route switching occurs just 1 time.

The experimental procedure is as follows.

1) Transfer data from Switch A to C using a Next Stream.
2) Turn on a switch to cause single link failure during data transfer on link 1.
3) Keep breaking the communications link for 10 s.
4) Measure throughput 20 times.

We define communications down time as the time in which data transfer is interrupted. Fig. 11 shows the comparison of the throughput of setup 1 and 2, and Table I shows the comparison of communications down time of setup 1 and 2. The solid line

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

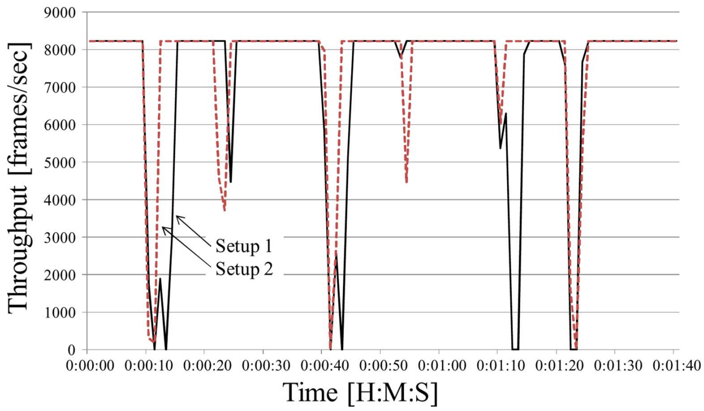

Fig. 11. Comparison of the throughput of setup 1 and 2.

TABLE I
COMPARISON OF COMMUNICATIONS DOWN TIME OF SETUP 1 AND 2

	Setup 1		Setup 2	
Switch	ON	OFF	ON	OFF
Average[s]	3.413	1.051	1.302	1.262
Variance	0.756	0.628	0.694	0.435
Maximum[s]	4.782	2.153	3.129	2.144
Minimum[s]	2.104	0.054	0.038	0.067

denotes the throughput of setup 1, whereas the dotted line denotes the throughput of setup 2 in Fig. 11. Both Fig. 11 and Table I suggest that communications down time of setup 1 is longer than that of setup 2 since setup 1 requires 2 times of route switching. Thereby, throughput naturally degrades in setup 1 because of increase in route switching. In this experiment, we set only 4 switches. If the number of switches increases, communications downtime is considered to become longer.

On the basis of these results, experiments to measure the number of times of route switching required to restore one point link failure are conducted to compare against RSTP. A tree that represents communication paths before link failure is denoted as T_o, and a renewed tree that 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:

$$d(T_o, T_n) = |T_o - T_n|. \tag{2}$$

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

$$A_s = \frac{\sum_{i=1}^{\rho} d_i(T_o, T_n)}{\rho}, (i = 1, 2, \ldots, \rho(= |T|)). \tag{3}$$

For a given bi-connected and undirected graph $G = (V, E)$, a graph G is created at random with the number of nodes $|V|$ ranging from 20 to 100. A tree is output by giving link costs at random, and executing the Spanning Tree Algorithm (STA). Tree 1 and Tree 2 stand for two different trees obtained by giving different link costs and executing STA. Fig. 12 is the experimental results that show the average A_s of the number of route switching points. As shown in Fig. 12, RSTP requires about twice as many switchings as the proposed method on average, while recovery based on tie-sets needs only one time switching. In addition, A_s shows modest upward tendency when a network becomes larger.

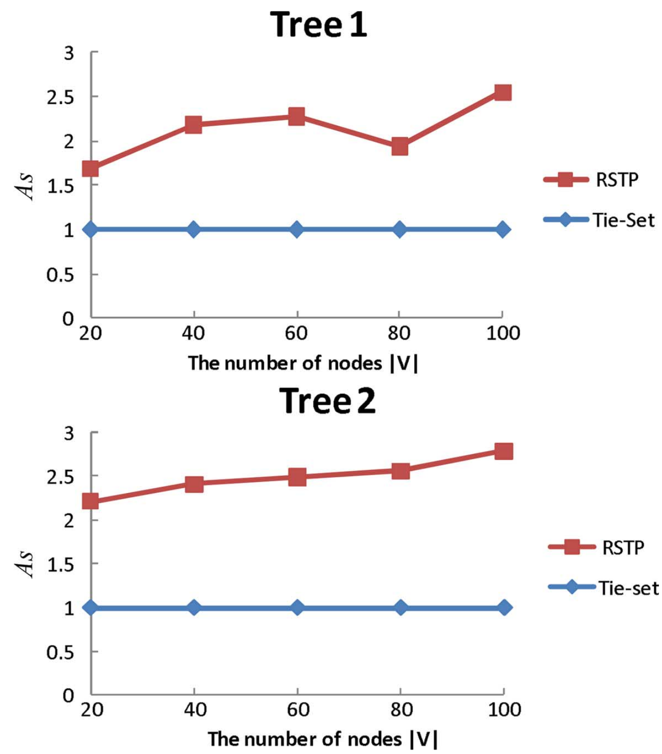

Fig. 12. The average times of route switching.

Fig. 13 shows the maximum times of route switchings $(\max\ \{d_i(T_o, T_n)\})$ for each given graph whose condition is the same as the experiment to measure A_s. While route switching based on tie-sets constantly needs one shifting, RSTP requires much more switching than the proposed method. This is because RSTP greatly changes its tree topology in case of failure in the vicinity of a root bridge. Failure near a root node is the biggest problem in operation of RSTP. The remarkable tendency of augmentation of the number of times of route switching is seen in a large-scale network. For example, 35 times of route switching, which can be seen in a graph with 100 nodes in Tree 2 of Fig. 13, greatly fluctuate the configuration of communication paths. In that case, throughput degrades seriously as well as convergence time greatly increases making an entire network unstable.

B. Estimation of Recovery Time

In order to estimate recovery time for link failure, we counted the number of hops from a node that detects failure to a node that opens its physical port. In operation of RSTP, route switching often occurs several times. In that case, the most remote node from a root bridge should be counted since restoration finishes when the node sets its port state as *Destination Port*.

Let $N_h(e_i)$ be the number of hops from a failed point to a restored point when link failure occurs on a tree link $e_i (\in T)$. Then the average number of hops A_h is defined as follows:

$$A_h = \frac{\sum_{i=1}^{\rho} N_h(e_i)}{\rho}, (i = 1, 2, \ldots, \rho(= |T|)). \tag{4}$$

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

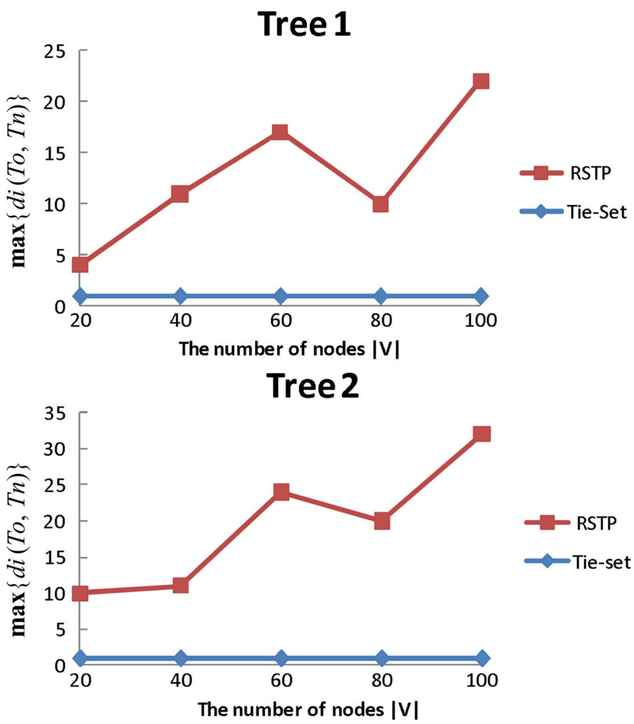

Fig. 13. The maximum times of route switching.

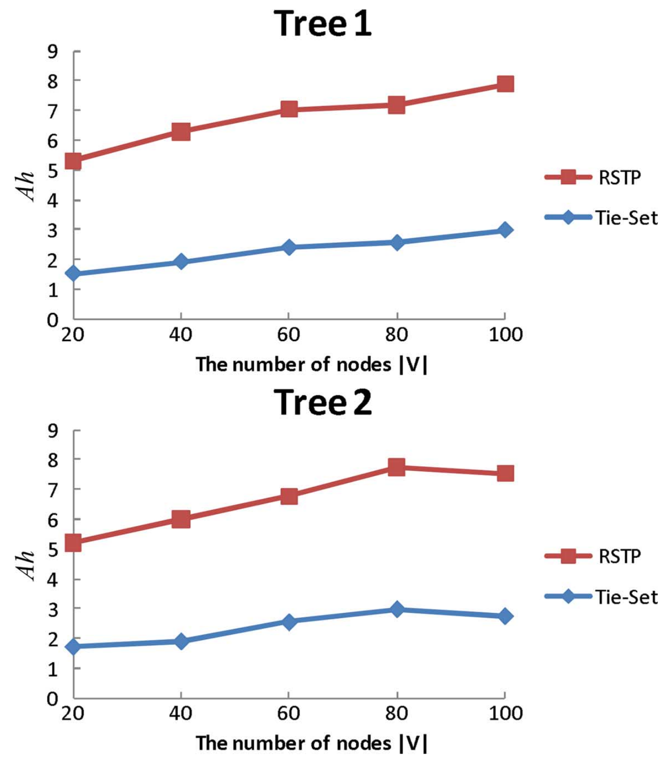

Fig. 14. The average number of hops.

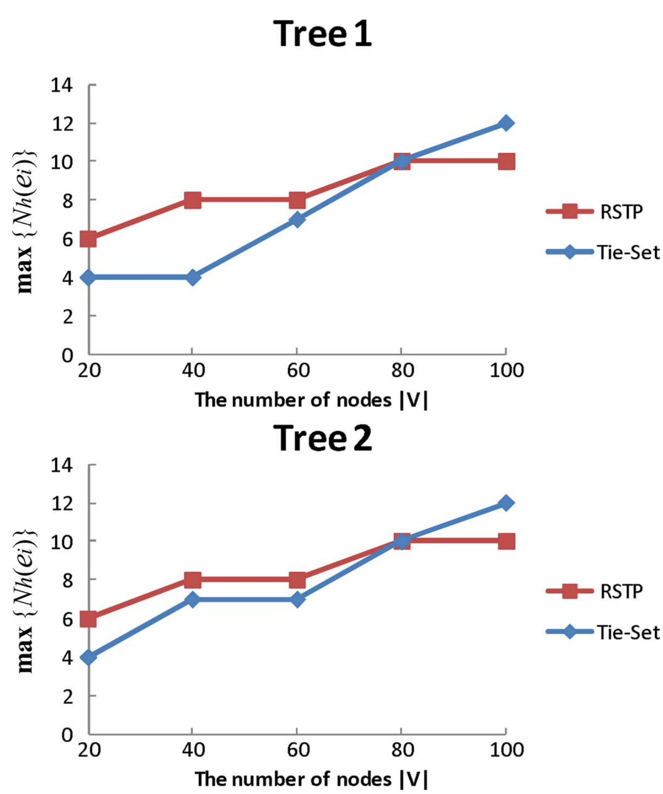

Fig. 15. The maximum number of hops.

The conditions of network configurations are the same as the experiments on route switching points.

Fig. 14 is the results that show the average number of hops A_h. As shown in Fig. 14, recovery time of RSTP is about three times longer than that of the proposed method on average. Fig. 15 shows the maximum number of hops $(\max\{N_h(e_i)\})$ for each given graph whose condition is the same as the experiment of A_h. As seen in Fig. 15, the maximum number of hops of RSTP and the proposed method is almost the same. This is because the number of hops from a root node to an alternative link is almost the same in both methods when failure occurs in the vicinity of a root bridge. Therefore, as for recovery time, the proposed method realizes faster restoration on average than RSTP, although the worst case is balanced out.

C. 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:* The states of communication paths on a network are 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. In the process of changing the states of communication ports, the loss of frames occurs. The number of nodes that change their port states should be a criterion to measure reliability of a restoration method, since increase in influenced nodes in port states directly leads to instability of a network.

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.

1) Nodes That Change Physical Port States: As mentioned, port states are important in data transfer. Therefore, if port states are changed by failure, the change of port states naturally influences communications on a network. Let $N_p(e_i)$ be the number of nodes that change their physical port states when link failure

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

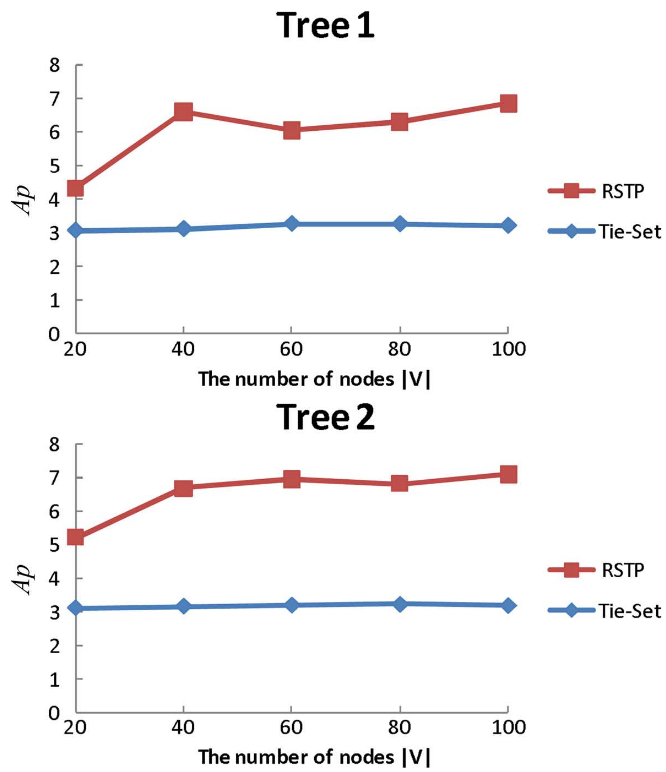

Fig. 16. The average number of nodes changing port states.

occurs on a tree link $e_i(\in T)$. Then the average number of nodes changing their port states A_p is expressed as follows:

$$A_p = \frac{\sum_{i=1}^{\rho} N_p(e_i)}{\rho}, (i = 1, 2, \ldots, \rho(= |T|)). \tag{5}$$

The conditions of network configurations are the same as the experiments on route switching points. Fig. 16 is the results that show the average number of nodes which change their port states A_p. As shown in Fig. 16, affected nodes in communications port states by RSTP are about twice as many as those by our method.

Fig. 17 shows the maximum number of nodes that change their communications port states ($\max\{N_p(e_i)\}$) for each given graph whose condition is the same as the experiment to measure A_p. As seen in Fig. 17, as a network scale becomes larger, the number of affected nodes in communications port states by RSTP greatly increases in the worst case. Thereby, communication reliability remarkably debases. The result shows a correlation between route switching points and influenced nodes. In other words, the increase in route switching points also leads to low communication reliability.

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 b) 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

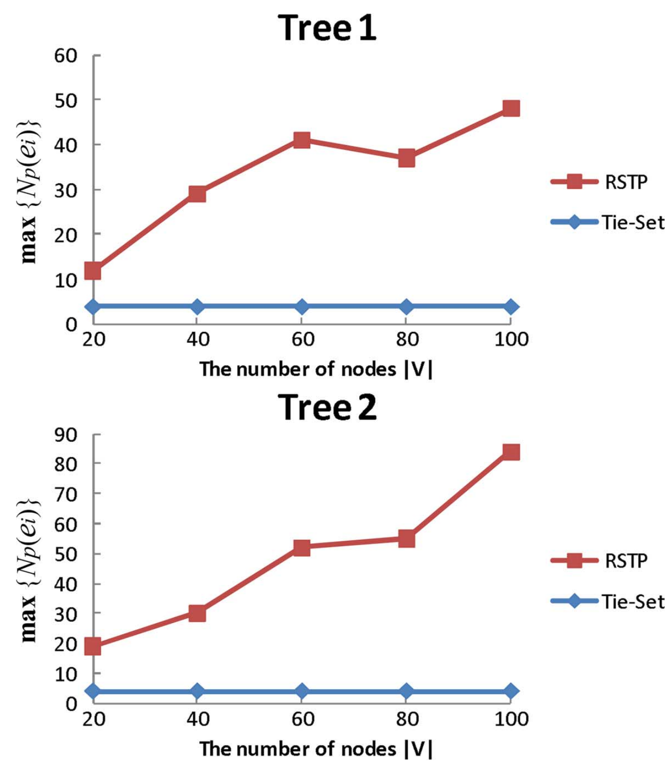

Fig. 17. The maximum number of nodes changing port states.

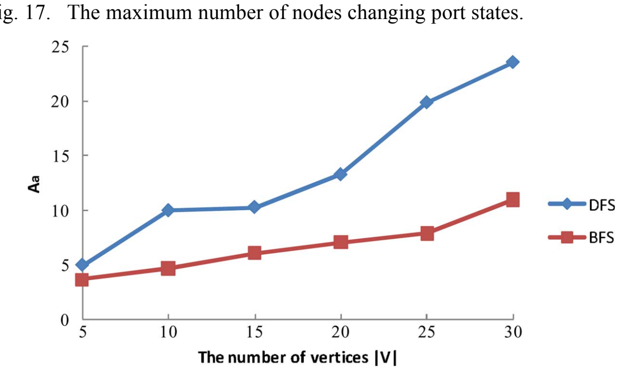

Fig. 18. The average number of nodes updating state information.

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. Let $N_a(e_i)$ be the number of nodes that change their state information when link failure occurs on a tree link $e_i(\in T)$. Then the average number of nodes changing their state information A_a is defined as follows:

$$A_a = \frac{\sum_{i=1}^{\rho} N_a(e_i)}{\rho}, (i = 1, 2, \ldots, \rho(= |T|)). \tag{6}$$

Fig. 18 is the results that show the average A_a. As shown in Fig. 18, the BFS is more suitable than DFS since BFS can reduce the number of nodes that change their state information in comparison with DFS.

D. Complexity Over Traditional Switch Design

One part that should be discussed is the increased complexity in terms of switch designs over traditional techniques.

1) Complexity in State Information: We discussed communications complexity and time complexity so far, and substantiated the superiority over traditional switches such as RSTP and STP. The better the communications and time complexity of an algorithm become, the faster and more stable the algorithm will carry out its work in practice. Apart from those complexities, its space complexity is also important: This is essentially the number of memory cells that an algorithm needs. A good algorithm keeps this number as small as possible, too.

As stated in Section II-B, this method requires state information of fundamental tie-sets that a node belongs to. Since a node checks its Tie-set Information when recovering link failure, the proposed method requires $O(\mu D)$ as space complexity, where μ is a nullity and D is a distance of a graph G. The complexity depends on the formation of tree. As mentioned, the structure of a tree becomes important in many ways so that the creation of the optimal tree for various problems is also under consideration.

Traditional switches mainly have information of adjacent nodes and the conditions of incident links. From a perspective of distributed algorithms, the space complexity of those switches is $O(|V|)$.

2) Complexity in Synchronization: Due to the tie-set agent crawling around each tie-set, this method requires $O(D)$ to synchronize state information. On another front, traditional schemes realize synchronization with $O(1)$ as they just communicate with adjacent nodes. However, traditional switches set some time interval (TI) to complete synchronization. That means time complexity to realize synchronization does not matter as $O(D)$ is not large value in today's advanced high-speed communication technologies.

V. Conclusion and Future Work

In this paper, an autonomous distributed control method for link failure in information networks is suggested based on tie-set graph theory. 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.

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 topology is used in 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.

As a future study, we should discuss how to cope with concurrent link failures as well as node failure. The proposed method easily works with RSTP, since our method employs the spanning tree algorithm. Therefore, switch failure can be dealt with by adding some improvement to port states of the proposed protocol.

Acknowledgment

The authors would like to thank colleagues K. Kadena, J. Fukuda, K. Konohara for their very helpful comments on the paper draft and their assistances in conducting simulations. The authors also acknowledge Prof. Emeritus K. Kobayashi with all our heart for his valuable advice and suggestions. This work has benefited from invaluable comments and constructive suggestions by reviewers and researchers of IEEE International Symposium on Circuits and Systems (ISCAS) 2010, IEEE Symposium on Computers and Communications (ISCC) 2011, and the editors and the reviewers of the IEEE Transactions on Circuits and Systems—Part I: Regular Papers.

References

[1] Understanding SONET UPSRs [Online]. Available: http://www.sonet.com/EDU/upsr.htm
[2] Understanding SONET BLSRs [Online]. Available: http://www.sonet.com/EDU/blsr.htm
[3] S. Shah and M. Yip, "Extreme networks' ethernet automatic protection switching (EAPS) version 1," *Network Working Group, Request for Comments: 3619*, Oct. 2003.
[4] F. Wu, "Stability and bifurcation of ring-structured genetic regulatory networks with time delays," *IEEE Trans. Circuits Syst. I, Reg. Papers*, to be published.
[5] K. Nakayama and N. Shinomiya, "Distributed control based on tie-set graph theory for smart grid networks," in *Proc. Int. Congr. Ultra Modern Telecommun. Control Syst. Workshops (ICUMT)*, Dec. 2010, pp. 957–964.
[6] M. Kantarci, B. Kantarci, and H. Mouftah, "Reliable overlay topology design for the smart microgrid network," *IEEE Netw.*, vol. 25, pp. 38–43, Oct. 2011.
[7] M. Medard, R. Barry, S. Finn, W. He, and S. Lumetta, "Generalized loop-back recovery in optical mesh networks," *IEEE/ACM Trans. Netw.*, vol. 10, no. 1, pp. 153–164, 2002.
[8] M. Swamy and K. Thulasiraman, *Graphs, Networks, and Algorithms*. New York: Wiley Interscience, 1981.
[9] M. Iri *et al., Graph Theory With Exercises*. Tokyo, Japan: Corona Publ., 1983.
[10] O. J. Wasem, "An algorithm for designing rings for survivable fiber networks," *IEEE Trans. Rel.*, vol. 40, pp. 428–439, 1991.
[11] L. Gardner, M. Heydari, J. Shah, I. Sudborough, I. Tolis, and C. Xia, "Techniques for finding ring covers in survivable networks," in *Proc. IEEE GLOBECOM*, San Francisco, CA, Nov. 1994, pp. 1862–1866.
[12] C. Thomassen, "On the complexity of finding a minimum cycle cover of a graph," *SIAM J. Comput.*, vol. 26, no. 3, pp. 675–677, 1997.
[13] N. Deo, "Minimum-length fundamental cycle set," *IEEE Trans. Circuits Syst.*, vol. CAS-26, no. 10, Oct. 1979.
[14] M. Syslo, "On the fundamental cycle set graph," *IEEE Trans. Circuits Syst.*, vol. CAS-29, no. 3, pp. 136–138, Mar. 1982.
[15] J. Jaja and S. Kosaraju, "Parallel algorithms for planar graph isomorphism and related problems," *IEEE Trans. Circuits Syst.*, vol. 35, no. 3, Mar. 1988.
[16] J. Pliam and E. Lee, "On the global properties of interconnected systems," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 42, no. 12, pp. 1013–1017, Dec. 1995.
[17] D. Stamatelakis and W. Grover, "Theoretical underpinnings for the efficiency of restorable networks using preconfigured cycles ("p-cycles")," *IEEE Trans. Commun.*, vol. 48, no. 8, pp. 1262–1265, Aug. 2000.
[18] D. Schupke, C. Gruber, and A. Autenrieth, "Optimal configuration of p-cycles in WDM networks," in *Proc. IEEE Int. Conf. Commun. 2002*, vol. 5, pp. 2761–2765.
[19] *IEEE Computer Society Sponsored by the LAN/MAN Standards Committee*, IEEE Standards 802.1D, IEEE Standard for Local and Metropolitan Area Networks: Media Access Control (MAC) Bridges, 9, Jun. 2004.
[20] M. Pustylnik, M. Vukotic, and R. Moore, "Performance of the rapid spanning tree protocol in ring network topology," RuggedCom, Inc..
[21] D. DesRuisseaux, "Use of RSTP to cost effectively address ring recovery applications in industrial ethernet networks," presented at the ODVA Conf. 13th Annu. Meet., Howey-in-the-Hills, Florida, USA, Feb. 25, 2009.
[22] N. Shinomiya, T. Koide, and H. Watanabe, "A theory of tie-set graph and its application to information network management," *Int. J. Circuit Theory Appl.*, vol. 29, pp. 367–379, 2001.

[23] T. Koide, H. Kubo, and H. Watanabe, "A study on the tie-set graph theory and network flow optimization problems," *Int. J. Circuit Theory Appl.*, vol. 32, pp. 447–470, 2004.

[24] T. Koide and H. Watanabe, "A theory of tie-set graph and tie-set path—A graph theoretical study on robust network system," in *Proc. 2000 IEEE Asia Pacific Conf. Circuits Syst.*, vol. 1, pp. 227–230.

[25] K. Nakayama, N. Shinomiya, and H. Watanabe, "Distributed control for link failure based on tie-sets in information networks," in *Proc. 2010 IEEE Int. Symp. Circuits Syst. (ISCAS)*, pp. 3913–3916.

[26] K. Nakayama and N. Shinomiya, "Autonomous recovery for link failure based on tie-sets in information networks," in *Proc. 2011 IEEE Symp. Comput. Commun. (ISCC)*, Aug. 15, 2011, pp. 671–676.

[27] N. A. Lynch*, Distributed Algorithms*. San Francisco, CA: Morgan Kaufmann, 1996.

Kiyoshi Nakayama (S'09) received the B.S. degree in the Department of Information Systems Science and the M.S. degree in the Graduate School of Engineering, from Soka University, Tokyo, Japan, in 2009 and 2011, respectively. He is currently working toward the Ph.D. degree in the Department of Computer Science, University of California, Irvine.

Mr. Nakayama received the Best Paper Award from International Congress on Ultra Modern Telecommunications and Control Systems (ICUMT) 2010.

Norihiko Shinomiya (S'99–M'00) received the B.E., M.E., and Ph.D. degrees in information systems engineering from Soka University in Tokyo, Japan, in 1995, 1997, and 2001, respectively.

From 2000 to 2005, he was a research engineer of the Network Systems Laboratories, Fujitsu Laboratories Ltd., Kawasaki, Japan. Since 2005, he has been an Associate Professor in the Department of Information Systems Science, Faculty of Engineering, Soka University. He has been engaged in the research and development of design method, control architecture and management system in photonic networks based on the graph theoretical algorithms.

Dr. Shinomiya is a member of IEEE ComSoc, LEOS, and the Institute of Electronics, Information and Communication Engineers (IEICE).

Hitoshi Watanabe (M'59–F'72–LF'96) was born in Shimane, Japan, in December 26, 1930. He received the B.E. degree in electrical engineering and the Ph.D. degree in 1953 and 1961, respectively, from Kyoto University, Kyoto, Japan.

In 1953 he joined NEC Corporation, where he engaged in development of circuit theory and digital computers. From 1967 to 1971, he was a Manager of the Computer Science Laboratory at NEC's Central Research Laboratory. From 1971 to 1980, he developed a small computer business as a General Manager. From 1980 to 1991, he was a Vice President of NEC Corporation. Since 1991, he has been a Professor of Department of Information System Science, Faculty of Engineering, Soka University, Tokyo, Japan. He has published a number of papers in circuit theory, network design, computer aided design, and office automation. He is the author of several books on circuit theory, computer aided design and office automation.

Dr. Watanabe has received the Inata Memorial Award in 1960, three Best Paper Awards in 1961, 1968, and 1969, and the Best Book Author's Award in 1969, respectively, all from IEICE of Japan. He also received the IEEE Centennial Medal in 1984, the Science and Technology Achievement Award from the Governor of the Tokyo Metropolis in 1990, and the IEEE Circuits and Systems Society Award in 1991. He has received a number of awards from various organizations as represented by the IEEE Gustav R. Kirchhoff Award in 2010 for pioneering contributions to filter design theory and computer-aided circuit design. Dr. Watanabe is currently a Professor Emeritus at Soka University, Tokyo, Japan.