Université du Québec
Institut national de la recherche scientifique
Énergie, Matériaux et Télécommunications (EMT)

MÉCANISME DE PROTECTION RAPIDE POUR UN RÉSEAU ETHERNET DE CLASSE OPÉRATEUR CONTRÔLÉ PAR LE PROTOCOLE IS-IS

Par

Marcelo Nascimento dos Santos

Mémoire présenté pour l'obtention du grade de
Maître es Sciences, M.Sc.
en télécommunications

Jury d'évaluation

Examinateur externe	Prof. Brunilde Samsò École Polytechnique de Montréal
Examinateur interne	Prof. Martin Maier INRS-EMT
Directeur de recherche	Prof. Jean-Charles Grégoire INRS-EMT

Résumé

Depuis sa création, la technologie Ethernet a évolué d'une technologie de réseaux locaux jusqu'à être utilisée dans les réseaux d'accès métropolitain et de transport. Plusieurs mécanismes sont utilisés pour que les nouveaux réseaux Ethernet soient capables d'offrir les caractéristiques d'un réseau de classe opérateur (*Carrier-class*). Pour le contrôle du réseau, l'utilisation d'un protocole à état de liens (*link-state*) à la place du *Rapid Spanning Tree Protocol (RSTP)* est l'une des améliorations utilisées par les nouvelles technologies Ethernet telles que le *Provider Link State Bridging (PLSB)*, le *Transparent Interconnection of Lots of Links (TRILL)* et le *Shortest Path Bridging (SPB)*; ces technologies utilisent le protocole *Intermediate System to Intermediate System (IS-IS)* qui était déjà utilisé comme protocole de routage dans les réseaux sans connexion comme *IP*. Pourtant, pour que ces nouveaux réseaux Ethernet puissent atteindre le niveau de performance d'un réseau de classe opérateur, comme pour les réseaux *Time-division Multiplexing (TDM)* traditionnels, il faut que de nouveaux mécanismes spécifiquement adaptés à ce genre de réseau soient développés. Le temps de convergence élevé des protocoles de contrôle Ethernet est un des problèmes rencontrés tant dans les réseaux traditionnels que de nouvelle génération. L'objectif de notre recherche est donc la proposition d'un mécanisme de recouvrement capable d'utiliser des chemins alternatifs préconfigurés pour améliorer la vitesse de recouvrement des réseaux Ethernet qui utilisent le protocole *IS-IS* dans le plan de contrôle. Dans ce mémoire nous présentons une analyse du temps de recouvrement des protocoles de contrôle utilisés dans les réseaux Ethernet. Nous développons aussi un mécanisme de recouvrement rapide pour un réseau Ethernet de classe opérateur contrôlé par un protocole à état de liens. Des simulations sont utilisées pour montrer l'avantage de l'utilisation de ce mécanisme en relation au mécanisme de recouvrement global du protocole *IS-IS*.

Mots-clés Ethernet, Classe opérateur, Recouvrement Rapide, IS-IS

Abstract

Ethernet has evolved from a use in local area networks to a technology that can be used in metro access as well as long-haul transport networks. Several mechanisms have been created to adapt the Ethernet control plane to the requirements of a Carrier-class network. The use of a link-state routing protocol rather than the traditional RSTP is one of the improvements employed by technologies such as Transparent Interconnection of Lots of Links (TRILL) and Shortest Path Bridging (SPB) to achieve better and more predictable reliability levels. These technologies use the IS-IS routing protocol, which has already been employed as a routing protocol in IP networks. Unfortunately even with these improvements, new Ethernet networks still cannot achieve the level of reliability of traditional TDM-derived Carrier-class networks. In order to improve Ethernet reliability and availability performance, we need to develop new, specifically adapted mechanisms. More specifically, the high convergence time of control protocols can cause problems in both traditional and new generation Ethernet networks. The goal of our research is to present a local recovery mechanism capable of using preconfigured alternative paths to improve the speed of recovery. We will also present an analysis of the recovery time of two control protocols used in Ethernet networks, namely RSTP and IS-IS. Our fast recovery mechanism is adapted to a Carrier-class Ethernet network controlled by a link-state protocol. Simulations are used to show the advantage of the use of this mechanism in comparison to the global recovery mechanism of link-state protocols.

Keywords Ethernet, Carrier-class, Fast Recovery, IS-IS

Table des matières

Liste des figures

Liste des abréviations

100-GET	100 Gbit/s Carrier-Grade Ethernet Transport Technologies
APS	Automatic Protection Switching
ATM	Asynchronous Transfer Mode
B-VID	Backbone VID
BPDU	Bridge Protocol Data Units
BRITE	Boston university Representative Internet Topology gEnerator
CBR	Constant Bit Rate
CFM	Connectivity Fault Management
CSMA/CD	Carrier Sense Multiple Access With Collision Detection
CSNP	Complete Sequence Number Packet
EAPS	Ethernet Automatic Protection Switching
ECMP	Equal Cost Multipath
ERP	Ethernet Ring Protection
ERPS	Ethernet Ring Protection Switching
ETNA	Ethernet Transport Networks Architectures of Networking
FIB	Forwarding Information Base
FIB	Forwarding Information Base
FRR	Fast Reroute
FRR-TP	Fast Rerouting-Transport Profile
GFP	Generic Framing Procedure
GFP	Generic Framing Procedure
IETF	Internet Engineering Task Force
IGP	Interior Gateway Protocol
IIH	Intermediate System to Intermediate System Hello
IP	Internet Protocol
IP-FRR	Internet Protocol - Fast Reroute
IS-IS	Intermediate System to Intermediate System
ISP	Internet Service Provider
LAN	Local Area Networks
LCAS	Link Capacity Adjustment Scheme
LSP	Label Switched Path
LSPDU	Link State Protocol Data Unit
LSPID	LSPDU Identifier
MAC	Medium Access Control
MPLS	Multiprotocol Label Switching
MPLS-FRR	Multiprotocol Label Switching - Fast Reroute

MPLS-TP	Multiprotocol Label Switching - Transport Profile
MSTI	Multiple Spanning Tree Instance
MSTP	Multiple Spanning Tree Protocol
OAM	Operations, Administration and Maintenance
OSPF	Open Shortest Path First
OTN	Optical Transport Network
PB	Provider Bridge
PBB	Provider Backbone Bridge
PBB-TE	Provider Backbone Bridge Traffic Engineering
PLSB	Provider Link State Bridging
PSNP	Partial Sequence Number Packet
QoS	Quality of Service
RBridge	Routing Bridge
RCB	Root Controlled Bridging
ROM-FRR	Optimized Multicast Fast Rerouting
RPF	Reverse Path Forwarding
RPFC	Reverse Path Forwarding Check
RPR	Resilient Packet Ring
RRR	Rapid Ring Recovery
RSTP	Rapid Spanning Tree Protocol
SDH	Synchronous Digital Hierarchy
SLA	Service-level Agreement
SONET	Synchronous Optical Networking
SPB	Shortest Path Bridging
SPF	Shortest Path First
STP	Spanning Tree Protocol
TCN	Topology Change Notification
TDM	Time-division Multiplexing
TRILL	Transparent Interconnection of Lots of Links
VCAT	Virtual Concatenation
VID	VLAN Identifier
VLAN	Virtual Local Area Networks

Chapitre 1

Introduction

Traditionnellement, les réseaux de transport étaient basés sur une structure de type *TDM*; ainsi les réseaux *SONET/SDH* ont été utilisés pendant longtemps comme la principale technologie de transport dans le cœur du réseau. Le principal service sur ces réseaux était le transport de circuits de voix qui utilisaient une hiérarchie TDM [1].

Avec la croissance des applications *Internet Protocol (IP)* et des services de commutation par paquets tels que l'accès Internet à haut débit, la téléphonie *IP* et réseaux mobiles, le transport de données devient plus important pour les opérateurs de télécommunications. Cependant les réseaux de transport traditionnels *SONET/SDH* sont encore utilisés pour offrir le moyen de transport physique à plusieurs technologies de commutation de paquets. En principe une technologie *TDM* n'est pas très efficace pour le transport de données, ainsi des mécanismes d'adaptation comme le *Generic Framing Procedure (GFP)*, *Link Capacity Adjustment Scheme (LCAS)*, *Virtual Concatenation (VCAT)* et la technologie *Optical Transport Network (OTN)* ont été développés pour adapter la technologie *TDM* à la transmission de paquets. L'utilisation d'une technologie de commutation par paquets dans le réseau de transport présente certains avantages par rapport aux réseaux *TDM* traditionnels à cause du multiplexage statistique [2]; pour cette raison il existe déjà un intérêt à l'utiliser comme remplacement à technologie *SONET/SDH* dans les réseaux de transport [3]. Il existe plusieurs technologies qui utilisent le multiplexage statistique, telles que le *Asynchronous Transfer Mode (ATM)*, le *IP*, le *MPLS* et la technologie Ethernet qui a reçu beaucoup d'attention à cause de son omni-présence, son coût, sa simplicité et sa haute vitesse [4].

Les réseaux *TDM* traditionnels présentent des caractéristiques de performance adaptées au transport de services en temps réel. Ainsi historiquement ces réseaux ont été utilisés largement pour le transport des services de téléphonie, et certains standards ont été créés. Avec le développement des technologies qui utilisent le multiplexage statistique pour le transport, il a eu une nécessité d'adapter ces technologies pour qu'elles présentent les mêmes caractéristiques de performance que les réseaux *TDM*. Dans la littérature le terme « Classe Opérateur » (*Carrier-class* ou *Carrier-grade*) est utilisé pour décrire un réseau, équipement ou service avec des propriétés des réseaux de transport des gros opérateurs de télécommunications. La qualité de service (*Quality of Service (QoS)*), la robustesse, le recouvrement rapide et la gestion de services sont présentés dans [5] comme des propriétés caractéristiques d'un réseau de classe opérateur. Une description objective de ces propriétés n'est pas universellement adoptée, pourtant la haute disponibilité de 99,999% (« cinq neuf ») et un temps de recouvrement de 50 ms est cité dans plusieurs textes comme standard pour les réseaux de classe opérateur [6] [7] [8].

Donc, les réseaux de transport de nouvelle génération doivent avoir certaines caractéristiques spécifiques pour qu'ils puissent offrir des services aux clients avec la qualité désirée et aussi offrir aux opérateurs un réseau de coût réduit en termes d'implémentation et de gestion. La technologie utilisée pour le transport doit disposer de nouveaux mécanismes pour augmenter sa performance [9], ainsi des mécanismes de *QoS*, d'ingénierie du trafic, de gestion et de recouvrement équivalents à ceux existant dans les réseaux traditionnels, tels que les réseaux *SONET/SDH*, ont été développés. La création de mécanismes pour doter la nouvelle génération Ethernet de ces attributs présente un défi pour les nouvelles technologies [10] [11]. À travers sa version de classe opérateur, la technologie Ethernet est apparue comme une réelle alternative aux réseaux de transport traditionnels et l'état de l'art de la technologie Ethernet utilisé présentement (telles que les technologies *Provider Bridge (PB)*, *Provider Backbone Bridge (PBB)*, *SPB*, *Provider Backbone Bridge Traffic Engineering (PBB-TE)*, *Multiprotocol Label Switching - Transport Profile (MPLS-TP)*, *Resilient Packet Ring (RPR)* et *Ethernet Ring Protection (ERP)* présente des mécanismes qui permettent la création d'un réseau avec plusieurs des attributs désirés pour un réseau de classe opérateur. Dans le domaine de la recherche, des projets tels que *Ethernet Transport Networks Architectures of Networking (ETNA)* [12] et *100 Gbit/s Carrier-Grade Ethernet Transport Technologies (100-GET)* [13] sont des exemples d'efforts qui ont été réalisés avec l'objectif de créer un réseau Ethernet de classe opérateur.

La combinaison des développements apportés par certaines nouvelles technologies telles que l'utilisation d'un protocole de routage à état de liens dans le plan de contrôle utilisé par les technologies TRILL et SPB [14] [15], le format de trames utilisés dans les normes IEEE 802.1ad et 802.1ah [16] [17] dans le

plan de données et des mécanismes de gestion du réseau tels que le 802.1ag (Connectivity Fault Management (CFM))/Y.1731 [18] [19] peuvent apporter une partie des caractéristiques désirées pour un réseau de classe opérateur. Toutefois, pour obtenir un réseau qui combine les avantages des réseaux Ethernet avec la performance des réseaux TDM traditionnels, il est nécessaire également de développer de nouveaux mécanismes, comme par exemple l'amélioration du temps de convergence du réseau à travers un mécanisme de réacheminement rapide *Fast Reroute (FRR)*.

Les réseaux de transport sont sujets à une série de problèmes qui peuvent causer des interruptions du trafic des utilisateurs si le réseau n'offre pas des mécanismes de recouvrement efficaces. L'augmentation des services de temps réel et l'adoption des contrats de type *Service-level Agreement (SLA)*, qui exigent une meilleure performance du réseau par rapport aux services de données traditionnels, ont augmenté l'intérêt pour le développement de mécanismes qui puissent éviter les interruptions ou minimiser les problèmes causés par les défaillances. Dans un réseau Ethernet traditionnel, la robustesse du réseau est garantie par le protocole de contrôle et le temps de recouvrement dépend du temps de convergence de ce protocole; ainsi en cas d'une défaillance dans le réseau, le mécanisme de recouvrement doit créer une nouvelle topologie logique sans provoquer l'occurrence de boucles. Cependant, ce mécanisme peut présenter un temps de recouvrement assez élevé; c'est le cas du *STP* utilisé dans les premières versions de la norme IEEE 802.1D. Pour résoudre le problème du temps de convergence élevé, le *RSTP* a été créé mais ce protocole n'offre pas un recouvrement rapide au niveau d'un réseau de classe opérateur. Plusieurs solutions adoptées présentement par les technologies *« Carrier Ethernet »* sont similaires aux solutions utilisées dans les réseaux de transport par paquets traditionnels. Ainsi, les technologies *PBB-TE* et *MPLS-TP*, qui utilisent des circuits orientés connexion pour l'établissement des chemins dans le réseau, utilisent des mécanismes de protection similaires à ceux utilisés dans les réseaux *MPLS* et SONET/SDH. Le *Internet Engineering Task Force (IETF)* a décrit deux méthodes de réacheminement rapide pour la protection du trafic: P2MP *Fast Rerouting-Transport Profile (FRR-TP)* et *Optimized Multicast Fast Rerouting (ROM-FRR)*. Certaines technologies utilisent une architecture physique en anneau pour garantir un temps de recouvrement rapide. C'est le cas des technologies *RPR*, *Ethernet Ring Protection Switching (ERPS)* (ITU-T 8032), *Ethernet Automatic Protection Switching (EAPS)* et *Rapid Ring Recovery (RRR)* (proposé par [20]). Pourtant en utilisant une architecture en anneau ces technologies imposent une limite au niveau de la mise à l'échelle du réseau. D'autres solutions comme celles décrites en [21] [22] et [23] utilisent les arbres multiples du protocole *MSTP* comme alternative en cas de défaillance. Dans le cadre des réseaux Ethernet contrôlés par un protocole à état de liens nous n'avons pas trouvé dans

notre revue de littérature l'existence d'un mécanisme de réacheminement rapide spécifique pour ce type de réseau, et cela a donc été une des motivations pour le développement de ce mécanisme dans notre travail.

Les objectifs de notre recherche sont de:

— Réaliser un état de l'art des technologies dérivées d'Ethernet et identifier les avantages de leur utilisation pour le transport;
— Examiner les technologies utilisées pour l'Ethernet de nouvelle génération et les mécanismes de recouvrement;
— Analyser l'impact de l'utilisation du protocole *IS-IS* sur le temps de recouvrement du réseau;
— Proposer un mécanisme de recouvrement rapide pour un réseau Ethernet contrôlé par le protocole *IS-IS*.

Notre travail est structuré de la façon suivante:

— Chapitre 1: Ce chapitre a décrit l'évolution des technologies de transport et l'importance de la robustesse du réseau;
— Chapitre 2: Le deuxième chapitre fait une description des technologies Ethernet traditionnelles et de nouvelle génération, ainsi que des protocoles de contrôle utilisés dans ces réseaux.
— Chapitre 3: Dans ce chapitre, nous présentons une description des mécanismes de recouvrement utilisés dans les réseaux de transport et aussi une analyse du temps de recouvrement des réseaux Ethernet traditionnels et des réseaux contrôlés par un protocole à état de liens.
— Chapitre 4: Dans le quatrième chapitre, nous faisons un survol des mécanismes de recouvrement rapide déjà utilisés dans les réseaux *IP* et du problème de formation de boucles causé par le changement de la topologie physique du réseau.
— Chapitre 5: Dans ce chapitre nous présentons un mécanisme de recouvrement rapide pour les réseaux Ethernet contrôlés par un protocole à état de liens et faisons une analyse de performance de ce mécanisme à travers des simulations.
— Chapitre 6: Le sixième chapitre est la conclusion du travail.

À travers l'étude et les analyses présentées, ce travail est une contribution au développement d'un réseaux Ethernet de nouvelle génération. L'utilisation des nouvelles technologies avec le mécanisme de recouvrement rapide proposé permet la création d'un réseau qui possède la robustesse exigée pour les réseaux de transport de classe opérateur.

Chapitre 2

Evolution des technologies Ethernet

La technologie Ethernet est la plus utilisée dans les réseaux locaux. Son coût et sa simplicité sont des facteurs qui ont fait augmenter sa popularité. Sa capacité d'expansion et d'adaptation sont des éléments essentiels pour son utilisation dans différents réseaux et architectures et aujourd'hui elle est employée aussi dans les réseaux métropolitains et étendus. La technologie Ethernet de base est décrite dans la norme IEEE 802.3 (méthode d'accès *Carrier Sense Multiple Access With Collision Detection (CSMA/CD)* et des spécifications pour la couche physique Ethernet) et dans la norme IEEE 802.1D (architecture pour interconnexion des réseaux locaux en utilisant des « *MAC Bridges* »). La popularité, la simplicité et le coût des réseaux Ethernet ont fait augmenter l'intérêt pour son utilisation dans les réseaux étendus, ainsi que des mécanismes tels que le *Q-in-Q/MAC-in-MAC* (IEEE 802.1ad/802.1ah) qui permettent d'améliorer la mise à l'échelle, les mécanismes de *Operations, Administration and Maintenance (OAM)* IEEE 802.1ag/ ITU-T Y.1731 qui améliorent la robustesse du réseau et les mécanismes de synchronisation (IEEE 1588 et ITU-T Synchronous Ethernet - SyncE) qui permettent la distribution d'une horloge de référence à travers le réseau. Au niveau des protocoles de contrôle, le *RSTP* est le protocole de contrôle le plus utilisé dans les réseaux Ethernet traditionnels. Pourtant il existe d'autres solutions utilisées pour le contrôle des réseaux de nouvelle génération. Le protocole *IS-IS* est un exemple. Dans les prochaines sections nous présentons une description des technologies Ethernet et des protocoles utilisés dans le plan de contrôle. À la fin du chapitre nous faisons une analyse du temps de recouvrement des protocoles *RSTP* et *IS-IS* dans différentes topologies de réseau.

2.1 Ethernet traditionnel

La norme IEEE 802.1D spécifie l'opération des commutateurs *Medium Access Control (MAC)* utilisés pour l'interconnexion de réseaux locaux [24]. Un commutateur réseau permet la séparation des domaines de collision où chaque port correspond à un domaine de collision unique. Dans cette norme est aussi défini le protocole *STP*, conçu pour résoudre les problèmes causés quand les réseaux locaux utilisent des connections redondantes. La norme IEEE 802.1Q spécifie l'opération de commutateurs qui supportent les réseaux locaux virtuels (*Virtual Local Area Networks (VLAN)*s) [25]. Selon la norme IEEE 802.1D, dans un réseau connecté par des commutateurs on a un seul domaine de diffusion (*broadcast*), et ainsi tous les éléments du réseau reçoivent les datagrammes transmis. Un *VLAN* est un groupe de segments logiques Ethernet qui partagent le même réseau physique, et dont les équipements (stations) dans chaque groupe peuvent communiquer comme si elles opéraient dans un réseau privé. On peut ainsi définir un *VLAN* pour un certain groupe d'équipements et ce groupe aura son propre domaine de diffusion. Pour établir l'appartenance des équipements à un *VLAN* spécifique, on peut déterminer les ports du commutateur ou les adresses *MAC* qui font partie du groupe. Les commutateurs ajoutent une étiquette à l'entête de la trame Ethernet pour identifier les différents réseaux virtuels. Pour la définition des classes de service dans une trame Ethernet 3 bits de l'étiquette *VLAN* 802.1Q (code de priorité) sont utilisés; de cette façon on peut avoir jusqu'à 8 classes de service avec des priorités de trafic différentes. La figure 2.1 montre les formats des trames 802.1D et 802.1Q.

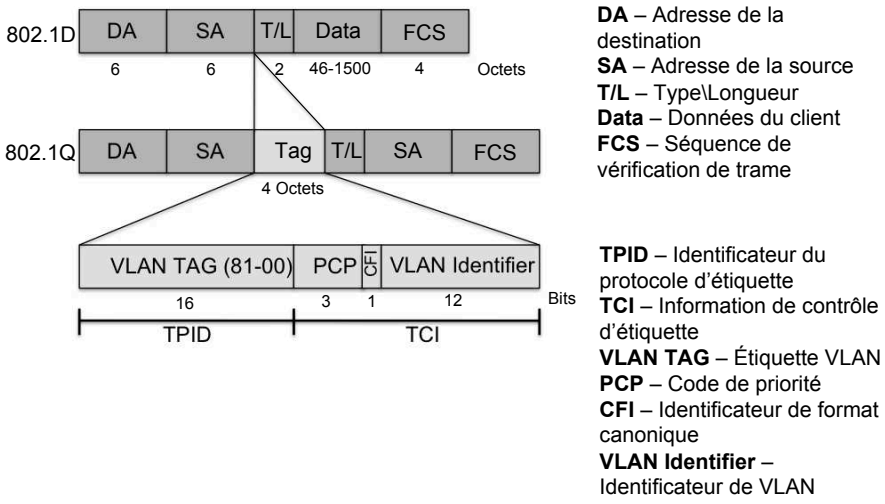


FIGURE 2.1 – Format de trame 802.1D/802.1Q [26] [25]

2.2 Ethernet de Classe Opérateur

Pour que la technologie Ethernet continue à évoluer et puisse être adoptée comme technologie de transport, elle doit être capable d'offrir les caractéristiques d'un réseau de classe opérateur. Le *Metro Ethernet Forum* (MEF) définit le *« Carrier Ethernet »* comme « un service omniprésent de classe opérateur, normalisé et caractérisé par les cinq attributs qui le différencient de l'Ethernet basée sur *Local Area Networks (LAN)*s » [27]. Les cinq attributs définis par le MEF pour le *« Carrier Ethernet »* sont les suivants:

— Services normalisés;
— Mise à l'échelle;
— Fiabilité;
— Qualité du Service (QoS);
— Gestion des services.

Dans la définition du MEF, le *« Carrier Ethernet »* est un service de transport de trames Ethernet avec les attributs cités, en permettant la création d'un réseau Ethernet étendu. Pourtant, dans cette définition, la technologie utilisée pour le transport des trames Ethernet n'est pas forcément celle définie par les normes de la famille IEEE 802.1/802.3 puisque les services de transport de trames Ethernet peuvent utiliser aussi des technologies de transport à longue portée.

Du point de vue de l'Ethernet normalisé par le IEEE le développement des technologies *« Carrier Ethernet »* comme technologie de transport a été fait progressivement à partir des technologies déjà existantes, d'abord pour son utilisation dans l'accès et agrégation et plus récemment pour le cœur du réseau de transport. Une série de normes ont été créées pour ajouter des mécanismes qui permettent son utilisation dans les réseaux de transport. Cela rend possible l'utilisation du mécanisme de création de classes de service et de réseaux virtuels de la norme IEEE 802.1Q, pour offrir la *QoS* ainsi que différents *VLAN*s dans le réseau. Aussi, les mécanismes *« Q-in-Q »* et *« MAC-in-MAC »*, les technologies 802.1ad (*Provider Bridges*) et 802.1ah (*Provider Backbone Bridges*) améliorent la mise à l'échelle et facilitent la gestion des services. Pourtant, bien que ces technologies fassent partie de la famille *« Carrier Ethernet »*, elles ne possèdent pas tous les attributs définis par le MEF. Particulièrement, concernant les attributs de fiabilité et gestion de services, le plan de contrôle n'est pas encadré dans les normes qui décrivent ces technologies. Dans la suite, nous présentons une brève description des technologies *« Carrier Ethernet »* les plus populaires.

2.2.1 Technologies « Carrier Ethernet »

IEEE 802.1ad (PB)

La norme IEEE 802.1ad est un document modificatif de la norme IEEE 802.1Q. Le PB modifie le format de trame de base pour créer une séparation entre les domaines de *VLAN* des clients et du fournisseur de service. Dans la norme IEEE 802.1Q l'étiquette est un champ de 4 octets qui contient 12 bits pour identifier les différents *VLAN*s (*VLAN* ID), permettant ainsi la création de 4096 instances (dont 4094 utilisables). Du point de vue du fournisseur ce nombre est limité, pas seulement à cause du nombre de *VLAN*s possibles mais aussi parce que différents clients qui utilisent déjà un réseau 802.1Q ne peuvent pas utiliser les mêmes *VLAN* IDs. Dans le PB les *VLAN*s du réseau administré par le client reçoivent le nom de *Customer VLANs* (C-*VLAN*) et celles du réseau administré par le fournisseur sont les *Service VLANs* (S-*VLAN*s). Le domaine de service est créé à travers l'ajout d'un nouveau champ dans la trame, le *Service TAG* (S-TAG). Dans le S-TAG il existe un champ de 12 bits pour la définition des S-*VLAN*s du domaine de fournisseur de services. Pourtant le mécanisme d'apprentissage des commutateurs fait que dans un réseau PB, toutes les adresses *MAC* des clients doivent être connues par le réseau du fournisseur. Cela pose un problème de mise à l'échelle et également de sécurité puisqu'un client peut envoyer un nombre illimité d'adresses vers le réseau de service, ce qui peut provoquer l'épuisement des tables et l'augmentation des diffusions de trames dans le réseau. La figure 2.2 montre les formats des trames 802.1ad.

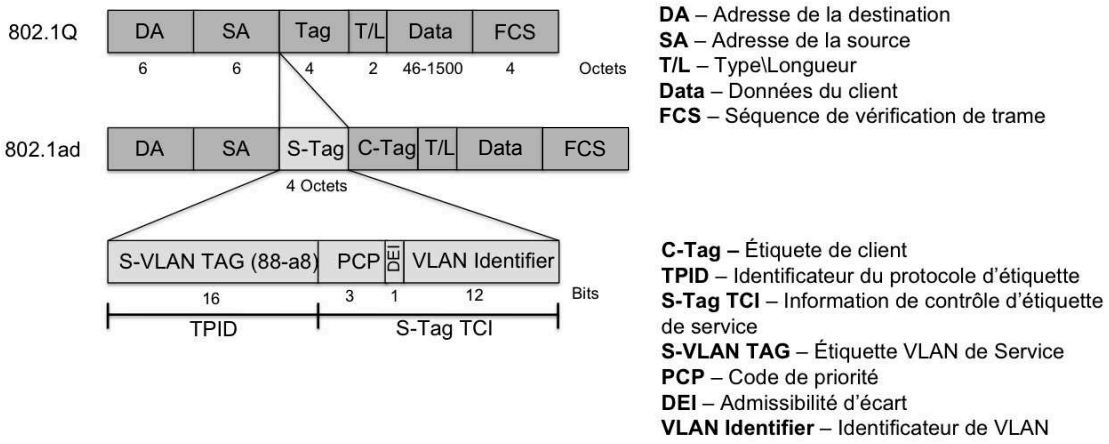
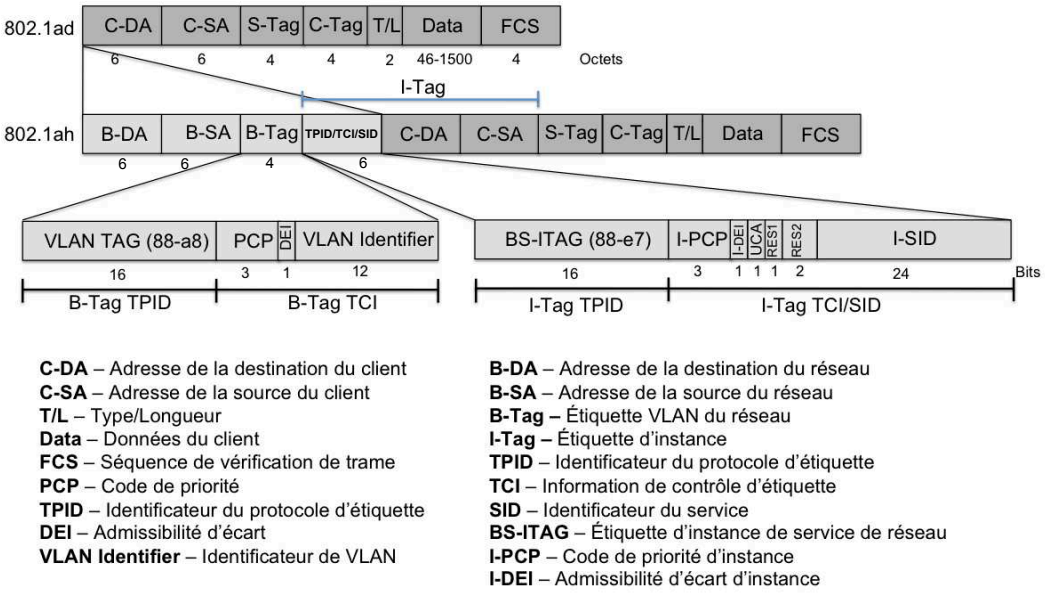


FIGURE 2.2 – Format de trame 802.1ad [26] [16]

IEEE 802.1ah (PBB)

La norme IEEE 802.1ah modifie la norme IEEE 802.1Q de façon à résoudre le problème de mise à l'échelle existant dans le PB. Ainsi le PBB spécifie un nouveau champ nommé *Backbone Service Instance tag* (I-TAG) qui encapsule les adresses du client et un nouveau champ de 24 bits, le *Backbone Service Instance Identifier* (I-SID) qui crée des instances de service. Un champ *Backbone VLAN tag* (B-TAG) est aussi défini; ce champ joue le rôle du S-TAG des réseaux PB. Dans un réseau PBB, il existe une séparation entre les domaines d'adresses du client et du fournisseur, et ainsi les champs B-SA et B-DA représentent les adresses *MAC* utilisées dans le réseau du fournisseur de transport et les champs C-SA et C-DA représentent les adresses du client. La figure 2.1 montre les formats des trames 802.1ah.



C-DA – Adresse de la destination du client
C-SA – Adresse de la source du client
T/L – Type/Longueur
Data – Données du client
FCS – Séquence de vérification de trame
PCP – Code de priorité
TPID – Identificateur du protocole d'étiquette
DEI – Admissibilité d'écart
VLAN Identifier – Identificateur de VLAN

B-DA – Adresse de la destination du réseau
B-SA – Adresse de la source du réseau
B-Tag – Étiquette VLAN du réseau
I-Tag – Étiquette d'instance
TPID – Identificateur du protocole d'étiquette
TCI – Information de contrôle d'étiquette
SID – Identificateur du service
BS-ITAG – Étiquette d'instance de service de réseau
I-PCP – Code de priorité d'instance
I-DEI – Admissibilité d'écart d'instance

FIGURE 2.3 – Format de trame 802.1ah [26] [17]

Les normes PB et PBB représentent une évolution en termes de mise à l'échelle et de gestion des services, mais pas en termes de fiabilité conforme aux standards des réseaux de transport traditionnels puisque ces technologies n'impliquent pas de mécanismes de contrôle améliorés par rapport à ceux utilisés dans les *LAN*s. Dans un réseau PB/PBB l'utilisation du *RSTP*/*MSTP* dans le plan de contrôle et les mécanismes de diffusion et apprentissage des commutateurs posent des problèmes pour la performance du réseau [28]. De nouvelles solutions telles que les protocoles *TRILL* (proposé par le IETF) et le *SPB* (proposé dans la norme IEEE 802.1aq) offrent une solution à certains problèmes en utilisant le protocole *IS-IS* pour la construction de la topologie logique du réseau. Pourtant, le mécanisme de recouvrement du protocole *IS-IS* ne garantit

pas un temps de recouvrement idéal conforme aux standards des réseaux de transport de nouvelle génération (< 50 ms) [6]. D'autres technologies telles que le *Resilient Packet Ring* (RPR) défini dans la norme IEEE 802.17 et le *Ethernet Ring Protection* (ERP) défini dans la norme ITU-T G.8032, peuvent être utilisées pour offrir une garantie de recouvrement attendue, mais cependant ces technologies imposent l'adoption d'une topologie en anneau et une limitation du nombre de nœuds dans le réseau [29].

Dans le cœur des réseaux *« Carrier Ethernet »*, le mécanisme de commutation de circuits virtuels est normalement proposé comme solution pour garantir la *QoS* et la fiabilité. Ce mécanisme offre une approche semblable à celle utilisée par la commutation de circuits traditionnelle, où les services sont offerts à travers la construction de circuits point à point. Présentement les deux technologies qui utilisent ce mécanisme sont: le *MPLS-TP* et le *PBB-TE* [30].

MPLS-TP

Le *MPLS-TP* utilise quelques éléments de la technologie *MPLS* comme base pour créer un réseau avec les caractéristiques *Carrier* [31]. L'utilisation des trames *MPLS* dans le plan de données permet que la commutation et le transport des services s'accomplisse de la même façon que dans les réseaux *MPLS* traditionnels. Les mécanismes de d'exploitation, de gestion et de maintenance (OAM) du *MPLS* et d'autres extensions sont aussi utilisés; ces mécanismes possèdent des fonctionnalités similaires à celles du SONET/SDH. Des trames *OAM* sont utilisées pour la surveillance du réseau à plusieurs niveaux. La garantie de la *QoS* et l'ingénierie du trafic sont faites à travers l'utilisation des circuits virtuels orientés connexion, et des mécanismes tels que le contrôle d'admission (CAC) et *« DiffServ »* peuvent être appliqués. Les mécanismes de protection sont aussi définis pour offrir des services similaires aux mécanismes utilisés par le SDH et ainsi garantir le temps de recouvrement maximal de 50 ms (*MPLS* Fast Re-route-FRR).

PBB-TE

La technologie *Provider Backbone Bridges - PBB-TE* (802.1Qay) est une évolution de la famille IEEE 802.1; elle est dérivée du PBB et utilise le même format de trame spécifié par la norme IEEE 802.1ah dans le plan de données, ce qui permet l'isolation des plans du client et du réseau en ce qui concerne l'utilisation des *VLAN*s et des identificateurs *MAC*. Tout comme le *MPLS-TP*, le PBB-TE utilise des circuits virtuels orientés connexion pour garantir la *QoS* et aussi pour faciliter l'ingénierie du trafic [32]. Le mécanisme de

OAM est défini par les normes ITU-T Y.1731 et IEEE 802.1ag, qui présentent une série de fonctions pour la surveillance des fautes et mesure de performance. Les normes ITU-T G.803/G.8032 peuvent être utilisées comme mécanisme de protection.

2.3 Les protocoles de gestion de liens

Le *STP*, le *RSTP* et le *MSTP* sont les protocoles de gestion de liens des réseaux Ethernet traditionnels. Le *STP*, conçu en 1985 par Radia Perlman, a pour objectif principal de permettre la connectivité dans un réseau Ethernet interconnecté par commutateurs à travers la création d'une topologie logique en arborescence. Le protocole/algorithme *STP* construit un chemin logique unique entre les stations éliminant l'occurrence de boucles [24]. Bien que l'algorithme *STP* soit capable de construire cette topologie et d'éviter les boucles, son temps de convergence est trop élevé (pouvant atteindre jusqu'à 50s) [33]. Pour solutionner ce problème, le protocole *RSTP* a été développé; ce protocole est défini dans la norme IEEE 802.1D pour les réseaux Ethernet [24]. En théorie le *RSTP* peut avoir un temps de convergence de quelques millisecondes, mais ce temps est très variable et dépend de la topologie adoptée. Dans certains cas (comme par exemple une défaillance du commutateur racine *« Root Bridge »*) il peut même atteindre l'ordre de quelques secondes [34]. Le *MSTP* est une extension du *RSTP* utilisé pour améliorer la performance du réseau à travers la création de multiples instances de contrôle pour les différentes *VLAN*s. Dans cette section nous décrivons le fonctionnement de ces protocoles.

2.3.1 *STP*

Pour construire la topologie active sans boucle, le *STP* établit différents états de ports. Dans chaque commutateur les ports qui participeront à la transmission et à la réception des trames sont en état de transmission (*Forwarding State*) et les ports qui ne font pas partie de cette topologie logique restent bloqués (*Blocking State*). On choisit également un commutateur racine parmi les commutateurs du réseau. La topologie logique du réseau ainsi construit fonctionne de façon à ce que chaque LAN dans le réseau ait un chemin unique vers le commutateur racine à travers un port désigné (*Designated Port*) qui fait partie d'un commutateur désigné *Designated Bridge* auquel le LAN est connecté.

Pour définir la topologie et choisir le rôle de chaque élément, le protocole utilise des identificateurs pour les commutateurs (*Bridges*) et ports. Chaque commutateur possède un identificateur *Bridge Identifier*

(BridgeID) qui lui est associé. Le BridgeID est composé d'un identificateur de priorité (*Bridge Identifier Priority*) modifiable et d'une addresse (*Bridge Address*), l'adresse *MAC* unique du commutateur qui peut être l'adresse d'un de ses ports (dans la norme l'utilisation de l'adresse la plus basse est recommandée). Les ports possèdent une valeur de coût de chemin vers le commutateur racine (*Root Path Cost*), un coût de chemin (*Path Cost*) avec des valeurs *defauts* selon la capacité du lien, et un identificateur *Port Identifier*. Dans le réseau, le commutateur avec le meilleur BridgeID (valeur numérique la plus basse) est choisi comme commutateur racine. Le *Root Path Cost* associé à chaque port est défini comme l'addition des valeurs du *Path Cost* de chaque port dans le chemin avec le coût minimal jusqu'au commutateur racine. Pour chaque LAN, le *Designated Port* est défini comme le port avec la valeur du *Root Path Cost* la plus basse; dans le cas de deux ports avec le même coût, le *Bridge Identifier* et le *Port Identifier* sont utilisés pour les départager. La transmission des informations sur la topologie est faite à travers la transmission de trames appelées *Bridge Protocol Data Units (BPDU)* entre les commutateurs; l'algorithme *Spanning Tree* utilise des informations transportées pour définir le rôle et l'état des ports du commutateur. La figure 2.4 montre un exemple de topologie avec 3 commutateurs dont le commutateur 1 est le commutateur racine. Le calcul de la topologie active est fait de la façon suivante:

— Chaque commutateur, au début de sa routine, prend le rôle de commutateur racine et génère des « messages de configuration » (*Configuration BPDUs*). Les *BPDU*s sont transmis dans un intervalle fixe (*Hello Time*) et portent les valeurs de *Root Identifier*, du *Root Path Cost* et de la priorité du commutateur et des ports. À ce moment tous les ports du commutateur racine sont des *Designated Ports* (port qui connecte le LAN au commutateur racine). La figure 2.4(a) montre cette étape.

— La priorité relative de chaque commutateur par rapport aux autres est calculée à partir de la comparaison entre les valeurs du commutateur et les valeurs reçues à travers les *BPDU*s. Un commutateur qui reçoit une *BPDU* avec une information de priorité relative supérieure (valeur numérique plus basse) retransmet cette information aux autres *LAN*s. Dans la figure 2.4(b) les commutateurs 2 et 3 considèrent l'information transmise par 1 comme de priorité supérieure.

— Un commutateur qui reçoit une information inférieure dans un port considéré comme *Designated Port* répond avec des *BPDU*s qui portent leurs propres informations (informations de priorité relative supérieure). Dans la figure 2.4(b) le commutateur 1 retransmet ses *BPDU*s après la réception des *BPDU*s des commutateurs 2 et 3.

— Si la topologie physique est stable, à la fin de la convergence du protocole le réseau possède un seul commutateur racine, et chacun des autres commutateurs possèdent un seul port racine (*Root Port*)

qui offre un chemin unique vers le commutateur racine aux *LAN*s attachés à ses *Designated Ports*. La figure 2.4(b) montre les états des ports à la fin de la procédure et la figure 2.4(c) la topologie logique finale.

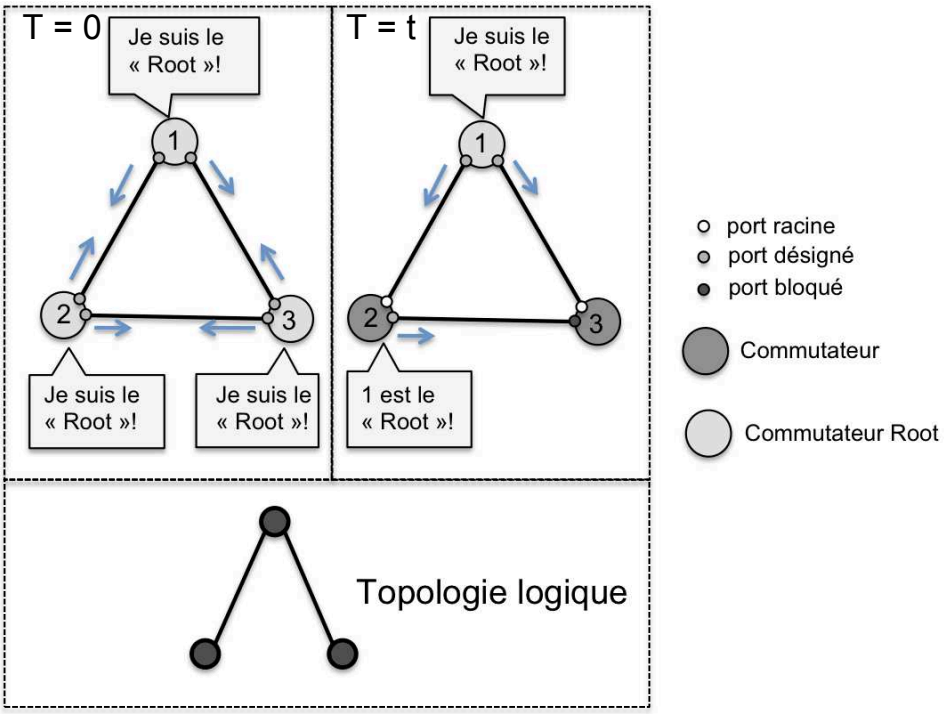


FIGURE 2.4 – Construction de la topologie

Chaque port possède aussi un état d'opération (*Port State*) qui règle la transmission des trames et le mécanisme d'apprentissage d'adresses (*learning*). La topologie active est composée des ports dans l'état *Forwarding*, ces ports transmettent et reçoivent les trames et réalisent l'apprentissage. Les ports qui ne participent pas à la topologie active sont dans l'état *Blocking*. L'algorithme peut changer les états des ports dans le cas de modification de la topologie physique du réseau. Ainsi un port peut passer de l'état *Forwarding* à l'état *Blocking* et vice versa. Pour passer de l'état *Blocking* à l'état *Forwarding*, les ports doivent passer par deux états intermédiaires, soit *Listening* et *Learning*. Ce mécanisme est utilisé pour éviter l'occurrence de boucles pendant la période de convergence. Un administrateur peut attribuer aussi l'état *Disabled* à un port pour qu'il ne participe pas à la topologie.

Les trames *BPDU* utilisent une identification de groupe *MAC* spéciale (01:80:C2:00:00:00) comme destination. Elles peuvent transporter des messages de configuration et de changement de topologie. Pour sélectionner le commutateur racine et le meilleur chemin les commutateurs échangent des informations dans

les messages de configuration *(*Configuration Messages). Les informations échangées reçoivent le nom de vecteurs de priorité (*priority vectors*) composés par:

— *Root Bridge Identifier*
— *Root Path Cost*
— *Bridge Identifier*
— *Port Identifier* du port à travers lequel le message a été transmis
— *Port Identifier* du port à travers lequel le message a été reçu

Ces informations sont utilisées par l'algorithme dans chaque commutateur pour choisir le meilleur message parmi tous les messages reçus et définir les états des ports, ce qui va définir la topologie du réseau.

Dans un réseau qui utilise le *STP* seulement, le commutateur racine transmet des *BPDU*s, les autres renvoient seulement les *BPDU*s reçus (après la modification de quelques informations champs spécifiques). Le *Router Bridge* transmet des *BPDU*s à travers ses *Designated ports* dans un intervalle fixe configuré dans le commutateur par le variable *Hello Time* (2s par défaut).

Il existe aussi deux autres types de *BPDU*s:

— *Topology Change Notification (TCN)*
— *Topology Change Notification Acknowledgment*

Ces *BPDU*s sont utilisées dans les cas de changement de la topologie physique du réseau.

S'il survient un problème dans un lien ou commutateur, le *STP* réagit à travers les mécanismes pour recalculer la topologie à travers la procédure suivante:

— Les *BPDU*s de configuration (*Configuration BPDUs*) possèdent un temps de vie limité *Message Age* qui est transmis avec le message et est incrémenté a chaque saut; après sa réception le temps en secondes de stockage de l'information est aussi compté. Si le temps de vie est écoulé pour un message reçu dans un *Root port* cela signifie que le commutateur a perdu son chemin vers le commutateur racine et doit chercher un nouveau chemin ou devenir le commutateur racine. Le temps d'expiration est configuré pour le commutateur à travers la variable *Max Age* (20s par défaut).
— Si un port dans l'état *Blocking* doit passer à l'état *Forwarding*, il doit attendre un certain temps pour que tous les autres commutateurs reçoivent l'information actualisée; ce temps entre les états

Listening et *Learning* est configuré dans le commutateur en utilisant le paramètre *Forward Delay* (15s par défaut).

Un commutateur avec une configuration par défaut pour les paramètres *Max Age* et *Forward Delay* doit attendre un temps total maximal de $T = 2 \times Forward\ Delay + Max\ Age = 50s$ et un temps minimal de $T = 2 \times Forward\ Delay = 30s$ pour s'adapter à la nouvelle topologie (cette variation est causée par l'attente d'expiration du temps de vie d'un message).

Le mécanisme d'apprentissage utilisé par les commutateurs permet que les informations sur la localisation des stations soient stockées dans les tables d'adresses et ainsi d'éviter la diffusion des trames. Dans l'opération normale du réseau il n'y a pas de changements fréquents de localisation et les tables utilisent un temps d'expiration relativement long pour retenir ces informations.

2.3.2 *RSTP*

Comme pour le *STP*, le *RSTP* configure l'état des ports de chaque commutateur et établit une connectivité stable entre les *LAN*s individuels attachés, tout en assurant une topologie logique libre de boucles. Le *RSTP* a été conçu pour éliminer le problème du temps de convergence élevé du protocole *STP*. La norme établit que la topologie active doit se stabiliser dans une courte période de temps (non spécifié), en diminuant le temps dans lequel le service est indisponible. Pour réaliser cette tâche, des modifications à l'opération du *STP* ont été faites avec l'addition de nouveaux mécanismes. Le mécanisme utilisé par le *RSTP* fonctionne de la façon suivante:

— Si un commutateur cesse de recevoir des *BPDU*s, il attend un intervalle de 3 *Hello Time* pour déterminer que le port a perdu la communication. Si le commutateur perd sa communication avec le commutateur racine il doit chercher un nouveau chemin à travers son *Alternate Port* ou, si le chemin est inexistant, il va lui-même devenir le commutateur racine.
— Un port dans l'état *Discarding* peut passer à l'état *Forwarding* immédiatement, dans le cas d'un *Alternate Port* qui devient un *Root Port* après que l'ancien soit passé à l'état *Disabled* ou est devenu un *Alternate Port*.
— Un *Designated Port* qui n'est pas dans l'état *Forwarding* peut proposer la transition d'état à son port correspondant dans le commutateur voisin; la transition est faite immédiatement après la réception d'un message d'accord.

Le mécanisme de calcul de la topologie dans le *RSTP* n'a pas été modifié par rapport au *STP*, ainsi le choix du commutateur racine, des *Root Ports* et *Designated* ports est fait de la même façon. Pourtant, dans le *RSTP*, il existe deux nouveaux rôles pour les ports:

— *Alternate Port* - offre un chemin alternatif vers le commutateur racine;
— *Backup Port* - offre un chemin alternatif au chemin offert par un *Designated Port* attaché à un même LAN.

Dans le fonctionnement normal d'un réseau avec une topologie logique stabilisée, ces ports restent bloqués; dans ce cas ils sont dans l'état *Discarding*.

Contrairement au *STP*, où seulement le commutateur racine crée des *BPDU*s, dans le *RSTP* tous les commutateurs transmettent des *BPDU*s à chaque intervalle *Hello Time* et aussi dans le cas de changement de l'information transportée. Pourtant, il existe une limite à la quantité de *BPDU*s transmises par seconde. Cette limite est configurée à travers le paramètre *Transmit Hold Count* (6s par défaut). Le temps de convergence du *RSTP* peut arriver à quelques centaines de millisecondes en fonction des conditions du réseau [33].

2.3.3 *MSTP*

Le *MSTP* est défini dans la norme IEEE 802.1Q. Ce protocole/algorithme permet aux trames qui appartiennent à différents *VLAN*s de suivre différents chemins dans le réseau. Le protocole est compatible avec le *STP* et le *RSTP*, donc un réseau peut avoir des commutateurs qui communiquent au moyen de ces différents protocoles. Un réseau *MSTP* peut être constitué de plusieurs régions. Dans chaque région les trames possédant des *VLAN Identifier (VID)* spécifiques sont attribuées à une *Multiple Spanning Tree Instance (MSTI)* indépendante qui utilise l'algorithme *spanning tree* pour le calcul de la topologie active de ces *VLAN*s. Dans une région *MSTP* chaque *MSTI* utilise des paramètres internes (*Regional Root Identifier*, *Internal Root Path Cost*, *Internal Port Path Cost*) pour construire son arbre. Les différentes régions peuvent être interconnectées et peuvent aussi connecter des commutateurs qui utilisent le *STP*/*RSTP*; le *MSTP* voit ce réseau comme une seule instance qui s'appelle *Common and Internal Spanning Tree* (CIST) composé de tous les commutateurs du réseau. Le commutateur *CIST Root* est élu parmi tous les commutateurs et est celui qui possède la meilleure valeur *BridgeID*. Dans chaque région interconnectée il existe un ou plusieurs commutateurs de bordure qui connecte les régions. Celui qui possède le meilleur chemin vers la racine CIST est élu comme le CIST *Regional Root*. Le meilleur chemin est calculé en utilisant le *External Root Path Cost*. Le *Common*

Spanning Tree est un arbre qui connecte toutes les régions et commutateurs qui n'utilisent pas le *MSTP*; cet arbre voit chaque région comme si elle était un seul commutateur. Les figures 2.5 et 2.6 montrent un diagramme des topologies physiques et logiques d'un réseau *MSTP* composé de plusieurs régions.

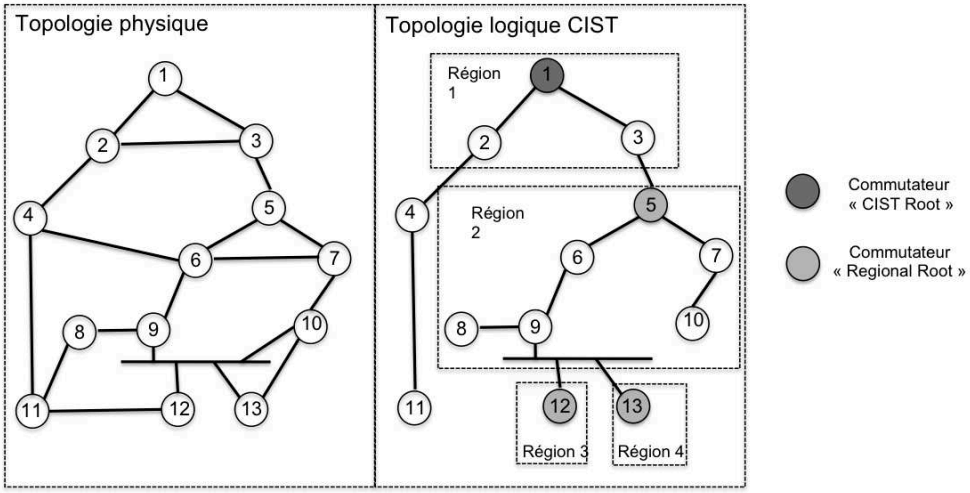


FIGURE 2.5 – Topologie *MSTP*

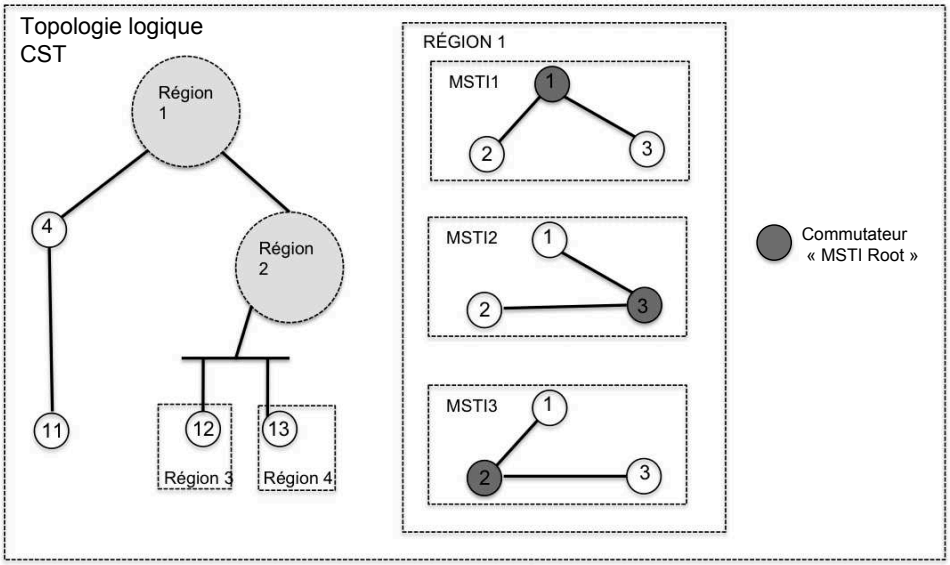


FIGURE 2.6 – Topologie *MSTP* (CST)

2.4 Protocoles de contrôle dans les réseaux Ethernet de nouvelle génération

Les nouvelles technologies Ethernet ont apporté plusieurs innovations pour améliorer la performance du protocole originel. Au niveau du plan de données l'utilisation d'identifiants de *VLAN*s spécifiques dans

le réseau de l'opérateur (*« Q-in-Q »*) est décrite dans IEEE 802.1ad, l'isolation du réseau du client et de l'opérateur est aussi complétée par l'utilisation d'un espace d'adresses *MAC* diffèrent (*« MAC in MAC »*), le mécanisme utilisé par le IEEE 802.1.ah. Pourtant, ainsi que les réseaux Ethernet traditionnels qui utilisent le *RSTP*/*MSTP*, ces technologies n'apportent pas de changements dans le plan de contrôle. Les limites d'utilisation des liens sont un problème des protocoles *STP*/*RSTP*, où le mécanisme de blocage des ports pour la construction des arbres est un limitant pour l'utilisation de toute la capacité du réseau. Pour contourner ce problème il existe des solutions telles que l'utilisation du *MSTP*. Ce protocole permet en effet l'utilisation de *VLAN*s pour distribuer le trafic entre plusieurs arbres qui ont comme racine différents commutateurs, ainsi les points de concentration du trafic de chaque *VLAN* peuvent être utilisés comme racine. Le *MSTP* a été proposé comme protocole de contrôle de la technologie *SPB*; dans cette solution chaque commutateur du réseau possède une *MSTI* et fonctionne comme commutateur racine [35]. Le protocole *Per-VLAN Spanning Tree* (PVST) est une solution propriétaire de Cisco qui utilise aussi ce mécanisme. Le protocole *Viking* utilise une approche similaire et il est proposé en [23] comme une solution aux problèmes d'ingénierie du trafic. Pourtant le mécanisme de recouvrement du *MSTP* est le même que celui du *RSTP*, et chaque instance du *MSTP* fonctionne comme un mécanisme *RSTP* indépendant pour la construction de la nouvelle topologie. Contrairement à ces technologies, le *PLSB*, *TRILL* et *SPB* utilisent le protocole *IS-IS* dans le plan de contrôle pour la construction de la topologie logique en arborescence. Le *SPB* est spécifié dans une modification à la norme IEEE 802.1Q et utilise le protocole *IS-IS* avec des extensions (*IS-IS-SPB*) pour la construction des *Shortest Path Trees (SPT)*. Les SPTs construits dans chaque commutateur *SPB* possèdent les chemins les plus courts vers les autres commutateurs du réseau et ce chemin est utilisé dans les deux directions. Le *SPB* possède un mécanisme pour briser l'égalité entre des chemins calculés par l'algorithme; de cette façon le meilleur chemin calculé par deux commutateurs sera toujours le même. Comme mesure de prévention des boucles, les tables de routage ne sont peuplées que quand le nœud est synchronisé avec le prochain saut, le mécanisme assurant que la trame sera renvoyée seulement si le voisin est prêt pour l'accepter. Le mécanisme de réduction de problèmes provoqués par les boucles utilise le filtrage des trames qui arrivent aux commutateurs; et ainsi une vérification est faite pour garantir que la trame suive bien le chemin établi (*Source Based Routing*).

2.4.1 Le *IS-IS*

Le protocole *IS-IS* est un protocole de routage de type état de lien, où chaque nœud du réseau *IS-IS* calcule les routes en utilisant l'algorithme du plus court chemin à partir des informations reçues de chacun

des autres nœuds. Ce protocole, défini pour la première fois dans la norme ISO 10589 est largement utilisé dans les réseaux *IP*, notamment dans le coeur du réseau des gros fournisseurs de service Internet [36]. Il s'agit d'un protocole facilement extensible et qui ne dépend pas de l'encapsulation *IP* puisqu'il se trouve directement au dessus de la couche liaison de données, et donc son utilisation ne se restreint pas aux réseaux *IP*, ce qui a été un facteur important pour son choix comme protocole de contrôle des technologies *TRILL* et *SPB*. L'utilisation de l'*IS-IS* dans les commutateurs Ethernet pour la construction des arbres de routage offre des avantages par rapport au *RSTP*, par exemple la meilleur utilisation des ressources et la construction de chemins optimisés entre chaque commutateur du réseau.

Construction de la topologie

Le premier pas pour la construction de la topologie dans un réseau qui utilise le *IS-IS* est la découverte des voisins à travers la procédure de *handshaking* qui peut être accomplie en deux ou trois étapes (*2-way et 3-way handshake*). Deux types de liaisons physiques entre les éléments du réseau peuvent être utilisés: point à point (p2p) et diffusion générale (*broadcast*). Tous les messages utilisés par le protocole *IS-IS* utilisent des identificateurs *MAC* connus (0180:c200:0014 ou 0180:c200:0015), dont les messages *Intermediate System to Intermediate System Hello (IIH)* sont utilisés pour la découverte des voisins. Ces messages sont identifiés à travers un champ spécifique dans l'entête *IS-IS* (*PDU Type*) (les liaisons p2p et *broadcast* utilisent des *PDU Types* différentes). Les *IIH*s sont envoyés à travers l'interface physique par un nœud du réseau (commutateur ou routeur) dans un intervalle de temps régulier et les voisins attendent un temps maximal (*Holding Time*) pour la réception de ce message; si aucun message n'a été reçu après ce temps le voisin est déclaré « mort ». L'intervalle de temps pour l'envoi des messages *IIH* est une fraction du *Holding Time* configuré à travers une constante « *Hello multiplier* ». Par exemple pour une valeur *Holding Time* de 30 secondes (valeur typique), si le *Hello multiplier* est 3 donc l'intervalle d'envoi des messages sera de 10 secondes. Les voisins sont identifiés par une adresse de 6 octets (*Source ID*) transportée par le message *IIH*. Après la procédure de *handshaking* et la découverte des voisins, le réseau est prêt pour l'échange des informations sur la topologie; pour cet échange le *IS-IS* utilise le *Link State Protocol Data Unit (LSPDU)* responsable de la distribution de la base de données (les états des liens vers les autres éléments du réseau). A partir des informations reçues il est donc possible de calculer les chemins à travers le réseau. Puisque toutes les informations sont diffusées à travers le réseau, tous les éléments possèdent la même base de données à la fin du processus. Les *LSPDU*s possèdent un temps de vie maximal et doivent être envoyés régulièrement pour que le réseau maintienne une base de données actualisée. Ils possèdent aussi un numéro de séquence qui est vérifié à la réception et qui

permet de s'assurer que seulement les nouvelles informations soient diffusées. Le *IS-IS* possède aussi des *Protocol Data Units (PDU)* pour la synchronisation des bases de données: le *Complete Sequence Number Packet (CSNP)* (envoyé régulièrement et qui contient une liste des *LSPDU*s existants dans la base de données) et le *Partial Sequence Number Packet (PSNP)* (utilisé pour solliciter des *LSPDU*s ou confirmer sa réception). Pour maintenir une base de données distribuée à travers tout le réseau, les protocoles à état de liens utilisent le mécanisme d'inondation (*flooding*). Ainsi les *LSPDU*s sont envoyés à partir du nœud d'origine vers tous leurs voisins et un nœud qui reçoit un *LSPDU* doit vérifier son identité à travers le *LSPDU Identifier (LSPID)* (qui identifie l'origine) et le numéro de séquence; dans le cas d'un *LSPDU* plus jeune ou inexistant dans la base de données locale, il sera installé et rediffusé à travers tous les liens sauf celui par lequel le *LSPDU* a été reçu. Si le *LSPDU* est plus ancien ou s'il existe déjà dans la base de données il sera ignoré. La figure 2.7 montre un exemple du fonctionnement du mécanisme d'inondation. Le nœud A envoie un nouveau *LSPDU* vers ses voisins B et C; E reçoit deux copies de ce même *LSPDU*, mais seulement une sera acceptée et installée dans sa base de données; il doit donc ignorer celle qui arrive en dernier et renvoyer la première à son voisin G; G reçoit trois copies du même *LSPDU*, et installe la première reçue dans sa base de données, ce qui termine la diffusion de cet *LSPDU* à travers le réseau est donc finie.

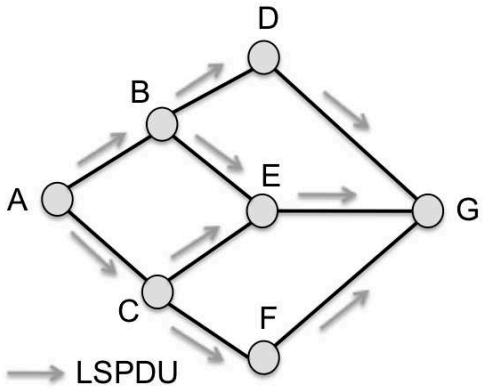


FIGURE 2.7 – Mécanisme d'inondation

2.4.2 Technologies Ethernet basées sur le *IS-IS*

Le *TRILL*

Le protocole *TRILL*, décrit dans un *Internet Draft* de l'IETF [37], a été conçu pour contourner les problèmes des protocoles *STP* tout en conservant les caractéristiques du réseau commuté [14]. Ainsi, il utilise le protocole *IS-IS* pour construire les tables de commutation qui vont déterminer les chemins entre deux

éléments du réseau. Les commutateurs qui composent ce réseau et implémentent le protocole *TRILL* sont nommés *Routing Bridge (RBridge)*. Le chemin optimal entre deux éléments du réseau est calculé à travers les informations du protocole à état de liens, les trames sont délivrés selon l'adresse de destination et celles qui ne possèdent pas d'adresse de destination connue sont envoyées à travers le mécanisme de diffusion. Une vérification des trames à l'entrée des commutateurs à travers le mécanisme de *Reverse Path Forwarding Check (RPFC)* permet la prévention des boucles. Pour atténuer les problèmes causés par la formation de boucles temporaires les trames possèdent un champ *hop count*. À l'entrée du réseau *TRILL*, le premier *RBridge* (commutateur d'entrée) encapsule la trame Ethernet avec l'en-tête *TRILL* qui possède une indication du dernier *RBridge* (commutateur de sortie). Chaque *RBridge* reçoit un pseudonyme de 2 octets construit à partir du *IS-IS* ID; ce pseudonyme est l'indication utilisé dans l'entête *TRILL* comme indication des *RBridge*s d'entrée et sortie.

Le *SPB*

Le *SPB* est spécifié dans une modification à la norme IEEE 802.1Q et utilise le protocole *IS-IS* avec des extensions (*IS-IS-SPB*) pour la construction des *Shortest Path Trees (SPT)* [15]. Les SPTs construits dans chaque commutateur *SPB* possèdent les chemins les plus courts vers les autres commutateurs du réseau. Ces chemins sont utilisés dans les deux directions. Le *SPB* possède un mécanisme pour le bris d'égalité entre des chemins calculés par l'algorithme, et de cette façon le meilleur chemin calculé par deux commutateurs sera toujours le même. Tel que le *TRILL*, le *SPB* utilise le mécanisme de prévention de boucles par filtrage à l'entrée de chaque commutateur à travers le *RPFC*. Pour établir les chemins le *IS-IS-SPB* peut utiliser aussi les identificateurs de *VLAN* (*VID*s) et les identificateurs de service (*Service identifiers (I-SID)*) utilisés par la norme IEEE 802.1ah. Il existe deux modes d'opération *SPB*, le SPBV utilise le *shortest pathVID* (SPVID) pour identifier les SPT et le SPBM utilise les adresses *MAC* pour l'identification des SPTs.

Dans ce chapitre nous avons présenté les technologies Ethernet traditionnelles et de nouvelle génération. On a vu que la popularité de l'Ethernet a provoqué le développement de nouvelles technologies qui permettent son utilisation dans les réseaux étendus. Pourtant, le concept du « Carrier Ethernet » exige quelques attributs qui ne sont pas encore existants dans les réseaux Ethernet utilisés présentement. Au niveau des protocoles de contrôle, l'utilisation de protocoles traditionnels comme le *RSTP* peut limiter la capacité du réseau, puisqu'ils n'ont pas été conçus pour être utilisés dans ce type de réseau. L'utilisation du protocole

IS-IS pour le contrôle des réseaux Ethernet de nouvelle génération a été adoptée pour les technologies *SPB* et *TRILL*, ce qui permet de résoudre les problèmes décrits plus haut, puisque ce protocole était déjà utilisé avec succès dans les réseaux IP.

Chapitre 3

La robustesse des réseaux

Les réseaux de transport doivent présenter une haute fiabilité, mais pourtant les défaillances du réseau ne sont pas un événement rare. Les problèmes peuvent avoir plusieurs origines telles que des interruptions dans le moyen physique, défaillance d'équipements ou des tâches de manutention. Pour diminuer les problèmes causés par les interruptions il est nécessaire d'implémenter des mécanismes pour augmenter la robustesse du réseau. Bien que ces mécanismes puissent être implémentés dans différentes couches du réseau, un mécanisme au niveau optique ne sera pas capable de détecter une défaillance dans une interface électrique d'un routeur par exemple [38]. Pour cette raison l'utilisation de mécanismes de recouvrement dans une couche supérieure, voire à chaque niveau, présente des avantages. Ce chapitre fera une description des mécanismes de recouvrement utilisés dans les réseaux de transport et Ethernet.

3.1 Les mécanismes de recouvrement

Pour contourner les problèmes causés par les interruptions dans le moyen physique et ainsi garantir la robustesse du réseau, plusieurs mécanismes de recouvrement ont été développés. Une façon de classifier ces mécanismes consiste à distinguer les mécanismes de protection et de rétablissement [39]. Les mécanismes de protection offrent un chemin alternatif qui est établi avant l'occurrence de la défaillance, alors que les mécanismes de rétablissement offrent le chemin alternatif après l'occurrence de la défaillance [40]. La figure 3.1 monte un exemple de cette classification.

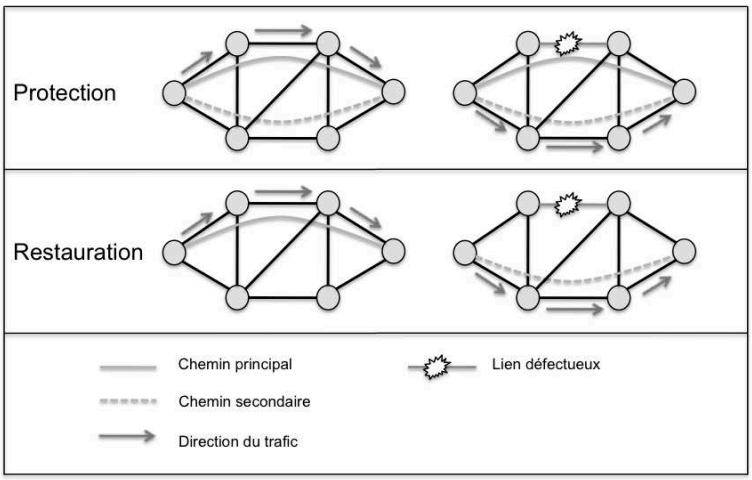


FIGURE 3.1 – Exemple de protection et rétablissement

Une autre classification utilisée pour les mécanismes de recouvrement est la distinction entre recouvrement local et global. Un mécanisme de recouvrement local produit un détour seulement autour de l'élément défectueux, alors que dans un mécanisme de recouvrement global le chemin alternatif n'utilise pas d'éléments communs au chemin principal, sauf à l'origine et à la destination. Les chemins principal et alternatif peuvent aussi utiliser une partie des éléments communs, ce qui serait une solution intermédiaire (mixte) entre les deux solutions [39]. La figure 3.2 montre des exemples.

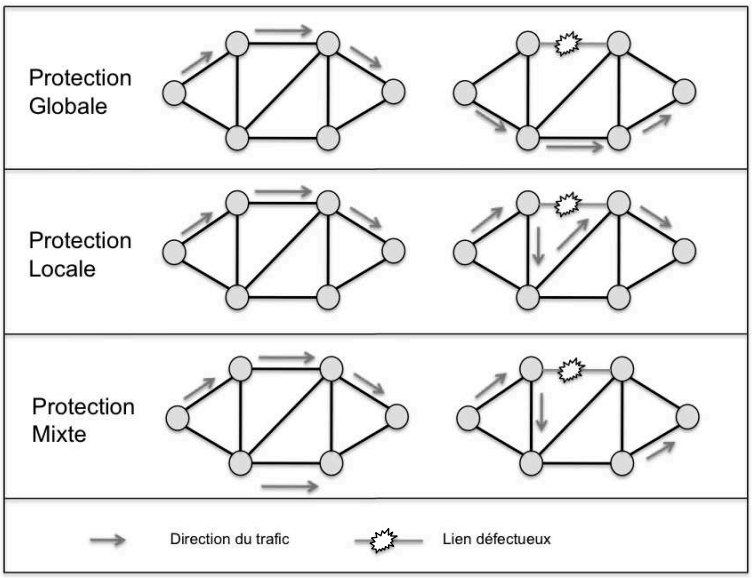


FIGURE 3.2 – Exemple de récuperation globale, locale et mixte

Le mécanisme de contrôle de recouvrement peut utiliser un système centralisé, quand un contrôle central est responsable pour les actions, ou un système décentralisé ou distribué, quand les éléments du réseau

initient la procédure de recouvrement d'une façon autonome. Nous présentons maintenant une introduction aux technologies utilisées dans les réseaux de transport et quelques mécanismes de recouvrement usuels.

3.1.1 Réseaux SONET/SDH

Les réseaux SONET/SDH sont utilisés pour offrir le médium de transport physique pour plusieurs technologies de commutation de paquets comme Ethernet, *ATM*, *IP* et *MPLS*. En principe une technologie *TDM* n'est pas très efficiente pour le transport de données, mais des mécanismes d'adaptation tel que le *GFP*, *LCAS*, *VCAT* et la technologie *OTN* ont été développés pour adapter la technologie à la transmission de paquets. Le mécanisme de recouvrement le plus utilisé dans les réseaux SONET/SDH est un mécanisme de protection qui utilise le protocole *Automatic Protection Switching (APS)*, ce mécanisme peut garantir un recouvrement rapide sur une topologie en anneau (*Ring Protection*) ou sur un lien entre deux équipements (*Linear Protection*). La topologie et la longueur des liens peuvent influencer le temps de recouvrement. Pour illustrer ceci, à partir des exemples montrés en [39] on peut obtenir une équation du temps maximal nécessaire pour que l'*APS* puisse compléter la procédure de recouvrement dans une topologie en anneau:

$$T_{max} = 3 \times (N-1) \times 0.875 \times L \times 10^{-2},$$

où N est le nombre de nœuds et L est la longueur de l'anneau en km; le délai de propagation du médium physique est de 8.75 μs par km. Ce temps maximal correspond à trois fois le délai de transit et n'intègre pas le temps de traitement des messages du protocole. L'ITU-T définit un temps maximal de 50 ms pour une topologie en anneau avec un nombre maximal de 16 nœuds et une longueur maximale de 1200 km [41].

3.1.2 Réseaux IP

Les réseaux *IP* traditionnels utilisent un protocole de routage interne (*Interior Gateway Protocol (IGP)*) pour transporter les informations qui permettront l'établissement des routes. Les protocoles *IGP* les plus utilisés comme le *Open Shortest Path First (OSPF)* et le *IS-IS* présentent un temps de convergence variable. En prenant compte du temps de détection d'un problème, du temps pour recalculer la nouvelle topologie dans le routeur qui détecte la défaillance, du temps de génération et distribution des annonces de changement et

du temps de calcul dans les autres routeurs du réseau, le temps total peut totaliser de l'ordre de quelques centaines de millisecondes ou même plusieurs secondes [42]. On trouve dans [43] une analyse des facteurs qui affectent le temps de convergence de ces protocoles. En utilisant le *IS-IS*, le temps de recouvrement dépend de la synchronisation des tables de chaque commutateur pour garantir la gestion de la topologie logique du réseau. En cas d'altération de la topologie physique c'est le temps nécessaire pour synchroniser les tables qui va déterminer le temps de convergence. La détection d'un problème dans un lien physique peut être faite par la couche physique (par exemple à travers les mécanismes du SONET/SDH), par l'utilisation d'un protocole de gestion (*keep alive checking*), ou par l'expiration du temps d'attente des messages *IIH*. Une fois que le niveau de contrôle du nœud est informé du changement d'état, il produit un nouveau *LSPDU* pour diffuser l'information aux autres nœuds du réseau, et ainsi chaque nœud peut mettre à jour sa base de données avec la nouvelle information et calculer la nouvelle topologie.

La convergence du protocole *IS-IS* a été étudiée dans plusieurs travaux dans le cadre des réseaux IP. Dans [43] [44] les facteurs qui influencent le temps de recouvrement dans un routeur sont identifiés, et cette équation montre la caractérisation du temps de convergence (Tc) du protocole *IS-IS* (dans ce cas le temps de recouvrement du réseau):

$$T_c = D + O + F + SPT + RIB + DD \tag{3.1}$$

où *D* est le temps de détection de la défaillance qui selon l'étude présentée est accompli en moins de 20ms dans la majorité des cas; *O* est le temps pour générer le *LSPDU* (moins de 12ms selon l'étude); *F* est le temps de diffusion à partir du nœud qui détecte la défaillance jusqu'aux nœuds qui doivent réaliser le changement dans leur base de données (ce temps est variable selon la topologie et l'utilisation de mécanismes comme le *Fast Flooding*); *SPT* est le temps pour le calcul de l'arbre, l'algorithme *Shortest Path First (SPF)* typique présente une complexité O($n$log(n)) qui peut diminuer à travers l'utilisation d'optimisations comme le *Incremental SPF (iSPF)* - l'étude a obtenu une valeur linéaire approximative de 45μs par nœud sans l'utilisation d'iSPF; *RIB* est le temps nécessaire pour remplir les tables, qui augmente avec le nombre de préfixes (IP) modifiés et dans l'étude il se trouve dans l'ordre des centaines de microsecondes par préfixe; *DD* est le temps de distribution des tables aux cartes de ligne, un temps moyen de moins de 50 ms dans l'étude. Dans l'article l'analyse du temps de convergence a été faite à travers la simulation de deux topologies réelles différentes (GÉANT et un *Internet Service Provider (ISP)* Tier-1) et le temps de convergence dans le cas de défaillance dans un lien pour la topologie *ISP* a présenté des valeurs entre 50 ms et 250 ms pour les diverses

configurations utilisées. Par contre, la relation entre temps de convergence et croissance du réseau n'a pas été évaluée.

3.1.3 Réseaux *MPLS*

La caractéristique « orienté connexion » du *MPLS* permet l'utilisation de mécanismes de protection et rétablissement similaires à ceux qui sont utilisés par les réseaux SONET/SDH. Les mécanismes de recouvrement *MPLS* sont basés sur l'établissement de chemins alternatifs à travers la construction de *Label Switched Path (LSP)*s. Un mécanisme de protection globale pour le *MPLS* utilise un *LSP* protégé par un autre *LSP* disjoint préétabli qui fonctionne comme *backup*. Un mécanisme de rétablissement global est réalisé à travers l'établissement d'un *LSP* alternatif après l'occurrence de la défaillance. La détection de la défaillance est faite à travers un protocole de gestion qui surveille le chemin et les routeurs à la bordure sont responsables pour le changement [45]. Les mécanismes *Multiprotocol Label Switching - Fast Reroute (MPLS-FRR)* offrent un recouvrement rapide à travers la protection locale dans un réseau *MPLS*. Dans [46] deux méthodes pour le *MPLS-FRR* sont définies, le *one-to-one backup* pour la protection de *LSP*s individuels et le *facility backup* pour la protection de plusieurs *LSP*s à travers le mécanisme du *label stacking*. La RFC définit des extensions au protocole RSVP-TE pour l'établissement des *LSP*s tunnels. Le *MPLS-FRR* peut offrir un recouvrement dans un temps équivalent aux mécanismes utilisés pour les réseaux SONET/SDH (50 ms) [47].

3.1.4 Réseaux Ethernet traditionnels

Dans un réseau qui utilise le *STP* comme protocole de contrôle, pendant un changement de topologie il est parfois nécessaire de faire des changements dans les tables d'adresses des commutateurs pour que les trames ne soient pas envoyées vers des chemins qui ne sont plus valides et donc empêcher l'existence d'informations ambigües. Un commutateur qui détecte un changement doit informer le réseau à travers la transmission des *TCN BPDU*s; cela se produit à chaque *Hello Time* et les messages sont envoyés à travers son *Root port* vers le commutateur racine. Chaque commutateur qui reçoit le *TCN BPDU* répond avec un message *TCN Acknowledgement*. Une fois que le message arrive au commutateur racine, il commence à transmettre ses *BPDU*s avec une indication de changement de topologie (*TCN flag*) pendant une certaine période de temps (*Max Age* + *Forward Delay*). Les commutateurs qui reçoivent ces *BPDU*s doivent diminuer le temps d'expiration des tables (*Ageing Time*), qui est réduit à la valeur du *Forward Delay* (15s par défaut).

Comme vu précédemment dans le chapitre 2, le *RSTP* implémente des améliorations à la procédure utilisée par le *STP*. Le mécanisme pour le changement des tables d'adresses après un changement de topologie a aussi été modifié. Un commutateur qui détecte un changement envoie un message *TCN* (RST *BPDU* avec un *flag* TC). Les commutateurs qui reçoivent le message *TCN* doivent décarter les informations des tables d'adresses dans tous les autres ports et renvoyer le message aux autres commutateurs.

Le *RSTP* résout le problème du temps de convergence trop élevé du *STP*, mais ce temps de convergence en cas de problème est variable et dépend de la topologie. Avec l'utilisation d'une topologie en anneau le temps de convergence peut devenir plus prévisible, mais la perte de communication avec le commutateur racine peut toujours mener à un temps de convergence élevé et à l'occurrence de boucles. Les articles [48] et [34] font l'analyse du problème *« count-to-infinity »* causé par l'utilisation du *RSTP* dans un réseau en anneau.

Le *RSTP* est le protocole utilisé dans les réseaux Ethernet traditionnels et plusieures études sur le temps de convergence de ce protocole ont déjà été publiées; l'article [49] montre une analyse du temps de recouvrement en utilisant une topologie en anneau. Dans ce travail est établi une équation du temps de convergence idéal (minimal) prenant en compte le nombre de commutateurs dans l'anneau, le temps de traitement des *BPDU*s et le délai des liens. L'analyse utilise des simulations et des améliorations au protocole sont proposées pour diminuer le temps de convergence et l'approcher du temps idéal, mais cette analyse se limite aux topologies en anneau. Dans [50] on trouve une comparaison entre le temps de convergence mesuré dans un réseau réel qui utilise des commutateurs de plusieurs fournisseurs et des réseaux simulés. Le travail montre que le temps de convergence du *RSTP* présente une variabilité qui dépend de l'implémentation des algorithmes dans les commutateurs et utilise des améliorations pour approcher les résultats des simulations à ceux obtenus dans le réseau réel. Les auteurs définissent les mesures de temps liées à la performance et montrent que le temps de traitement des *BPDU*s est la partie la plus importante de la mesure de performance.

Le blocage des ports pour éviter les boucles représente aussi un problème d'ingénierie du trafic, vu qu'une partie de la capacité du réseau reste inutilisée. L'utilisation du *MSTP* peut apporter un avantage en termes d'ingénierie du trafic par rapport au *RSTP*, étant donné que chaque *MSTI* configure sa nouvelle topologie active indépendamment. Par contre le mécanisme de protection fonctionne de la même façon que le protocole *RSTP*, donc le protocole n'offre pas d'amélioration en termes de vitesse de recouvrement d'un arbre spécifique en cas de défaillance et il est donc sujet aux mêmes problèmes vérifiés dans les réseaux qui utilisent le *RSTP*. La gestion du réseau et la mise en échelle est un autre problème, parce que la correspon-

dance des VPNs avec les *MSTI*s est faite de façon manuelle par l'administrateur [51]. Le nombre de *MSTI*s dans une région est aussi limité (maximum de 64).

3.1.5 Réseaux Ethernet à état de liens

L'utilisation d'un protocole de routage à état de liens dans le plan de contrôle est utilisé pour résoudre le problème d'ingénierie du trafic des réseaux Ethernet traditionnels qui utilisent le *RSTP*. Pourtant, comme dans les réseaux *IP* traditionnels, le temps de recouvrement en cas de défaillance dans ces réseaux dépend du temps de convergence du protocole de routage. L'article [35] présente une analyse de performance des protocoles de contrôle proposés pour le *SPB*.

Dans l'étude trois topologies de base sont utilisées pour mesurer le temps de convergence des protocoles de contrôle définis comme *Distance-vector SPB* qui utilise le *MSTP* et le *Link-state SPB* qui utilise le *IS-IS*. Deux versions de ce dernier sont utilisées: la version de base, plus similaire au protocole original et le *Root Controlled Bridging (RCB)* avec des améliorations spécifiques pour le *SPB*.

Le temps de convergence mesuré dans l'étude a varié entre 5 ms et 50 ms pour des réseaux de trois topologies différentes avec un nombre de nœuds entre 50 et 280. Pourtant les topologies en anneau, *Heavy Mesh* et *Light Mesh* analysées ne sont pas parmi les topologies les plus utilisées dans les réseaux réels et le délai des liens n'a pas été pris en compte.

Les réseaux Ethernet qui utilisent un protocole à état de liens dans le plan de contrôle sont aussi des réseaux non-orientés connexion. Les mécanismes de recouvrement rapide développés pour les réseaux *IP* peuvent apporter une solution, mais les caractéristiques de transmission par diffusion et l'utilisation de mécanismes de filtrage à l'entrée du type *RPFC* sont des éléments qui doivent être traités pour le développement d'un mécanisme de recouvrement rapide spécifique. Dans la littérature il n'existe pas beaucoup de publications sur les mécanismes de recouvrement rapide spécifiques pour ce genre de réseau. Les technologies *PLSB* et *SPB* peuvent utiliser le mécanisme *Equal Cost Multipath (ECMP)* en utilisant des arbres couvrants minimaux de même coût, chaque arbre utilisant un *Backbone VID (B-VID)* différent. Pourtant, comme pour les réseaux *IP*, ce mécanisme repose sur l'existence de chemins alternatifs de même coût; de plus, l'établissement d'un nouveau *B-VID* est fait à l'origine et non localement. Au delà de ce mécanisme, le recouvrement est basé sur la convergence du protocole *IS-IS*, et les améliorations du matériel et de l'implantation de modifications des paramètres du protocole sont des solutions qui peuvent diminuer le temps de convergence.

Pourtant ces solutions sont limitées et ne peuvent pas garantir un temps de convergence minimal à cause de l'impact de la topologie du réseau. L'article [35] pose une analyse sur la convergence du protocole *IS-IS* avec les améliorations pour le *SPB*, et dans cette étude on peut voir les variations du temps de convergence pour différentes topologies. La figure 3.3 montre le résultat des mesures du temps de convergence des protocoles *MSTP*, *IS-IS* et *IS-IS* modifié pour le *SPB*. Dans le graphique on peut observer l'augmentation du temps de convergence avec l'agrandissement du réseau. Bien que pour un nombre maximal de 280 nœuds le temps de convergence puisse être de moins de 50 ms la topologie analysée ne représente pas le cas d'une topologie maillée aléatoire.

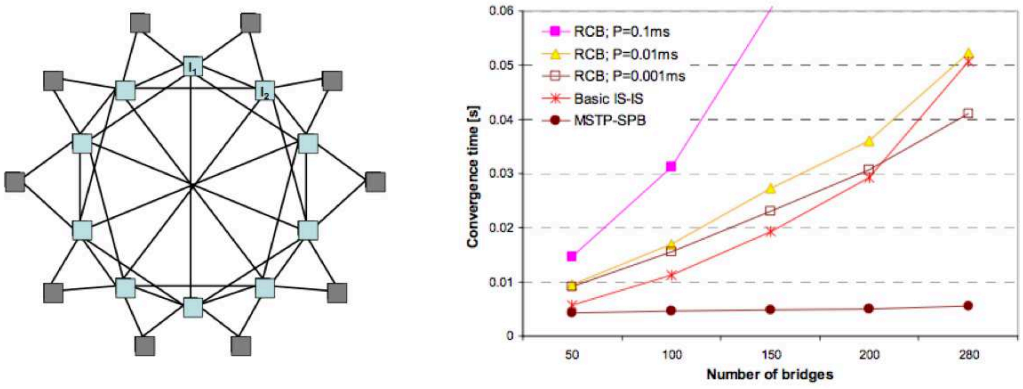


FIGURE 3.3 – Temps de convergence d'une topologie *light-mesh* [42]

Dans [52] on présente une étude sur le temps de convergence du protocole *IS-IS* avec les mécanismes de prévention de boucles du *SPB* implémentés; dans cette étude on utilise des topologies réelles et des topologies maillées générées aléatoirement. La figure 3.4 montre les résultats de cette étude pour les défaillances des liens du réseau. Dans le tableau on peut voir que pour une topologie aléatoire de 100 et 150 nœuds ce temps peut arriver à des centaines de millisecondes.

3.2 Simulation I - Temps de convergence

Dans cette section nous montrons des simulations qui ont été réalisées pour l'analyse du temps de convergence des réseaux. Nous décrivons les simulations faites pour analyser le temps de convergence du protocole *IS-IS* à travers une comparaison avec le *RSTP*. Ces simulations ont comme objectif de montrer les avantages de l'utilisation d'un protocole à état de liens pour le contrôle d'un réseau Ethernet.

	AT&T	Germany50	Cost266	R100	R150
Default IS-IS					
min	2667	4941	5305	3898	4921
avg	**3641**	**5472**	**5767**	**4593**	**5499**
max	4281	5899	6050	5321	6092
Fine tuned IS-IS, no loop prevenction					
min	7.38	34.66	17.98	162.69	394.08
avg	**7.98**	**35.43**	**18.63**	**163.65**	**394.81**
max	8.39	35.67	19.10	164.09	395.74
Fine tuned IS-IS with neighbour synchronization					
min	8.38	35.50	18.99	164.09	395.09
avg	**9.01**	**36.02**	**19.62**	**164.73**	**395.76**
max	9.44	36.67	20.11	165.10	396.75

FIGURE 3.4 – Temps de convergence (en ms) du protocole IS-IS [52]

3.2.1 Méthode d'analyse

Pour mesurer le temps de convergence du protocole *RSTP*, le simulateur BridgeSim [53] avec les modifications suggérées par [49] a été utilisé; ces modifications permettent d'augmenter le nombre maximal de commutateurs dans une topologie en anneau à travers l'altération du mécanisme qui contrôle l'âge maximal des *BPDU*s (*Max Age*) et le nombre maximal de *BPDU*s transmises par un port dans un certain intervalle de temps (*TxHoldCount*). Dans la configuration normale du protocole, ces mécanismes posent des limites qui diminuent sa performance. Le simulateur est utilisé pour construire la topologie et simuler des défaillances dans les liens. À la fin de chaque simulation il génère une archive avec l'enregistrement de tous les messages et altérations d'état des commutateurs du réseau. Dans les mesures, le temps de convergence est considéré comme le temps entre l'occurrence d'une défaillance et le dernier message qui provoque un changement d'état des ports du commutateur, mais le temps de détection de la défaillance n'est pas pris en considération. La simulation du protocole *IS-IS* utilise le simulateur *OMNeT++* avec l'implémentation du protocole utilisé par [54] et modifiée pour l'adaptation aux changements de topologie provoqués par les défaillances des liens. Le protocole simulé implémente aussi des altérations au fonctionnement du protocole original pour augmenter sa performance: il n'y a pas de limite au nombre de *LSPDU*s transmises et après l'occurrence de la défaillance du lien le *LSPDU* généré est immédiatement envoyé. Le temps de convergence mesuré est le temps entre l'occurrence de la défaillance et la réception du dernier message qui provoque un nouveau calcul de l'arbre. Tel que montré par [50], dans un réseau réel le temps de traitement des *BPDU*s est variable; dans nos simulations le même principe a été utilisé pour le temps de traitement des *LSPDU*s du protocole *IS-IS*. Pour les analyses qui suivent, une distribution normale avec moyenne de 3 ms et écart type de 0,5 a été utilisée pour la simulation du temps de traitement des *BPDU*s et des *LSPDU*s; cette valeur est la même

que celle qui est utilisée dans les simulations de [35]. Pour les topologies en anneau et en grille on a utilisé une valeur de 1 ms pour le délai des liens, et pour la topologie *Boston university Representative Internet Topology gEnerator (BRITE)* un délai avec une distribution uniforme entre 1 ms et 10 ms.

3.2.2 Tests réalisés

La simulation utilise trois topologies physiques pour l'analyse de performance: la topologie en anneau, la topologie en grille et une topologie maillée (Waxman) créée à travers le générateur de topologies *BRITE* [55]. Pour chaque topologie, le nombre de nœuds (n) est variable et l'occurrence de défaillances dans chacun des liens entre les commutateurs est simulée. La figure 3.5 montre un diagramme de ces topologies.

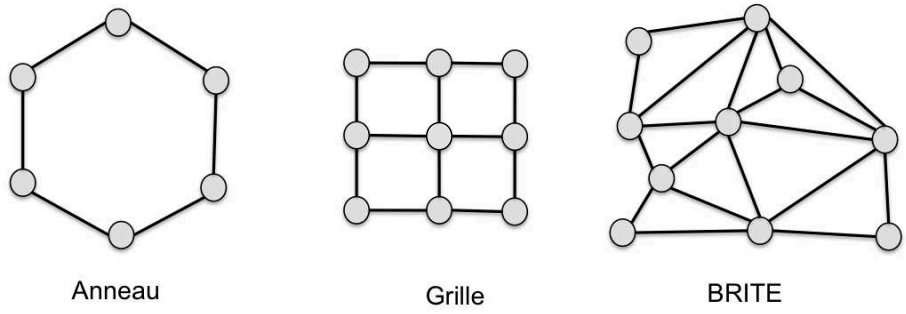


FIGURE 3.5 – Les différentes topologies utilisées

Pour la topologie en anneau on a utilisé un nombre de nœuds n=5,10,15,20,25; pour la topologie en grille, n=9, 16,25; enfin, pour la topologie *BRITE*, n=10,20,30. Les simulations ont été répétées 20 fois pour chaque topologie.

Topologie en anneau

L'anneau est la topologie physique la plus simple utilisée dans cette analyse. La topologie a été utilisée pour étudier le comportement des protocoles et valider les mesures. Dans cette topologie, avec les paramètres utilisés, le temps de recouvrement augmente de façon linéaire selon le nombre de nœuds, le délai des liens et le temps de traitement des messages. La figure 3.6(a) montre un diagramme de la topologie et de l'échange des messages dans le cas du *RSTP* en considérant le nœud A comme la racine après la défaillance du lien entre les nœuds A et B. La séquence d'événements se passe de la façon suivante:

1. En détectant la défaillance, B se considère lui-même comme racine, puisqu'il n'a pas de chemin alternatif à travers un autre port. Il envoie donc un *BPDU* vers C pour annoncer sa décision.
2. C reçoit le *BPDU* envoyé par B et après le traitement du message il l'accepte comme nouvelle racine, et le *BPDU* est envoyé vers D.
3. Le *BPDU* avec la nouvelle information est reçu par D, mais ce nœud possède un chemin vers A à travers son port bloqué; il va donc proposer le changement de topologie en modifiant l'état du port bloqué qui deviendra un port racine. Cette décision est annoncée vers C à travers un nouveau *BPDU*.
4. C reçoit le message envoyé par D et retransmet l'information sur le nouveau chemin vers B, ce message est reçu par B qui accepte A comme racine à travers le nouveau lien. Un message pour confirmer sa décision est envoyé vers C.
5. Le message de confirmation est reçu par C, donc la nouvelle topologie peut devenir active.

Dans [49] on montre une équation pour calculer le temps de recouvrement idéal après une défaillance pour le *RSTP* dans une topologie en anneau:

$$T_c = \sum_{l=1}^{n} d_l \tag{3.2}$$

où T_c est le temps de recouvrement idéal (temps de convergence maximal idéal), n est le nombre de nœuds dans l'anneau et d_l est le délai du lien plus le temps de traitement du message dans le nœud adjacent au lien.

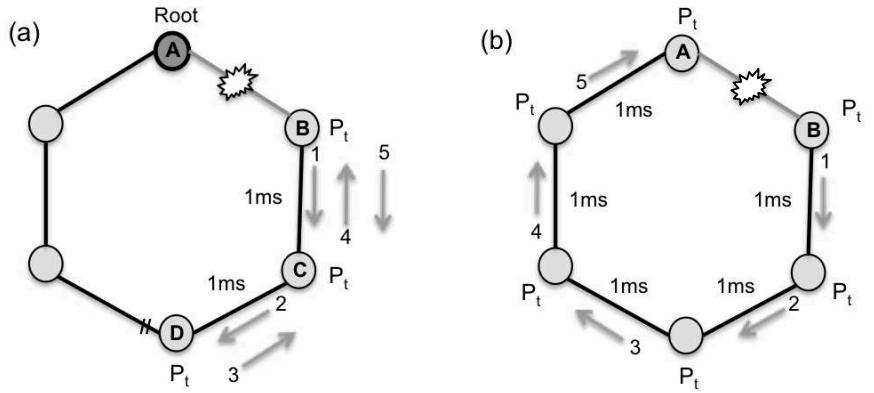


FIGURE 3.6 – La topologie en anneau

Dans un réseau contrôlé par le protocole *IS-IS* il n'existe pas de changement d'état de ports, la topologie logique étant définie par les informations présentées dans les tables de routage. Le changement de topologie

se passe à travers l'actualisation des tables après le calcul de la topologie en utilisant les informations reçues par les messages. Dans notre topologie en anneau, en détectant la défaillance les nœuds A et B vont générer un nouveau *LSPDU* et les diffuser à travers le réseau. La figure 3.6 (b) montre un diagramme avec le message produit par le nœud B, qui suit la séquence d'événements suivante:

1. La défaillance est détectée par B qui va générer un *LSPDU* pour informer les autres nœuds de la nouvelle topologie; le *LSPDU* est ensuite reçu par chacun des nœuds subséquents de l'anneau qui calculeront la nouvelle topologie.
2. Après avoir fait le tour de l'anneau le *LSPDU* envoyé par B arrive à A.

Le message produit par A suivra la même séquence dans le sens contraire, jusqu'à sa réception par le nœud B. Dans le cas du *IS-IS* le temps de convergence peut être calculé en utilisant l'équation 3.2.

Topologie en grille

La topologie en grille est la deuxième analysée dans nos simulations. Elle présente une complexité plus grande que la topologie en anneau puisque les nœuds ont de 2 à 4 voisins. La figure 3.7 (a) montre un diagramme de la topologie et les échanges des messages pour le *RSTP*. Dans la figure, le nœud A est considéré comme racine.

1. En détectant la défaillance, B se considère lui même comme racine. Il envoie donc une *BPDU* vers C et D pour annoncer sa décision.
2. C et D reçoivent la *BPDU* envoyé par B. Après le traitement du message, C accepte B comme le nouveau commutateur racine (puisqu'il utilisait B comme chemin vers A), la *BPDU* avec cette nouvelle information est envoyé vers E. Pourtant, D a un chemin vers A , il envoie donc une *BPDU* vers B en informant que A est le vrai commutateur racine.
3. La *BPDU* avec la nouvelle information est reçu par E qui possède aussi un chemin vers A (à travers D), il rejette B comme racine et envoie donc une *BPDU* vers C l'informant que A est la racine. En même temps B reçoit la *BPDU* envoyé par D et accepte le nouveau chemin vers A, il envoie une *BPDU* vers C avec la nouvelle information.
4. C reçoit le message envoyé par B et D informant A comme racine; pour C les deux chemins sont équivalents et dans cet exemple il accepte le chemin proposé par B. Il confirme donc le message reçu par B et rejette le message reçu par D.

Cet exemple montre une des possibilités, mais comme il existe d'autres arbres possibles à partir du nœud A l'échange des messages entre les nœuds peut varier. La façon dont les nœuds sont distribués dans le réseau peut aussi modifier le comportement pendant un changement de topologie, une fois que l'élection d'un nouveau commutateur racine dépend aussi du *Bridge identifier*.

La figure 3.7 (b) montre un diagramme de la topologie et les échanges des messages pour le *IS-IS*. Dans la figure, seuls les messages envoyés à partir de A sont montrés, mais il existe également un flot de messages à partir de B.

1. La défaillance est détectée par A qui va générer un *LSPDU* pour informer les autres nœuds du changement de topologie, le *LSPDU* est reçu par chacun des nœuds de l'anneau subséquents qui calculeront la nouvelle topologie.
2. Après avoir parcouru le réseau, le nœud C est le dernier à recevoir le *LSPDU* produit par A, puisqu'il est le plus éloigné.

Dans le cas du protocole *IS-IS*, on observe que le diamètre du réseau a un rôle important pour déterminer le temps de convergence maximal. En considérant le délai des liens et le temps de traitement des *LSPDU*s, on peut établir une équation pour ce temps maximal:

$$T_c = D \times d_l \tag{3.3}$$

où T_c est le temps de recouvrement idéal (temps de convergence maximal idéal), D est le diamètre du réseau et d_l est le délai du lien plus le temps de traitement du message dans le nœud adjacent au lien.

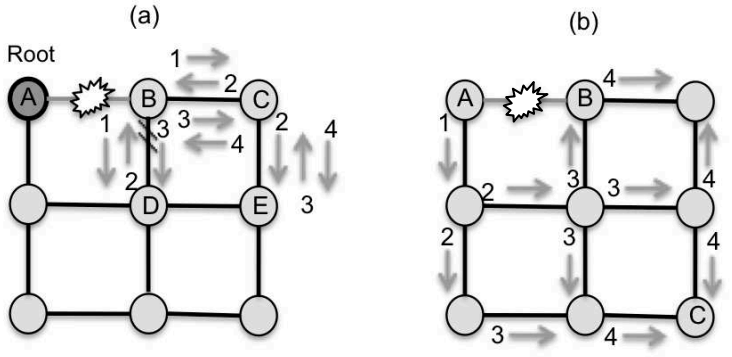


FIGURE 3.7 – La topologie en grille

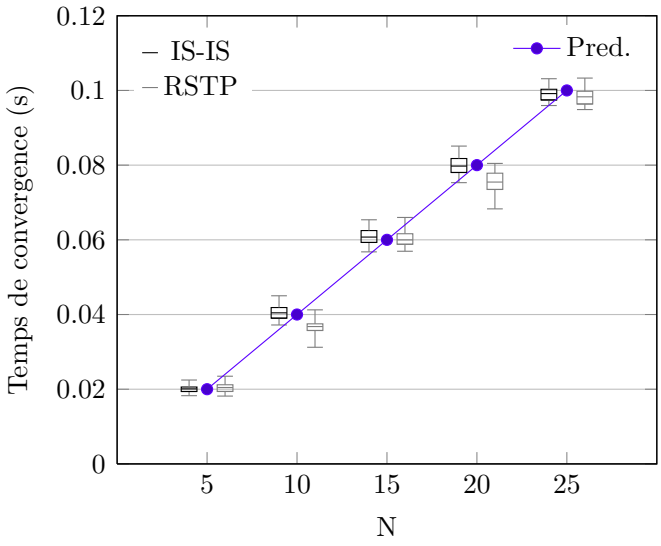


FIGURE 3.8 – Temps de convergence de la topologie en anneau

Topologie maillée *BRITE*

Les topologies montrées précédemment ont été utilisées dans plusieurs études sur le temps de convergence des protocoles de contrôle; pourtant ces topologies ne représentent pas le cas des réseaux de transport réels. Les anneaux sont utilisés principalement dans les réseaux métropolitains, mais une topologie maillée présente une flexibilité plus élevée. La topologie maillée obtenue par le générateur *BRITE* possède un grand nombre de connexions entre les nœuds, mais n'est pas une topologie symétrique comme celles étudiées précédemment. Ainsi que pour la topologie en grille, dans le cas du *RSTP* la modification du commutateur utilisé comme racine provoque une variation du temps de convergence calculé pour chaque lien et dans le cas du *IS-IS*, c'est le diamètre du réseau qui va déterminer le temps de convergence maximal.

3.2.3 Résultats

La figure 3.8 montre les résultats pour le temps de convergence maximal pour la topologie en anneau. On peut observer que le temps de convergence de la topologie simulée suit les résultats obtenus par l'équation 3.2 qui donne le temps de convergence idéal.

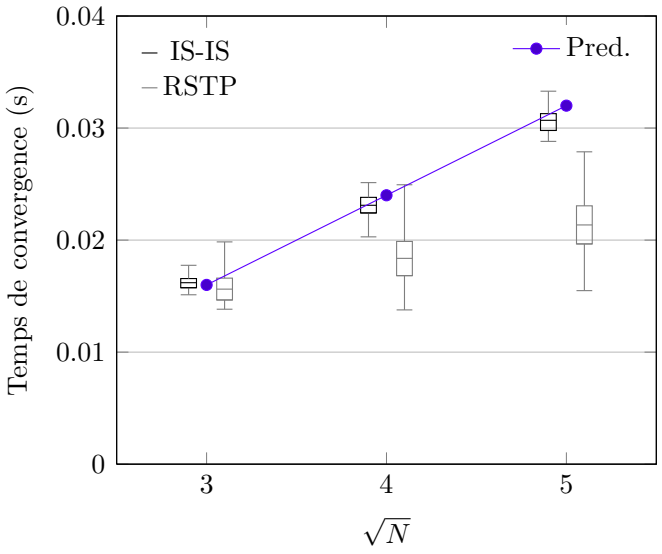


FIGURE 3.9 – Temps de convergence maximal pour la topologie en grille

Les résultats obtenus pour le *IS-IS* montrent que, avec les paramètres utilisés dans la simulation, les résultats obtenus pour les temps de convergence de ce protocole sont équivalent à ceux obtenus pour le *RSTP*, ce qui confirme l'analyse faite dans la section précédente.

La figure 3.9 montre les résultats pour le temps de convergence maximal pour la topologie en grille.

Pour cette topologie on observe que les résultats obtenus pour le *IS-IS* sont plus élevés, une fois qu'ils se rapprochent du temps de convergence maximal obtenu à travers l'équation 3.3

La figure 3.10 montre les résultats pour le temps de convergence moyen parmi tous les liens dans la topologie maillée.

Pour cette topologie, on observe que les résultats obtenus pour le temps de convergence moyen parmi tous les liens du *RSTP* sont plus élevés que dans le cas du *IS-IS*. On peut observer aussi que l'écart entre les valeurs mesurées pour un certain nombre de nœds est plus prononcé pour le RSTP, le temps de convergence de ce protocole peut atteindre la valeur de 50 ms même dans le cas d'un réseau avec 10 nœuds. Pour le *IS-IS*, même si la plupart des valeurs se trouvent en dessous de 50 ms, on ne peut pas garantir cette valeur maximale avec la croissance du réseau.

La figure 3.11 montre le nombre moyen de paquets de contrôle (*RSTP*/*BPDU*s et *IS-IS*/*LSPDU*s) traités par nœud. On observe que le nombre est presque constant dans le cas du *IS-IS*, mais que, dans le cas du *RSTP*,

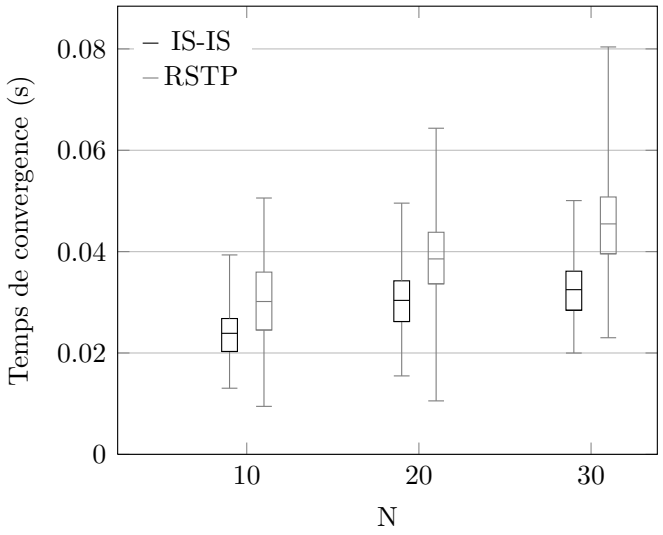


FIGURE 3.10 – Temps de convergence moyen pour la topologie maillée

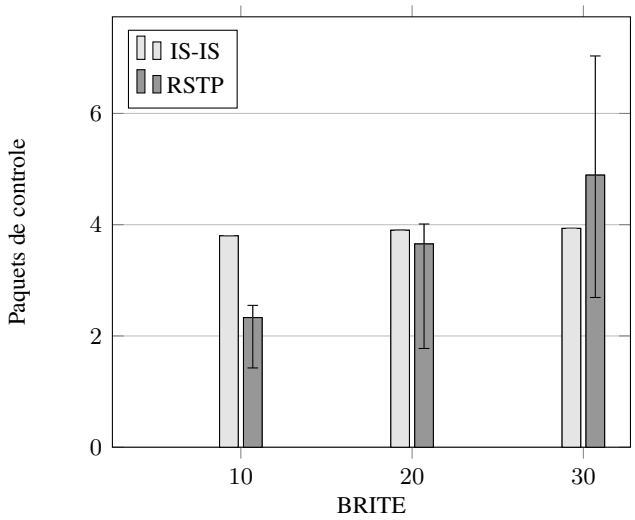


FIGURE 3.11 – Nombre moyen de paquets de contrôle par nœud

il augmente selon le nombre de nœuds du réseau. Pour un même réseau, dans les différentes simulations, le nombre reste constant pour le *IS-IS* alors qu'il est variable pour le *RSTP*. Cela montre que le RSTP est un protocole moins efficace, et qui peut présenter des problèmes pour la mise à l'échelle. Dans la figure 3.12 on peut observer le nombre total de paquets de contrôle dans le réseau et le nombre maximal de paquets de contrôle traités par un seul nœud; dans les deux cas les valeurs observées pour le *RSTP* sont beaucoup plus élevées que pour le *IS-IS*. Cela renforce notre conclusion sur les problèmes de mise à l'échelle du RSTP, vu que l'augmentation du nombre de paquets traités par un nœud peut affecter sa performance.

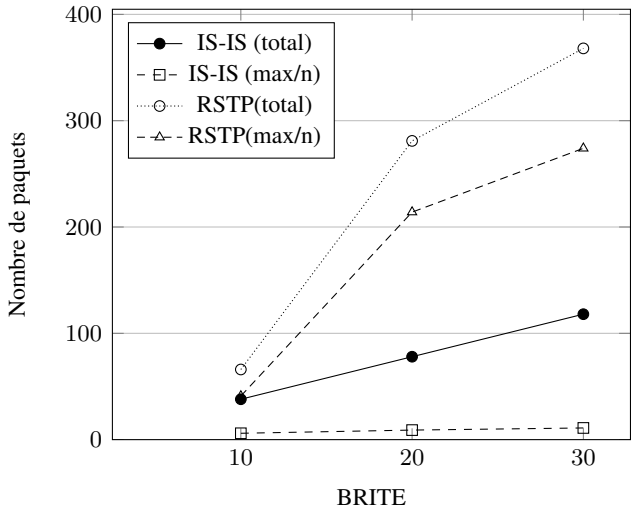


FIGURE 3.12 – Nombre de paquets de contrôle total dans le réseau et maximal dans un seul nœud

Dans ce chapitre on a fait un survol des mécanismes de recouvrement utilisés dans les réseaux de transport. On a vu que les problèmes causés par les interruptions du moyen physique justifient l'utilisation de stratégies pour augmenter la robustesse du réseaux. Il existe différentes façons de classer les mécanismes utilisés, par exemple le recouvrement local et global identifient l'extension des éléments du réseau qui sont affectés par le mécanisme. La performance des réseaux de transport *TDM* traditionnels est la base pour le développement des mécanismes utilisés dans les réseaux de nouvelle génération, et le temps de recouvrement maximal de 50 ms est devenu un standard pour les réseaux de classe opérateur. Dans nos simulations, on a pu observer que ce temps de recouvrement maximal peut dépasser les temps standards si le réseau possède un grand nombre de nœuds. Ainsi, le mécanisme de recouvrement global offert par les protocoles de contrôle *RSTP* et *IS-IS* ne peut pas garantir un temps de recouvrement maximal dans les réseaux de transport communs, vu que le temps de convergence de ces protocoles dépend de l'architecture du réseau. Cela justifie l'utilisation d'un mécanisme de protection local pour les réseaux contrôlés par un protocole à état de liens. Les mécanismes de recouvrement rapide sont le sujet du prochain chapitre.

Chapitre 4

Mécanismes de Recouvrement Rapide

Historiquement le temps de recouvrement de 50 ms est utilisé comme standard pour les réseaux de transport [56] et normalement les mécanismes utilisés dans le niveau physique sont capables d'offrir le recouvrement rapide, quoique dans les couches supérieures le rétablissement offert par les protocoles de contrôle est plus lente [40]. Avec l'utilisation des réseaux *IP* pour le transport d'applications multimédia sensibles à des interruptions du trafic, la recherche d'une solution pour le problème du temps de convergence des protocoles de contrôle est devenu un sujet important. Ce chapitre contient une description des mécanismes de recouvrement rapide utilisés dans les réseaux contrôlés par un protocole à état de liens. Le chapitre s'adresse aussi au problème de la formation de boucles durant le recouvrement du réseau et à la planification des réseaux robustes.

4.1 Mécanismes de recouvrement rapide

Plusieurs mécanismes de recouvrement rapide pour les réseaux *IP-FRR* ont été développés et le IETF a publié quelques propositions [57]. Ces mécanismes de protection locale offrent des chemins alternatifs aux chemins originels, et fonctionnent pendant le temps de convergence du protocole *IGP*. Nous présenterons maintenant un survol de ces mécanismes.

— *ECMP* - Le *ECMP* est une technique de routage qui consiste à utiliser les chemins de même coût pour la transmission sur plusieurs liens. Les divers chemins peuvent donc être utilisés comme alternative

en cas de défaillance. La figure 3.2a montre un exemple où il existe deux chemins de même coût entre les routeurs A et B.

— *Loop Free Alternates* (LFA) - Le LFA est un mécanisme de recouvrement décrit dans [58]; ce mécanisme consiste à utiliser un prochain saut (*next hop*) alternatif en cas de défaillance; pour éviter que le trafic retourne au même routeur qui a fait le détour, le *next hop* doit avoir certaines caractéristiques. Dans la figure 3.2b le nœud qui détecte la défaillance sur le lien détourne le trafic en direction de B vers son LFA.

— *U-Turn Alternates* [59] - Le principe est l'utilisation d'un *next hop* par un routeur qui ne possède pas un LFA mais possède un voisin qui l'utilise comme route principale, ce voisin à son tour possède un LFA. En cas de défaillance le trafic est réorienté vers ce voisin qui est nommé *« U-Turn »*. Dans la figure 3.2c le nœud qui détecte la défaillance détourne le trafic vers son nœud U-Turn. Le nœud U-Turn peut détecter qu'il ne doit pas utiliser son interface usuelle vers B, puisque que le trafic a été reçu par cette même interface (ou à travers une indication explicite dans le paquet faite par le nœud qui a réorienté le trafic).

— Tunnels - Le principe de ce mécanisme est l'utilisation d'un tunnel qui a comme destination l'autre coté de l'élément qui présente la défaillance; ces tunnels sont construits en utilisant des techniques de « tunnellisation » *IP* (IP-in-IP, GRE et L2TP par exemple). Dans la figure 3.2d après une défaillance le trafic vers B est envoyé à travers le tunnel, en arrivant au nœud « fin du tunnel » le trafic est donc envoyé par sa route normale à partir de ce nœud.

— Not-via Addresses - Ce mécanisme a été conçu pour surpasser les limitations des mécanismes décrits précédemment. Il utilise une adresse additionnelle (l'adresse *« Not-via »*) pour chaque interface protégée; ainsi, en cas de défaillance les paquets sont encapsulés en utilisant cette adresse spéciale qui va les envoyer vers un autre élément du réseau à travers un nouveau chemin qui n'utilise pas l'élément affecté par la défaillance. Dans la figure 3.2e le trafic est envoyé par le nœud qui détecte la défaillance vers C à travers l'adresse de l'interface « Not-via D », en arrivant au nœud C il va renvoyer ce trafic vers la destination B en utilisant une autre interface différente de celle qui utilise D comme chemin.

Une analyse de ces mécanismes de routage rapide (à l'exception du Not-via) est faite dans [42]. L'article vérifie le nombre de cas où ces mécanismes sont efficaces avec la croissance du réseau. Le résultat de cette étude a montré que la méthode qui utilise les tunnels est la seule capable de couvrir 100% des cas de défaillance sur les liens. Par contre dans l'article une analyse du temps de recouvrement n'a pas été réalisée.

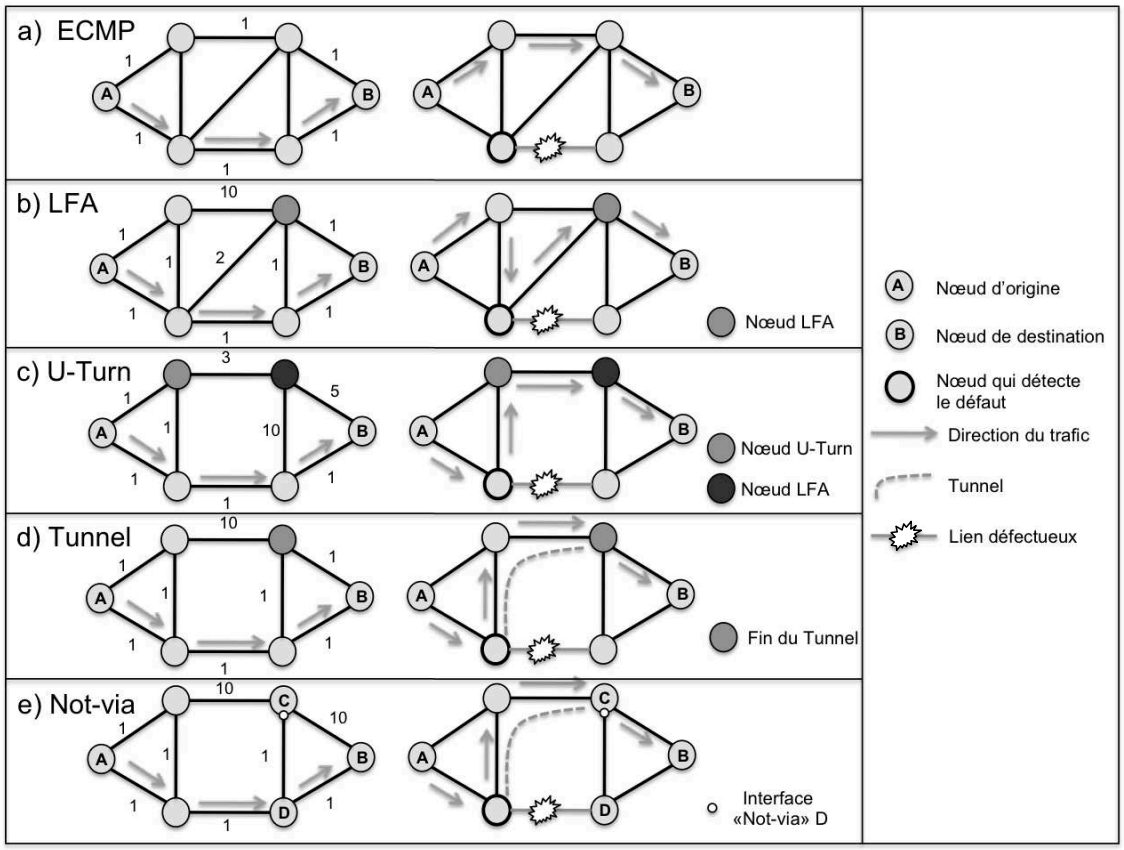


FIGURE 4.1 – Exemple des mécanismes IP-FRR

4.1.1 Le mécanisme de protection *p-Cycle*

Le mécanisme des *p-Cycles* a été décrit dans [60]; il a comme objectif d'améliorer la robustesse des réseaux maillés en offrant un temps de recouvrement similaire à celui des réseaux en anneau. À travers le calcul de chemins alternatifs avant l'occurrence du problème on peut éviter le temps de calcul nécessaire dans un mécanisme global de recouvrement, en vue que seulement les nœuds qui détectent le problème soient responsables pour le détour du trafic (recouvrement local). À l'origine les *p-Cycles* ont été créés pour une utilisation dans les réseaux optiques, mais le concept a été aussi utilisé pour d'autres technologies [61] [62]. La figure 4.2 montre un exemple d'utilisation de ce mécanisme: dans l'état normal du réseau le trafic entre les nœuds 1 et 2 traverse le lien direct entre les deux nœuds, mais dans la condition de défaillance sur ce lien le trafic est réorienté sur le cycle formé par les nœuds 1-2-3-4 (exemple (a)). Le p-cycle protège tous les liens qui forment le cycle mais aussi les liens entre deux nœuds du cycle, comme pour le lien entre les nœuds 2 et 3 (exemple (b)).

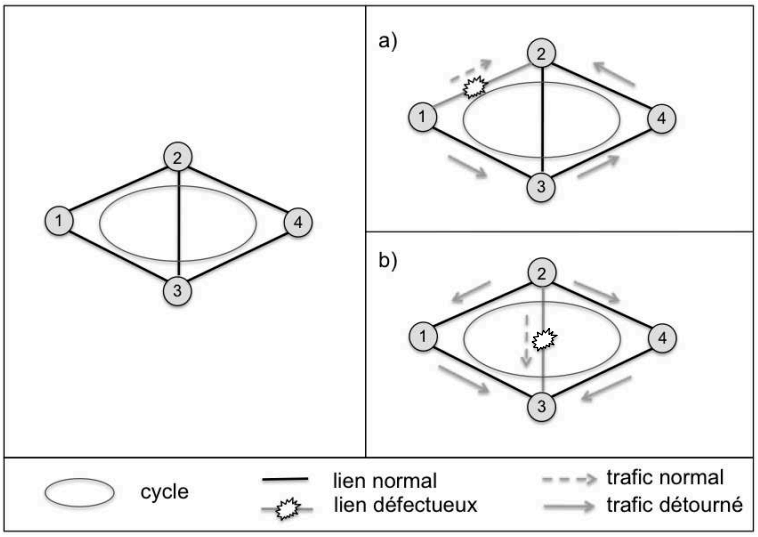


FIGURE 4.2 – Exemple de p-Cycle

4.2 Formation de boucles

4.2.1 Réseaux IP

Dans un réseau *IP* qui utilise un système d'adressage hiérarchique, un message envoyé dans une communication point à point peut arriver à sa destination en utilisant des chemins définis par le protocole de routage. Dans ce cas, même s'il existe des boucles dans la topologie physique, pendant le fonctionnement du réseau le message suivra son chemin sans problème. Pourtant s'il y a un changement de la topologie physique (à cause d'un problème ou d'une manœuvre de manutention) des boucles peuvent apparaitre pendant le temps de convergence du protocole de routage. Cette situation apparaît quand les tables des différents routeurs ne sont pas synchronisées et le message est envoyé à travers une mauvaise interface qui le ramène au même routeur. L'occurrence de boucles temporaires dans les réseaux *IP* a été étudiée dans [44]; dans l'article il est démontré que des boucles peuvent être formées à cause des disparités entre les tables de routage (*Forwarding Information Base (FIB)*) des multiples routeurs du réseau. Ces disparités sont causées par le fait que la propagation des informations sur un changement de topologie n'est pas immédiat et pendant le temps de convergence du protocole de routage, qui peut durer plusieurs secondes, des boucles peuvent se former dans le réseau.

L'établissement d'un ordre pour les changements des FIBs dans chaque routeur peut éviter les boucles, cette solution est proposée par [63] et [44]. Dans [64] on propose une solution pour éviter les boucles temporaires à travers quatre méthodes d'élimination de paquets différentes; pourtant il ressort de cet l'article

que ces méthodes ne sont efficaces que dans certaines conditions seulement, et que même la méthode la plus agressive ne peut pas éviter les boucles dans certaines conditions. Même pendant l'occurrence de la boucle, ses effets sur le fonctionnement du réseau peuvent être minimisés à travers l'élimination des paquets par expiration du TTL, évitant ainsi que le paquet circule indéfiniment dans la boucle, en provocant une surutilisation des ressources de transmission.

4.2.2 Réseaux Ethernet traditionnels

Dans un réseau Ethernet, le mécanisme de diffusion permet l'utilisation d'une adressage non hiérarchique (*flat address*): une trame arrivera à sa destination même si l'adresse de destination ne donne pas une indication sur sa localisation physique. La trame est ainsi envoyée à tous les commutateurs du réseau et cela garantit qu'elle arrivera à sa destination. Un réseau qui utilise le mécanisme de diffusion sans un mécanisme d'évitement de boucles ne pose pas de problème si la topologie ne possède pas de boucles physiques. Par contre dans un réseau maillé plusieurs boucles physiques peuvent exister, ce qui cause la multiplication des trames (les fameuses « tempêtes de diffusion »). Ce problème est évité dans les réseaux Ethernet commutés à travers la construction d'une topologie logique en arborescence en utilisant le *RSTP*; les trames sont donc diffusées seulement sur les liens qui font partie des branches de l'arbre. À cause des mécanismes de diffusion des trames, les problèmes causés par la formation de boucles dans les réseaux Ethernet peuvent être amplifiés. Le *RSTP* est par définition un protocole qui évite la formation de boucles, puisqu'il limite la diffusion des trames aux ports qui font partie de la topologie logique en arbre. Pourtant, selon l'architecture du réseau cette opération peut mener au problème de *« count to infinity »* et causer des boucles temporaires comme décrit par [65]. Dans l'article on montre un exemple de topologie où le problème de formation de boucles peut apparaître, la figure 4.3 montre cette topologie.

Dans cette topologie, le commutateur A est le commutateur racine. S'il existe une défaillance sur le lien entre A et B isolant le commutateur A, le commutateur B prend le rôle de racine puisqu'il n'a pas de chemin alternatif vers A. B envoie donc des messages (*BPDU*s) vers 3 et 4 pour annoncer son nouveau rôle (1). Le commutateur C va passer la nouvelle information vers 4, mais supposant que, pendant ce temps, en recevant le message envoyé par B, D lui propose un nouveau chemin vers A à travers le commutateur C (3), ce qui est une information fausse, B accepte cette information comme vraie et la renvoie vers 4. Cette fausse information propagée par D va circuler à travers les commutateurs jusqu'à ce que son temps de vie (*« MaxAge »*) soit écoulé. Pendant le temps de convergence du réseau les ports de la boucle physique

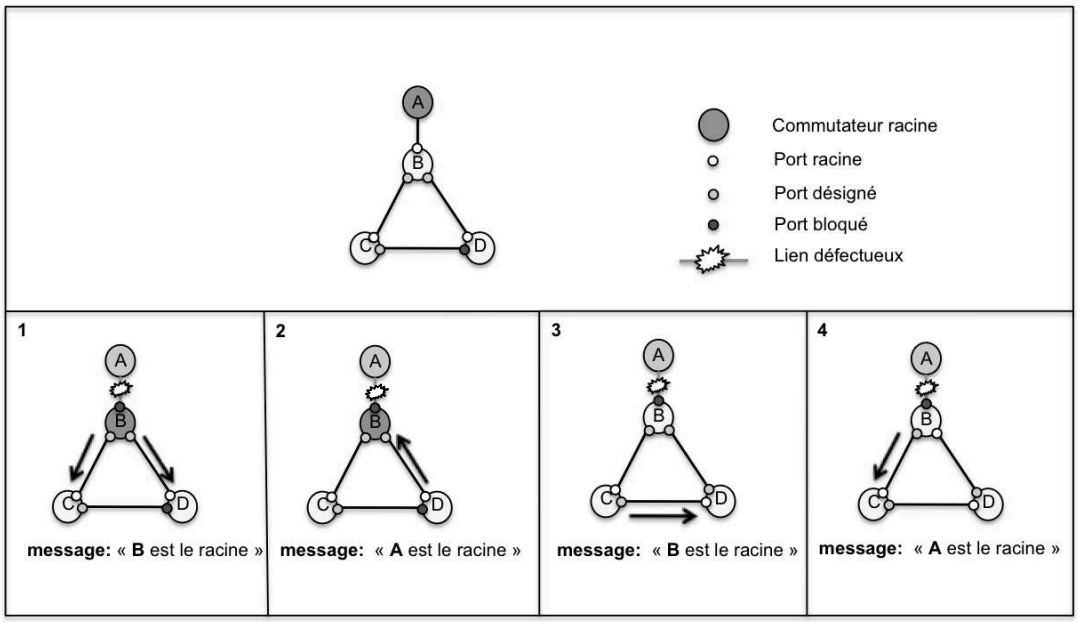


FIGURE 4.3 – Exemple de topologie

peuvent être dans un état de transmission dans un même moment, ce qui provoque une boucle pour les trames. Le mécanisme de formation de boucles à partir du *« count to infinity »* est démontré en [65]. Le fonctionnement du protocole *RSTP* dépend des algorithmes (*State Machines*) qui déterminent les transitions d'état des ports et des variables qui influencent le comportement du réseau comme décrit dans cet exemple. Ainsi le *« MaxAge »*, le nombre maximal de messages par port dans une période de temps (*« MaxAge »*), le *« HelloTime »*, le temps que le *BPDU* prend pour se propager et la séquence d'événements dans les différents états (*State Machines*) sont des facteurs qui vont déterminer l'occurrence de boucles.

4.2.3 Réseaux Ethernet à état de liens

Bien que les mécanismes du protocole *RSTP* qui causent la formation de boucles dans les réseaux traditionnels ne soient pas présents dans les réseaux qui utilisent un protocole à état de liens, un réseau qui utilise le protocole *IS-IS* n'est pas libre des problèmes causés par les changements de la topologie physique, puisque pendant le temps de convergence du protocole les différences entre les tables de routage des commutateurs peuvent mener à la formation de boucles pour les mêmes raisons que pour les occurrences dans les réseaux IP. L'article [64] montre une situation de formation de boucle, la figure figure 4.4 montre des diagrammes qui décrivent cet exemple.

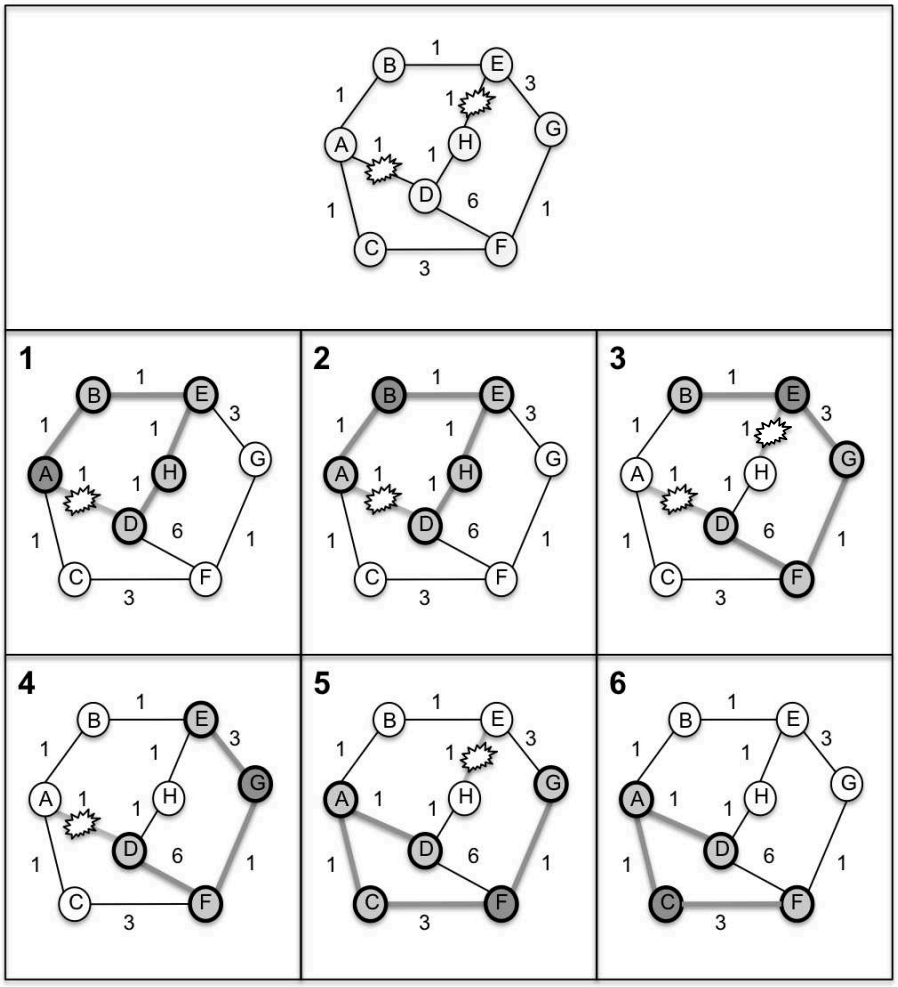


FIGURE 4.4 – Exemple de formation de boucle [64]

Dans l'exemple, le nœud A veut transmettre des messages vers D, les liens A-D et E-H présentent une défaillance qui n'est pas perçue par tous les nœuds au même moment. Dans chaque diagramme, les nœuds et liens en gris montrent le chemin calculé par le nœud qui reçoit le message (en gris foncé); les nœuds en blanc ne font pas partie du chemin établi par la table de routage de ce nœud spécifique. Les nœuds C et F ne sont pas au courant de la défaillance sur A-D et A,B,C ne savent pas pour E-H. Ainsi A envoie ses messages vers B qui les envoie vers E. Le message suit donc le chemin jusqu'à C qui la renvoie à A, en causant le boucle.

Le problème de formation de boucles dans les réseaux Ethernet qui utilisent un protocole de routage à état de liens est similaire à celui qui est rencontré dans les réseaux *IP*, pourtant le mécanisme de diffusion et l'absence du champ TTL dans les trames imposent l'utilisation de mécanismes pour éliminer ou diminuer les conséquences de la formation des boucles.

L'algorithme *Reverse Path Forwarding (RPF)* a été conçu pour les réseaux qui utilisent la transmission de paquets par diffusion. Cet algorithme est une technique de filtrage à l'entrée du paquet *(Ingress Checking)*. L'adresse d'origine est utilisée pour savoir si le paquet a été reçu par un port qui représente le meilleur chemin vers la source, et pour cela il suffit de vérifier le chemin optimal dans la *Forwarding Information Base (FIB)*, sans besoin de tables additionnelles. Si le port d'entrée est correct, le paquet est diffusé vers les autres ports. Ce mécanisme garantit la diffusion du paquet à tous les nœuds du réseau, mais produit aussi des copies indésirables sur des ports qui ne font pas partie de l'arbre couvrant minimal à partir de l'origine.

L'algorithme *RPF* peut être utilisé pour éviter la formation de boucles, mais cette technique ne peut pas garantir qu'aucune boucle ne sera formée. L'article [66] montre un exemple de topologie où un problème dans plusieurs liens peut mener à la formation d'une boucle à cause des disparités dans les tables de routage même si le *RPF* est utilisé, la figure 4.5 montre cet exemple. Dans cette topologie, une défaillance simultanée dans les liens B-F, C-F, D-F et E-F et qui n'est pas propagée instantanément parmi tous les nœuds du réseau va provoquer la boucle.

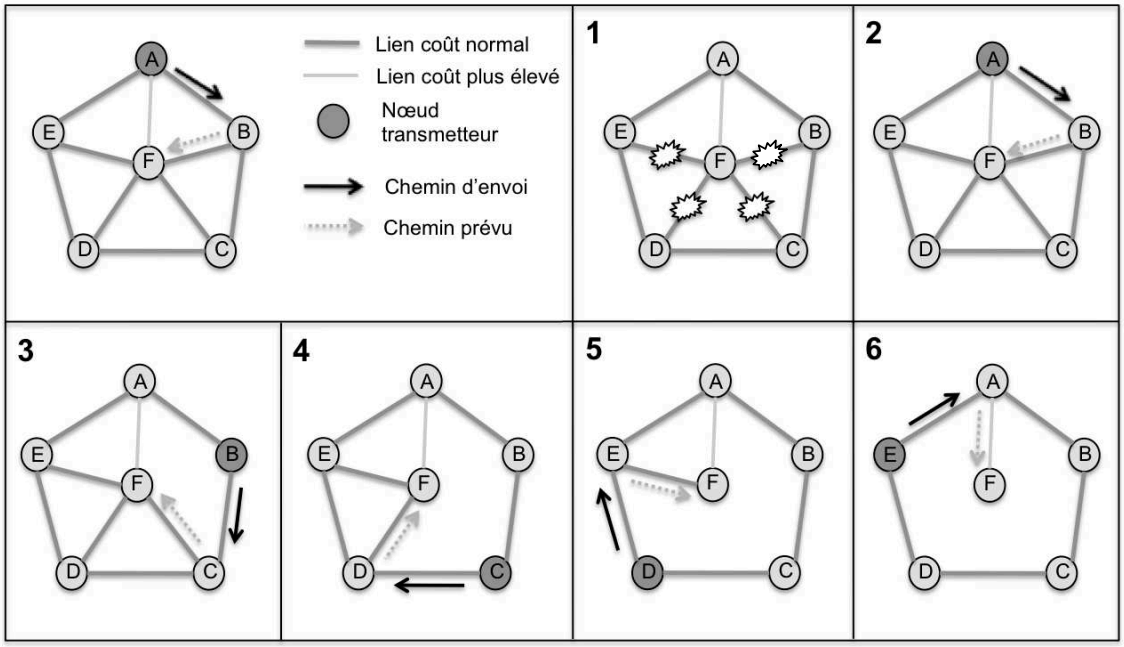


FIGURE 4.5 – Exemple de topologie [64]

Dans l'exemple, le commutateur A envoie des trames vers F à travers B. Au moment de la défaillance, A n'est pas au courant des modifications de la topologie physique et continue à envoyer les trames vers B. Celui-ci sait que son lien vers Y ne fonctionne pas et envoie la trame vers C qui, à son tour, l'envoie vers D. La même procédure est adoptée par D en envoyant vers E qui l'envoie vers A, causant ainsi la boucle.

Pour corriger ce problème, une extension de l'algorithme l'original est présentée; dans cette version, les paquets sont envoyés seulement vers les ports qui font partie de l'arbre, en vérifiant la source du paquet pour déterminer l'arbre *Source Based Forwarding*. Bien que dans [67] l'algorithme soit présenté comme une méthode de transmission de paquets en diffusion, il peut être utilisé comme un mécanisme pour éviter la formation de boucles. Les technologies *TRILL* et *SPB* utilisent le mécanisme *RPFC* qui combine le *RPF* et le Source Based Forwarding [68] [69]. Tout comme le protocole *IP*, le protocole *TRILL* implémente aussi un champ TTL.

4.3 Planification des réseaux

L'utilisation de mécanismes de recouvrement peut avoir un impact sur la planification du réseau, puisque le trafic réorienté par le mécanisme exige l'utilisation d'une capacité additionnelle qui doit être prise en compte. Dans le cas des réseaux avec un mécanisme de recouvrement l'objectif de la planification est de diminuer le coût lié à la capacité additionnelle. Il existe différentes formulations selon le type de mécanisme de recouvrement utilisé; dans le cas du recouvrement rapide à travers le rétablissement de liens (recouvrement local) une formulation du problème est présentée par [70]. L'équation (4.1) montre la fonction objectif qui vise à réduire le coût de la capacité additionnelle.

$$minimiser\ \mathbf{F} = \sum_{e} \xi_e (y_e + y'_e) \tag{4.1}$$

Dans cette équation ξ_e représente le coût unitaire du lien e, y_e la capacité normale du lien et y'_e la capacité de protection.

Les contraintes de cette fonction sont montrés dans (4.2)

$$\sum_{p} x_{dp0} = h_d \tag{4.2a}$$

$$\sum_{d} \sum_{p} \delta_{edp} x_{dp0} \leq y_e,\ e = 1, 2, ..., E \tag{4.2b}$$

$$\sum_{q} z_{eq} = y_e,\ e = 1, 2, ..., E \tag{4.2c}$$

$$\sum_{q} \beta_{leq} z_{eq} \leq y'_l,\ l = 1, 2, ..., E\ e = 1, 2, ..., E\ l \neq e \tag{4.2d}$$

Dans ces équations x_{dps} représente le flot de la demande d sur le chemin p ($s = 0$ est la condition normale du réseau), h_d est le volume de la demande d, δ_{edp} est une constante binaire (1 si le lien e appartient au chemin p de la demande d et 0 autrement), z_{eq} représente la capacité de rétablissement du lien e sur le chemin de rétablissement q, β_{leq} est une constante binaire (1 si le lien l appartient au chemin de rétablissement q du lien e).

La figure 4.6 montre un exemple de comparaison entre le coût relatif de la capacité additionnelle ($F' = \sum_e \xi_e y'_e$) de deux solutions de recouvrement différentes pour le réseau montré dans la figure 4.2. Dans cet exemple nous avons des demandes entre les nœuds adjacents qui correspondent aux flots sur les liens dans la condition normale du réseau. Par simplification on ne montre que les demandes dans une seule direction et on considère le coût unitaire pour tous les liens $\xi_e = 1$. Dans la figure (a) on montre le cas d'une défaillance du lien $e = 1$ ($s = 1$) et le recouvrement de la demande $d = 1$ sur le chemin le plus court (liens $e = 2$ et $e = 5$). Dans ce cas le coût relatif associé est $F' = 2 \times 2 = 4$ (correspondant à la capacité additionnelle totale nécessaire pour le recouvrement du lien $e = 1$ sur les liens $e = 2$ et $e = 5$). Si on considère le coût relatif pour le recouvrement sur le chemin le plus court de toutes les demandes dans les conditions de défaillance sur les autres liens (s=2,3,4,5), le coût relatif total associé sera $F' = 5 \times 2 = 10$. Pourtant, si on utilise un cycle formé par les liens $e = 1, 2, 3$ et 4 pour le recouvrement de tous les liens, ce coût sera $F' = 4 \times 2 = 8$, puisque seul les liens appartenant au cycle ont besoin de la capacité additionnelle. Dans cet exemple on voit que l'utilisation du cycle est avantageuse en termes de minimisation des coûts.

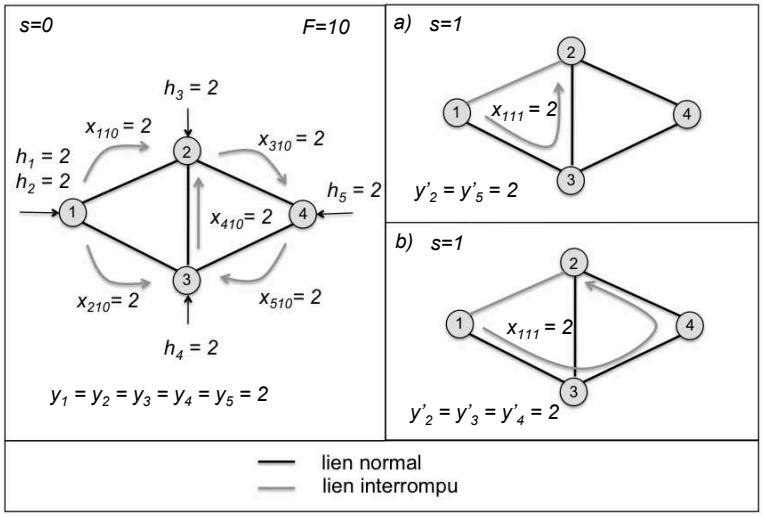


FIGURE 4.6 – Exemple de solutions

Dans un réseau avec une topologie maillée il existe plusieurs possibilités pour la protection à travers les p-Cycles, et la problématique pour la planification consiste à déterminer le groupe de cycles avec le coût

minimal. Dans [61] on présente une formulation pour le problème de minimisation de coûts dans un réseau *IP* en considérant le facteur sur-souscription sur les liens comme objectif pour assurer une *QoS* maximale. L'objectif de cette formulation est de minimiser le facteur de sur-souscription montré dans (4.3).

$$minimiser\ \eta_M \tag{4.3}$$

Dont les contraintes sont montrées dans (4.4):

$$\sum_{i=1}^{N_C} \delta_i \leq M \tag{4.4a}$$

$$\alpha_{i,j} \leq \delta_j\ \forall i = 1, 2, ..., N_S, j = 1, 2, ..., N_C \tag{4.4b}$$

$$\sum_{j=1}^{N_C} \alpha_{i,j} = 1\ \forall i = 1, 2, ..., N_S \tag{4.4c}$$

$$l_{i,j} = w_i + \sum_{k=1}^{N_C} \beta_{i,j,k} \cdot w_j \cdot \alpha_{j,k}\ \forall i = 1, 2, ..., N_S, j = 1, 2, ..., N_S, k = 1, 2, ..., N_C, i \neq j \tag{4.4d}$$

$$\eta_M \geq \frac{l_{i,j}}{c_i}\ \forall i = 1, 2, ..., N_S, j = 1, 2, ..., N_S, i \neq j \tag{4.4e}$$

Dans ces équations, η_M est le facteur sur-souscription maximal d'entre tous les liens pendant un événement de rétablissement, N_C est le nombre de cycles choisis dans le groupe du total de cycles dans le réseau, N_S est le nombre total de liens; δ_i est une constante binaire (1 si le cycle i est utilisé et 0 autrement); M est le nombre maximal de p-cycles utilisés (un nombre déterminé par le planificateur); w_i est la quantité de trafic sur le lien i en opération normale; c_i est la capacité totale du lien i; $l_{i,j}$ est le trafic total sur le lien i pendant le rétablissement du lien j; $\alpha_{i,j}$ est une variable égale à la proportion du trafic sur le lien i que le cycle j devra restaurer; et $\beta_{i,j,k}$ est la proportion de charge sur le lien i quand le cycle k est utilisé pour restaurer le lien j.

Pour résoudre analytiquement les problèmes de planification des réseaux en utilisant des *p-Cycles*, différentes méthodes peuvent être utilisées, comme par exemple l'optimisation linéaire proposée par [60]. Dans [71] nous pouvons trouver une classification des méthodes d'optimisation.

Dans ce chapitre nous avons décrit les mécanismes de recouvrement rapide utilisés dans les réseaux IP. Ces mécanismes ont été créés pour diminuer le temps d'interruption provoqué par la convergence du

protocole de contrôle après l'occurrence d'une défaillance. Le mécanisme *p-Cycle* est un mécanisme de protection local. Il permet l'obtention d'un temps de recouvrement du même ordre de grandeur que celui obtenu dans les réseaux en anneau; cette solution peut être utilisée dans les réseaux de différentes technologies. La formation de boucles pendant le temps de recouvrement du protocole de contrôle est un problème qui peut apparaître dans des différents réseaux, mais l'algorithme *RPF* utilisé dans le réseaux Ethernet contrôlés par le *IS-IS* peut contourner ce problème. Pourtant, l'utilisation du *RPF* représente un défi pour l'implémentation d'un mécanisme de recouvrement rapide basé sur les réseaux IP. La création d'un mécanisme de recouvrement rapide à travers des cycles, pour un réseau Ethernet contrôlé par un protocole à état de liens est le sujet du prochain chapitre. L'utilisation de cycles permet la création d'un mécanisme de récupération capable de diminuer le temps de recouvrement du réseau. Ce temps de recouvrement est indépendant de la topologie utilisée. Cette solution permet aussi l'obtention d'un coût de capacité additionnelle prévisible et optimisable à travers la planification du réseau.

Chapitre 5

Proposition d'un mécanisme de recouvrement rapide

La création d'un mécanisme de protection rapide dans un réseau carrier Ethernet qui utilise un protocole à état de liens dans le plan de contrôle présente des défis pour son implémentation. Contrairement aux réseaux *IP*, à cause de la construction d'arbres de recouvrement, l'implémentation exige la conception de nouveaux mécanismes pour remédier aux problèmes causés par l'utilisation de l'algorithme *RPF* et de la transmission par diffusion. Dans ce chapitre nous proposons l'utilisation d'un mécanisme de protection locale qui utilise des tunnels pour les paquets réorientés. À la fin du chapitre nous faisons l'analyse du mécanisme à travers des simulations et une comparaison avec le mécanisme de recouvrement global du protocole *IS-IS*.

5.1 Description du mécanisme

L'utilisation de l'algorithme *RPF* pour éviter les boucles consiste à bloquer la circulation des paquets sur des liens qui ne font pas partie de l'arbre de diffusion. Pourtant, dans le cas d'un re-routage fait à travers l'utilisation d'un mécanisme de protection rapide il est nécessaire que les paquets passent par un chemin alternatif. La figure 5.1(a) montre un exemple du problème. Dans cette figure le nœud 5 détecte la défaillance sur le lien vers le nœud 8; en utilisant une protection à travers le chemin alternatif (cycle) passant par les nœuds 4 et 7, les paquets peuvent utiliser un chemin alternatif pour arriver à destination. Pourtant à cause de l'algorithme *RPF*, ces paquets seraient rejetés à l'entrée du nœud 4.

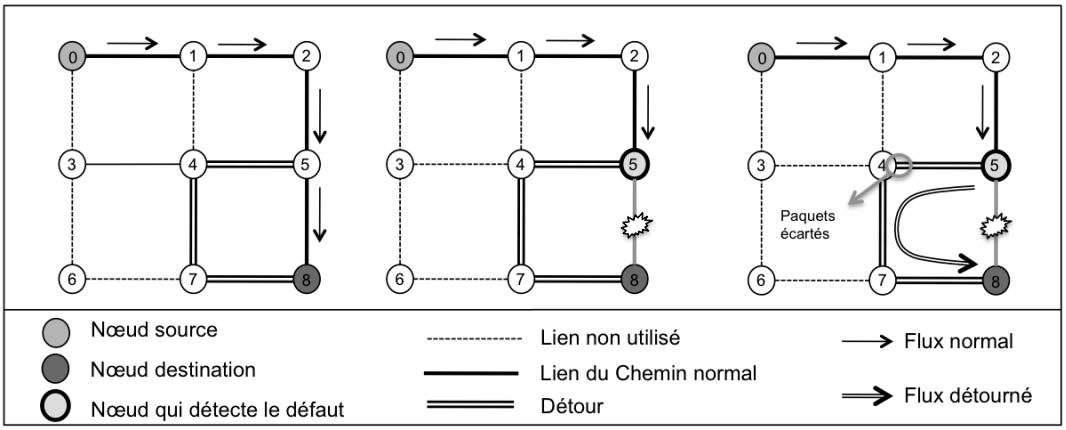


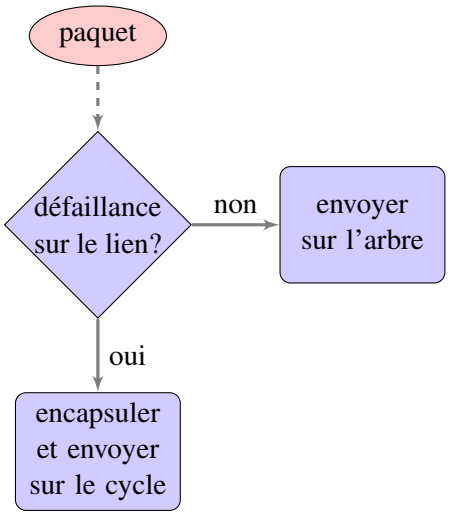
FIGURE 5.1 – Exemple du mécanisme de protection

Pour éviter l'élimination des paquets réorientés nous proposons l'utilisation de tunnels, soit une encapsulation de ces paquets et l'utilisation de tables de routage spéciales à leur intention dans les nœuds qui font partie du chemin de contournement; la figure 5.2 montre un logigramme du mécanisme (le pseudo-code de cet algorithme est montré dans l'annexe A.1). Le nœud qui détecte la défaillance doit encapsuler les paquets vers le lien alternatif, qui fait partie de la boucle (cycle) vers le nœud qui se trouve à l'autre extrémité du lien défectueux (le dernier nœud du cycle). La figure 5.3 montre un exemple de table de routage. La table normale est utilisée par un nœud qui n'exploite pas le mécanisme; dans ce cas, seules les informations d'origine, destination et ports d'entrée (Pin) et sortie (Pout) sont présentes. La table de routage avec cycle est la table spéciale pour un réseau qui exploite le mécanisme; dans ce cas, on ajoute les informations d'origine et destination, ainsi que les ports d'entrée (PTin) et sortie (PTout) des tunnels. Les paquets réorientés ont comme adresse d'origine le nœud qui détecte la défaillance et comme adresse de destination le dernier nœud du cycle.

5.2 Problèmes

L'utilisation de tunnels et de tables de routage pour le cycle est une solution simple, pourtant le mécanisme présenté précédemment peut causer la transmission de paquets dupliqués sur certains liens du cycle. Ce problème est montré dans la figure 5.4. Dans la figure, le nœud 0 détecte la défaillance et commence la transmission des paquets encapsulés sur le chemin alternatif vers le nœud 1. Pourtant la destination de ces paquets est le nœud 8, ce qui oblige une retransmission du même paquet (cette fois sans la nouvelle encapsulation) sur le lien $1 \rightarrow 4$ jusqu'à la destination, comme montré par la figure 5.4(b).

(a) Nœud qui détecte la défaillance



(b) Nœud sur le chemin de détour

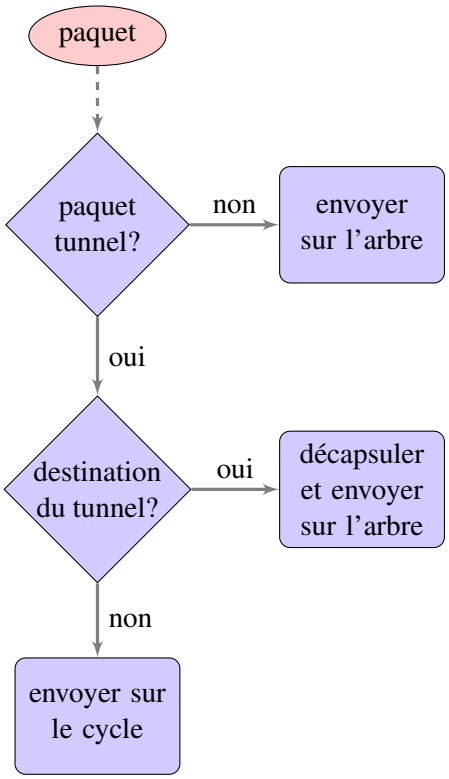


FIGURE 5.2 – Logigramme du mécanisme

Nœud A – Ports {a..d}

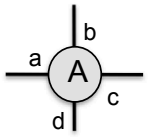


Table normale

Origine	Destination	P_{in}	P_{out}
A	D	--	b
⋮	⋮	⋮	⋮
B	E	--	--

Table avec cycle

Origine Tunnel	Destination Tunnel	Origine	Destination	P_{in}	P_{out}	PT_{in}	PT_{out}
A	B	A	D	--	b	--	c
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
B	C	B	E	--	--	a	c

FIGURE 5.3 – Exemple de table

Une solution à ce problème pourrait être d'offrir aux nœuds qui font partie du cycle des informations sur les arbres alternatifs pour le chemin de la source vers la destination. Par exemple, dans la figure 5.4(c), le nœud 4 connaît à priori le chemin utilisé pour les paquets transmis par 0 vers 8 dans le cas d'une défaillance sur le lien $0 \rightarrow 1$; ainsi, au lieu de détourner les paquets encapsulés vers 1, il envoie les paquets originaux vers

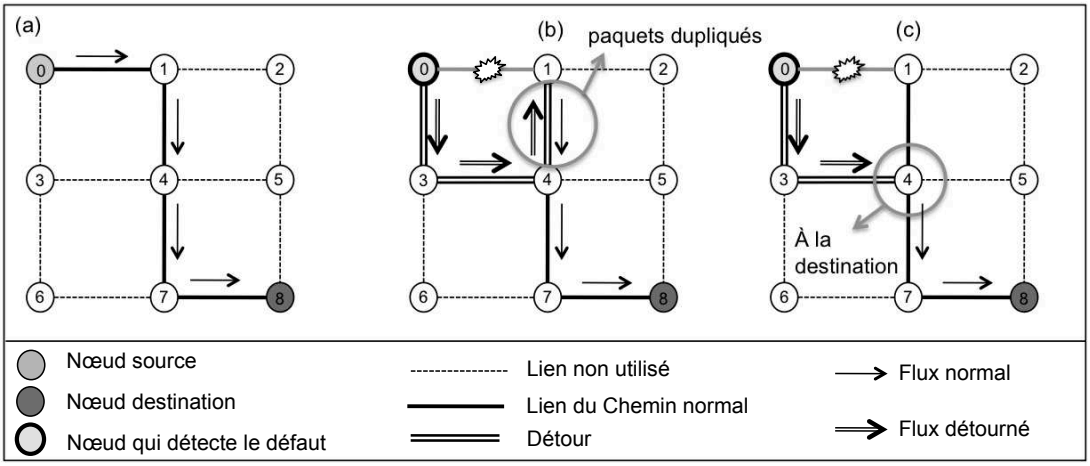


FIGURE 5.4 – Retransmission des paquets

7. Pourtant une telle solution exige l'utilisation de tables de routage supplémentaires avec une complexité qui augmente proportionnellement au nombre de liens à protéger et au nombre de nœuds du réseau.

La figure 5.5 montre un exemple dans lequel un nœud A veut envoyer des paquets vers un nœud B à travers un chemin qui utilise le lien entre les nœuds 0 et 1. Dans ce scénario, on considère l'utilisation d'un mécanisme au niveau physique pour la gestion des défaillances, le problème est détectée par le niveau physique et communiqué à travers un message au niveau de contrôle (les détails du mécanisme de détection sortent du cadre de notre étude). Avant la défaillance, en recevant les paquets normaux les nœuds utilisent leurs tables normales (Tn) pour l'acheminement des paquets reçus à travers les ports d'entrée (Pin) vers les ports de sortie (Pout). Après la défaillance sur le lien 0-1 ces nœuds utilisent leurs tables tunnel (Tt) pour acheminer les paquets tunnel reçus à travers les ports d'entrée (PTin) vers les ports de sortie (PTout). Ainsi le nœud 0 détecte la défaillance et encapsule les paquets reçus à travers son port 1; ces paquets sont alors transmis vers le cycle 0-3-4-1 à travers son port 0. En recevant les paquets tunnel sur son port 1, le nœud 3 analyse les adresses du tunnel et les adresses originales (dans ce cas il n'y a pas d'entrée dans la table Tn pour ces adresses puisque les paquets originaux utilisent le chemin sur le nœud 1), donc le nœud 3 achemine les paquets tunnel sans modification à travers son port 2. En recevant le paquet tunnel le nœud 4 vérifie les adresses d'origine et de destination du tunnel et les adresses originales. Cette fois il existe des entrées dans les tables Tn et Tt, mais pourtant les ports Pout et PTout sont différents, ce qui signifie que ce nœud possède un meilleur chemin vers la destination originale. Dans ce cas, il désencapsule le paquet tunnel et envoie le paquet original sur le chemin plus court vers la destination. Les nœuds 7 et 8 et les nœuds suivants utilisent les tables Tn pour faire l'acheminement des paquets originaux.

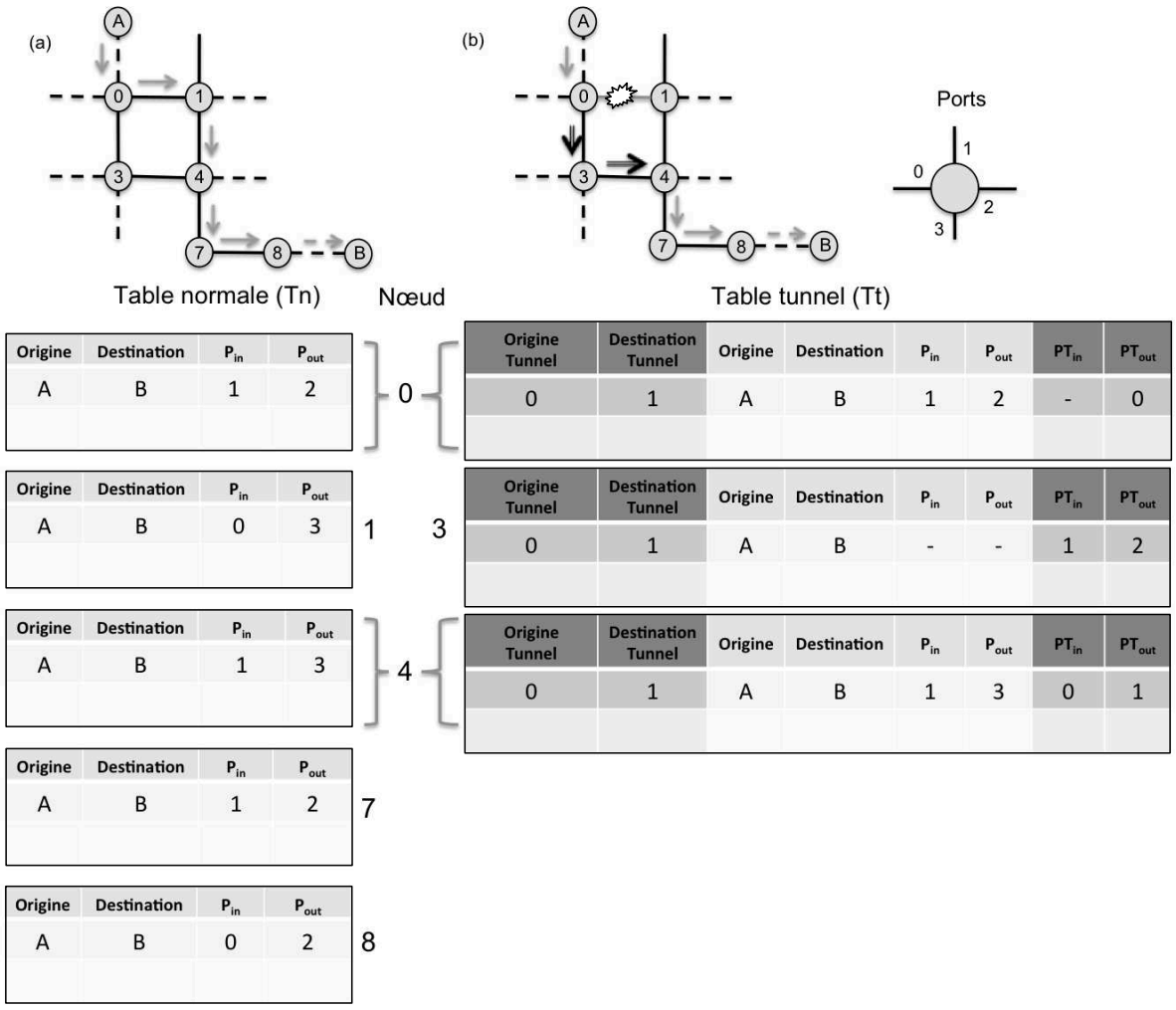


FIGURE 5.5 – Transmission point-à-point

Dans le cas d'une transmission par diffusion (point-à-multipoint) des copies du même paquet doivent arriver à destination. Donc, si le mécanisme de protection détourne des paquets on peut avoir des liens avec des copies du même paquet, ce qui est montré dans la figure 5.6(a). Dans la figure, après la détection du problème par le nœud 5 et le détournement des paquets dans ce nœud vers 8, le nœud 4 doit les envoyer vers 7. Pourtant il existe déjà un flot des mêmes paquets sur ce lien, ce qui va causer la transmission dupliquée du même flot sur le lien $4 \rightarrow 7$.

Une solution possible serait l'analyse des flots par les nœuds qui font partie du cycle. Par exemple, si un nœud dans le cycle reçoit des paquets tunnel (qui appartiennent à un flot déterminé) et le lien sur lequel il devra envoyer ces paquets tunnel transporte déjà ce même flot, il devra choisir d'envoyer seulement une des copies (les paquets tunnel ou les paquets du flot existant). Par contre, les paquets du flot existant et les paquets

tunnel n'arrivent pas au même moment dans le nœud puisqu'ils suivent des différents chemins, et donc la décision de faire un choix entre les deux va forcément provoquer des pertes ou la duplication de paquets. La figure 5.6(b) montre un exemple où le nœud 4 doit envoyer des paquets encapsulés vers le nœud 7; ces paquets appartiennent au flot original qui circulait sur ce lien et qui n'est pas synchronisé avec les paquets réacheminés par 5. Dans ce cas, le nœud choisit de transmettre seulement les paquets du flot existant, cette fois encapsulés. Le nœud 7 recevra un seul flot, mais enverra des copies vers le nœud 8 qui va recevoir ces copies au lieu des copies envoyées par le nœud 5; il y aura donc des paquets perdus qui n'arriveront pas au nœud 8. Ainsi que dans le cas de la transmission point-à-point analysée précédemment les nœuds qui font partie du cycle ont besoin d'informations supplémentaires pour réaliser leurs décisions.

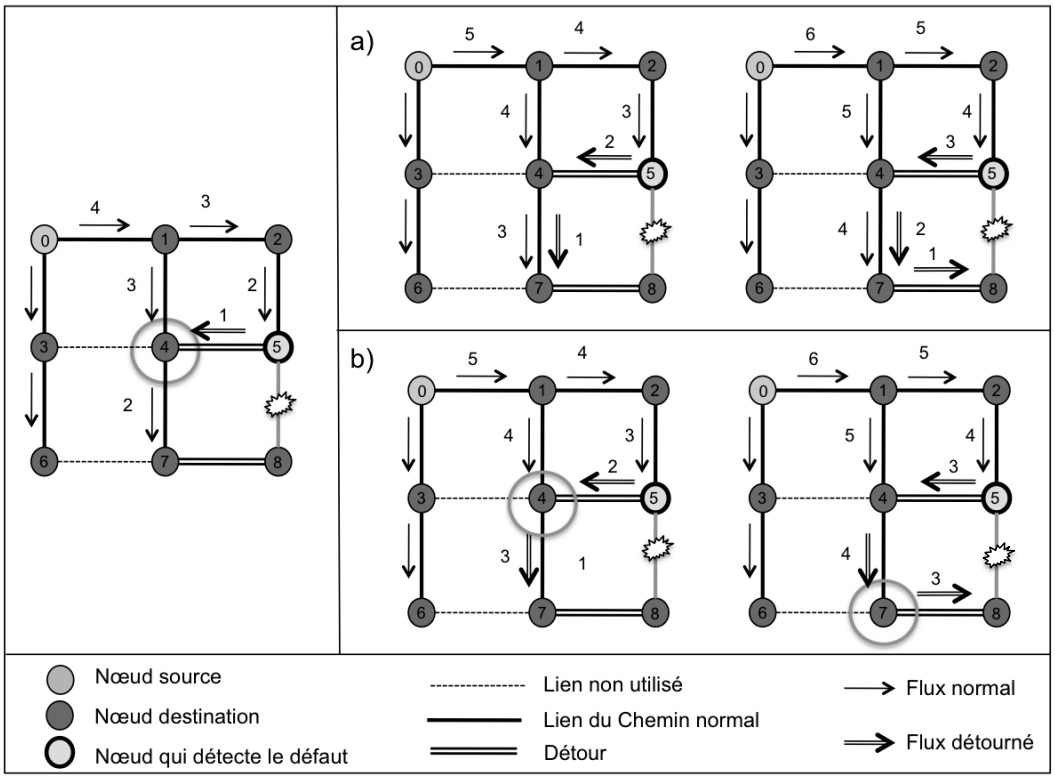


FIGURE 5.6 – Transmission point-à-multipoint

La figure suivante montre une solution avec l'utilisation de tables supplémentaires dans chaque nœud qui fait partie du cycle. La figure 5.7(a) montre la situation normale du réseau où chaque nœud dans le cycle utilise la table Tn pour l'acheminement sur l'arbre de recouvrement. Dans le cas de défaillance sur le lien 5-8 montré dans la figure 5.7(b), les nœuds utilisent la table Tt; le nœud 5 détecte la défaillance et envoie des paquets tunnel vers le cycle 5-4-7-8 à travers le port 0, alors que les paquets qui n'ont pas été affectés par la défaillance continuent à être envoyés vers l'arbre de recouvrement, à travers le port 2. En recevant le paquet tunnel sur le port 2, le nœud 4 est informé du problème. L'analyse de l'origine et destination du

tunnel permet la découverte du cycle à utiliser, et le nœud fait aussi l'analyse de l'origine et destination originales, et ainsi pour éviter l'envoi de flots dupliqués il va envoyer les paquets reçus à travers son port 1 dans un paquet tunnel à travers son port 3. En recevant le paquet tunnel, le nœud 7 utilise Tn pour envoyer des paquets normaux à travers son port 3 et des paquets tunnel à travers le port 2.

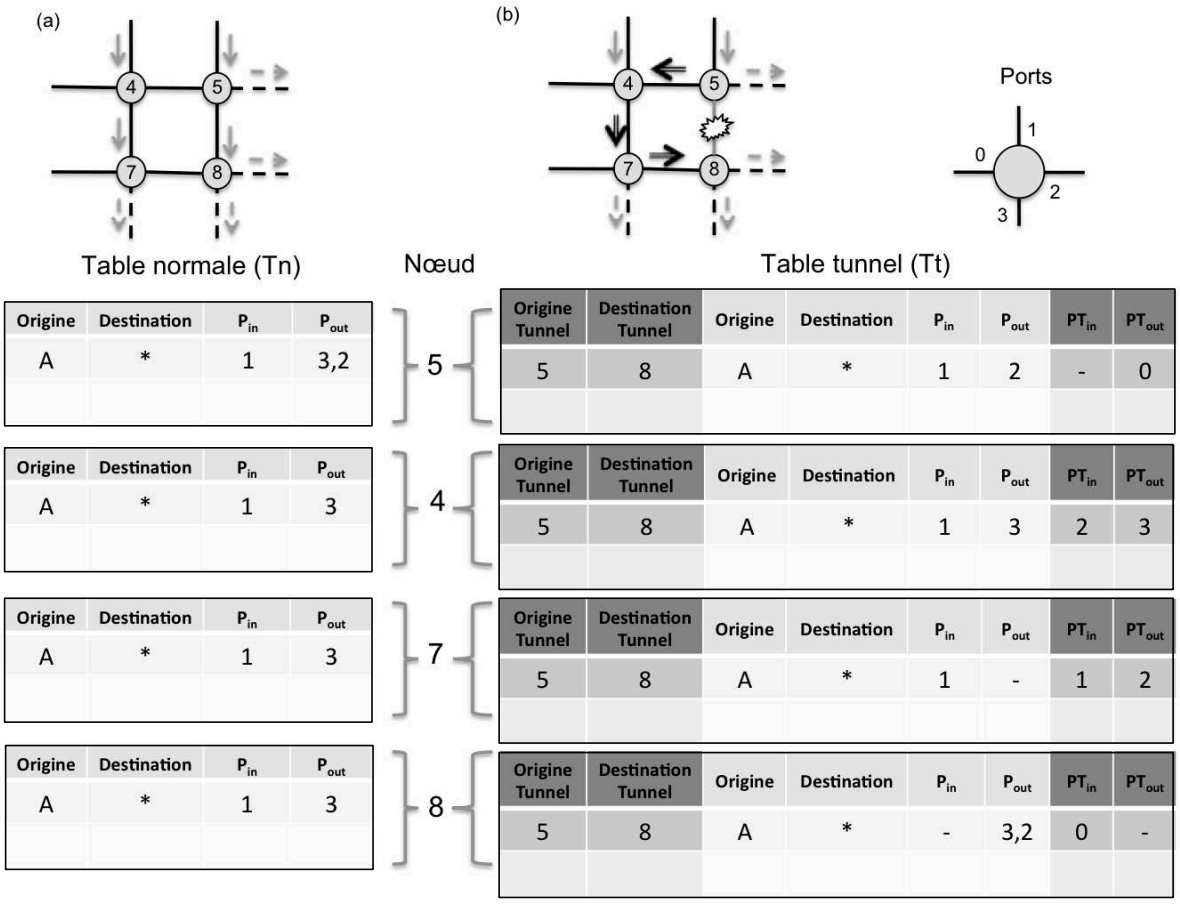
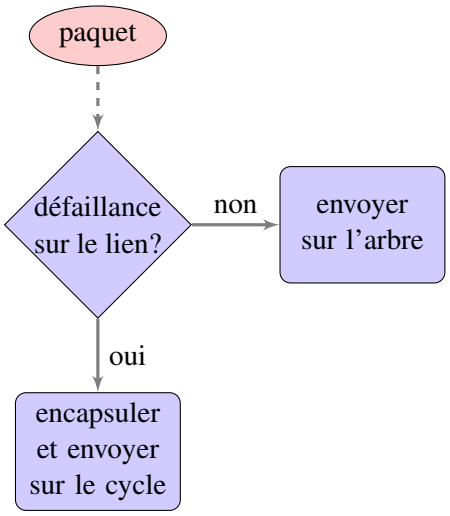


FIGURE 5.7 – Example de table point-à-multipoint

Les figures 5.8 et 5.9 montrent les logigrammes de la solution présentée précédemment, dans les cas des transmissions point-à-point et point-à-multipoint (le pseudo-code de cet algorithme est montré dans l'annexe A.2). La complexité des tables de routage tunnel augmente avec la quantité de liens à protéger; ainsi un nœud qui fait partie d'un cycle avec n_L liens à protéger doit calculer des arbres alternatifs pour chaque lien pour construire la table Tt. Par exemple si la table Tn a n entrées donc la table Tt aura $n \times n_L$ entrées. Les figures 5.10 et 5.11 illustrent cet exemple.

(a) Nœud qui détecte la défaillance



(b) Nœud sur le chemin de détour

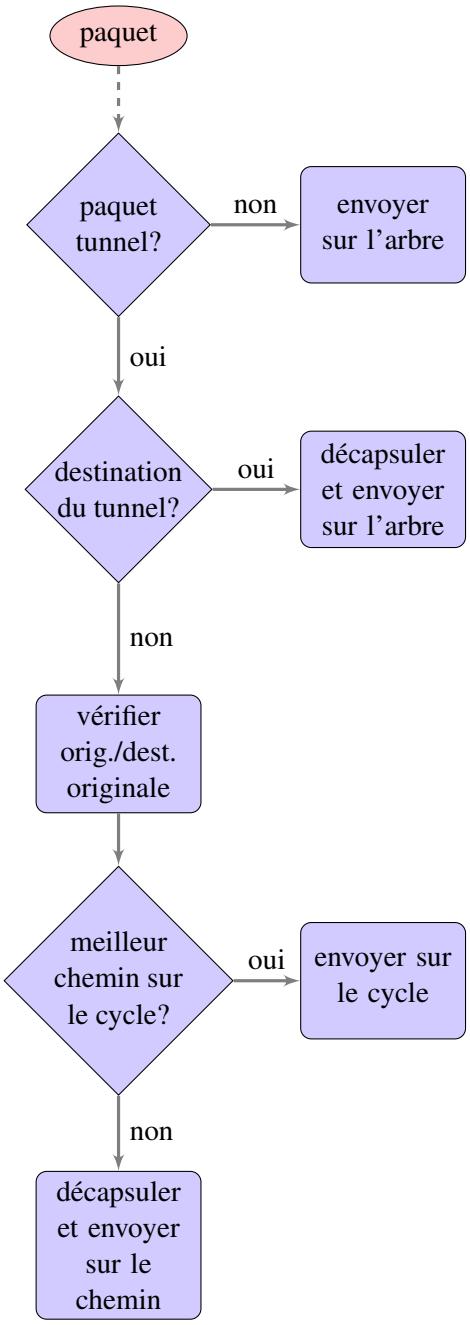
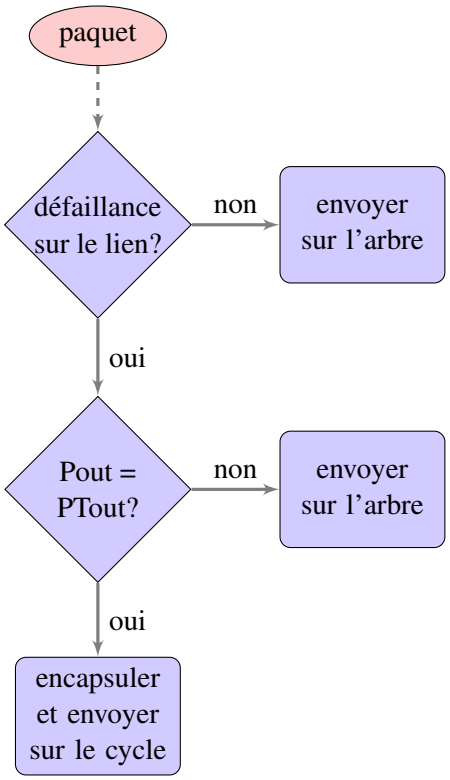


FIGURE 5.8 – Logigramme du mécanisme pour éviter les flots dupliqués (point-à-point)

5.3 Simulation II - Analyse du mécanisme de recouvrement rapide

La deuxième simulation a été conçue pour faire une analyse de l'impact du temps de convergence et du mécanisme de protection rapide sur le trafic de données. Pour cette analyse, nous mesurons le nombre de paquets perdus pendant le recouvrement du réseau. Nous analysons le cas d'un réseau protégé par la

(a) Nœud qui détecte la défaillance



(b) Nœud sur le chemin de détour

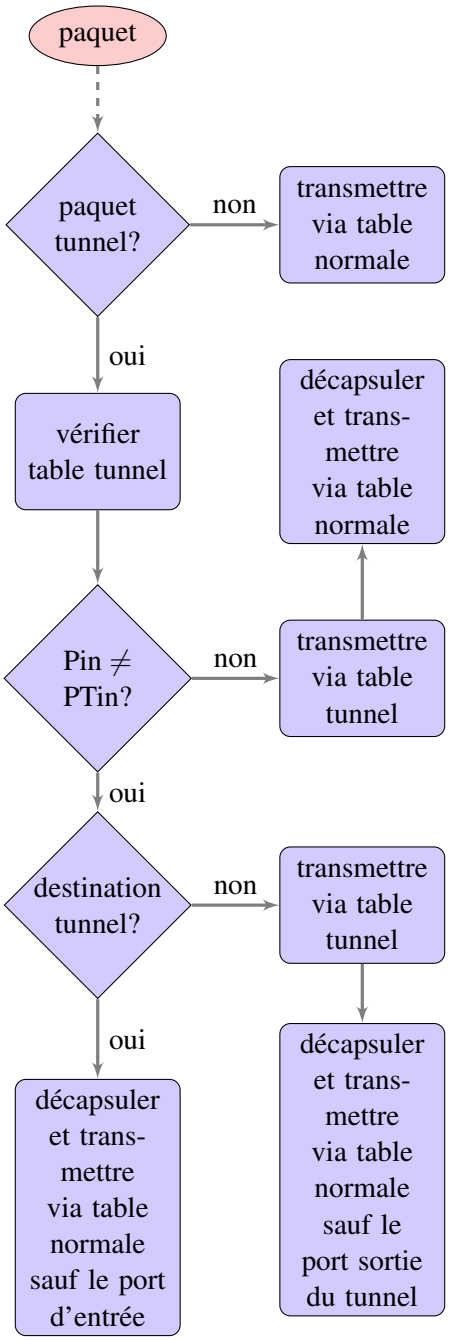


FIGURE 5.9 – Logigramme du mécanisme pour éviter les flots dupliqués (point-à-multipoint)

convergence normale du protocole et d'un réseau protégé par le mécanisme de recouvrement rapide proposé. Cette simulation a comme objectif de mesurer l'amélioration offerte par l'utilisation du mécanisme.

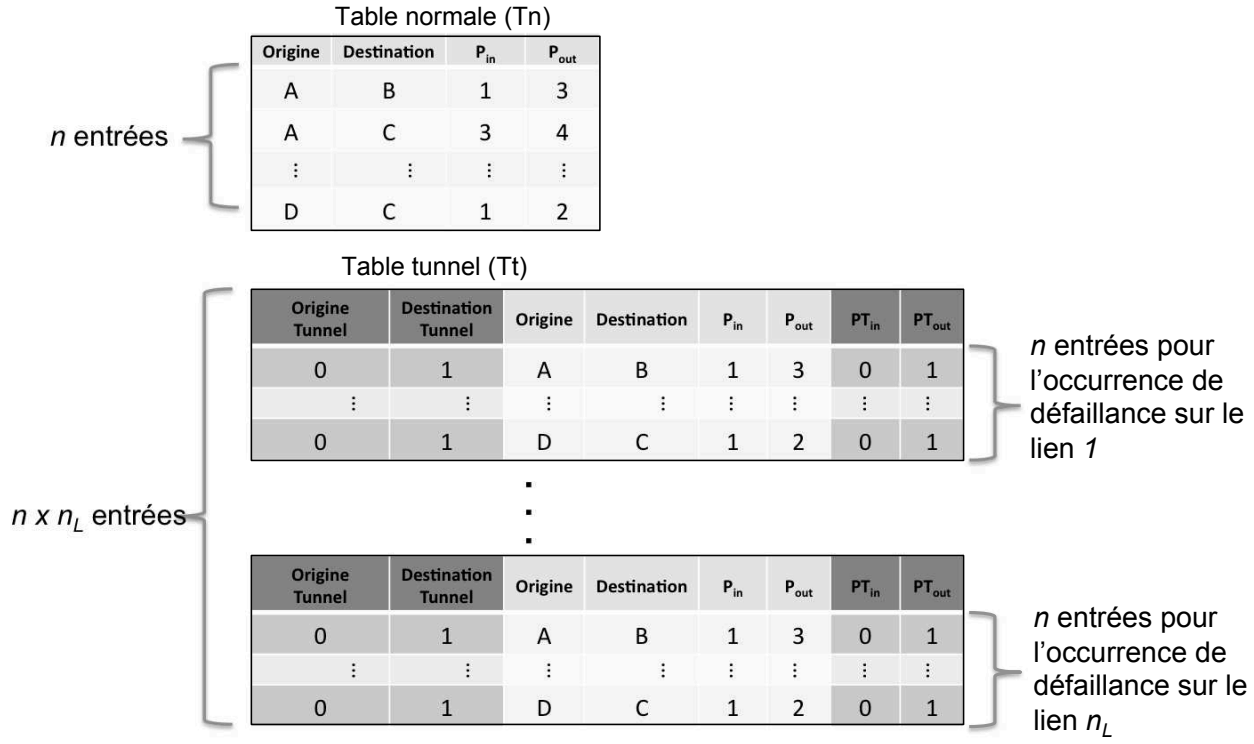


FIGURE 5.10 – Construction des tables Tt

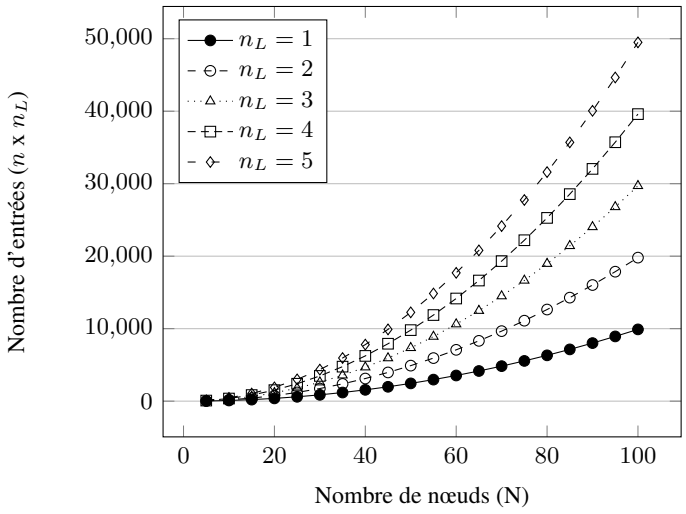


FIGURE 5.11 – Taille de la table de routage selon le nombre de nœuds et liens à proteger

5.3.1 Méthode d'analyse

Dans notre premier réseau simulé, le trafic est généré entre des stations à travers un réseau composé de commutateurs qui utilisent le protocole *IS-IS* dans le plan de contrôle. Un deuxième réseau simule l'utilisation du mécanisme de recouvrement rapide (*FRR*). Pour cette simulation du protocole *IS-IS*, nous utilisons le

simulateur *OMNeT++* et l'implémentation du protocole *IS-IS* avec les modifications décrites dans notre première simulation. La défaillance sur un des liens est simulée et le nombre de paquets perdus est utilisé pour l'analyse du temps de recouvrement. Le test est répété 20 fois pour chaque réseau. Pour cette simulation, on simule également un temps de réaction pour le mécanisme qui détecte la défaillance avec une distribution uniforme entre 0.1ms et 1ms: une fois que le commutateur détecte la défaillance, le mécanisme de protection va agir seulement après ce temps de réaction. Pour les simulations on a considéré des topologies en grille, puisque le mécanisme de récupération va agir localement, seulement les nœdus qui font partie du cycle sont impliqués dans la procédure, les résultats peuvent être étendus à d'autres topologies plus complexes. Le trafic simulé est du type *Constant Bit Rate (CBR)*, ce qui permet un rapport direct entre le nombre de paquets perdus et le temps de recouvrement du réseau, nous faisons de cette façon un test de référence pour mesurer indirectement la capacité de récupèration rapide du mécanisme, donc on ne considère pas l'impact sur d'autres types de trafic ni les applications. Le service *CBR* représente aussi un trafic critique, parce qu'il est utilisé pour le transport de audio et vidéo en temps réel [2].

5.3.2 Le mécanisme de recouvrement rapide

Le mécanisme de recouvrement rapide simulé utilise comme base les mécanismes utilisés pour les réseaux IP. Au moment d'une défaillance sur un lien entre deux nœuds, le mécanisme va fonctionner à travers le détournement des paquets vers le deuxième lien qui fait partie de la boucle entre les deux nœuds. Pour éviter le rejet des paquets à la réception provoqué par le mécanisme d'évitement de boucles, les paquets réorientés sont enveloppés avec un entête, ce qui va signaler au nœud de réception que ce paquet ne doit pas être rejeté même s'il est reçu à travers un lien qui ne fait pas partie de l'arbre de diffusion originel. Ensuite nous montrerons les tests réalisés pour la comparaison entre le temps de recouvrement des flots de donnés dans un réseau qui ne possède pas le mécanisme de recouvrement rapide et un réseau où le mécanisme a été implémenté.

5.3.3 Tests réalisés

Traffic point-à-point

Le premier test est celui de la simulation d'un trafic de type *CBR* point-à-point entre deux stations. La topologie utilisée pour ce test est une grille avec 9 nœuds, une source et un récepteur du trafic *CBR*. La

défaillance est simulée sur le lien du nœud attaché à la source qui fait partie de l'arbre de diffusion, ce qui oblige le détournement du trafic sur un chemin alternatif au moment de la défaillance. La figure 5.12 montre un diagramme du test. Avant la défaillance, le trafic circule à travers le chemin correspondant à l'arbre de diffusion original entre l'origine et la destination; au moment de la défaillance le nœud attaché au lien interrompu commence à envoyer des paquets enveloppés pour le tunnel vers la destination sur un deuxième lien. Pour éviter la retransmission de paquets sur un même lien, dans cette simulation on ne traite que les nœuds sur le chemin qui connaissent un chemin alternatif pour les paquets réorientés.

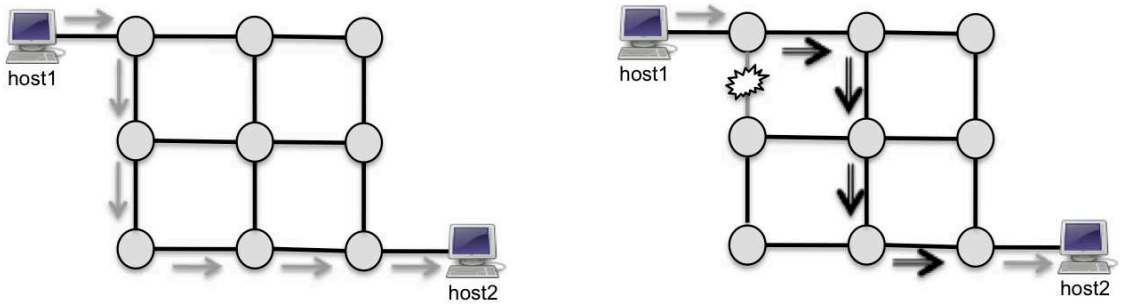


FIGURE 5.12 – Topologie pour la simulation point-à-point

Traffic de diffusion

Le deuxième test est celui de la simulation d'un trafic *CBR* de diffusion généré par une station vers plusieurs stations. La topologie utilisée est composée de quatre nœuds, une source et 3 récepteurs du trafic. Les nœuds sont interconnectés en anneau (cette topologie peut représenter une section d'un réseau plus grand) et la défaillance est simulée sur le lien entre les deux premiers nœuds. La figure 5.13 montre un diagramme du test.

FIGURE 5.13 – Topologie pour la simulation de diffusion

Le troisième test utilise la topologie en anneau vue précédemment et deux autres topologies en grille (N=9 et N=16), montrées dans la figure 5.14. Dans ce test, on analyse la perte de paquets avec la croissance du réseau. Comme dans le test précédant, on simule une défaillance sur le lien entre les deux premiers nœuds.

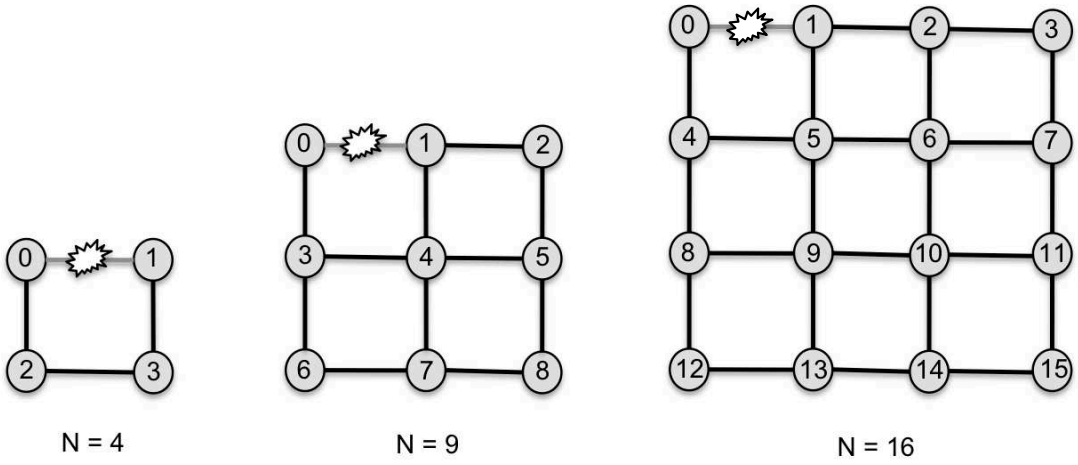


FIGURE 5.14 – Topologies en grille pour la simulation de diffusion

5.3.4 Résultats

La figure 5.15 montre le résultat du trafic perdu dans le cas du trafic point-à-point. On voit clairement que l'utilisation d'un mécanisme *FRR* présente un nombre de paquets perdus moins élevé que dans le cas d'un recouvrement qui dépend du temps de convergence du protocole de contrôle.

La figure 5.16 montre le résultat pour le nombre de paquets perdus dans le cas du trafic point-à-multipoint; une fois de plus on peut observer la différence causée par l'utilisation d'un mécanisme de recouvrement rapide. On peut également remarquer que dans le cas du protocole *IS-IS* sans recouvrement rapide le nombre de paquets perdus dépend de l'emplacement de la destination, une fois que les commutateurs ne réalisent pas la mise-à-jour des tables au même moment.

La figure 5.17 montre le coefficient de corrélation entre le temps de convergence de chaque commutateur (dernière mise-à-jour de sa table de routage) et le temps de convergence global Tc (dernière mise-à-jour parmi tous les commutateurs); on peut observer que les commutateurs B1 et B2 représentent mieux le temps global Tc. Dans la table 5.1 on montre la corrélation entre les paquets perdus de chaque destination et les temps de convergence; on observe que la corrélation la plus représentative est celle entre la station 2 et le commutateur 4 et entre la station 4 et le commutateur 3. Pour le *FRR*, il n'existe pas de corrélation significative entre les pertes et le temps de convergence, vu que le temps d'interruption ne dépend pas de la

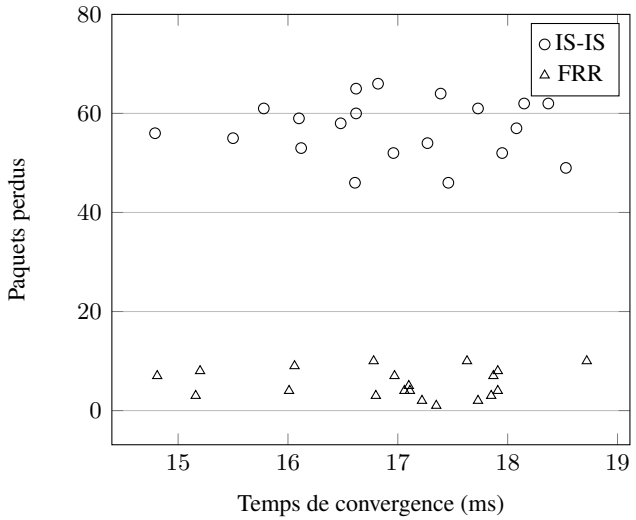


FIGURE 5.15 – Nombre de paquets perdus

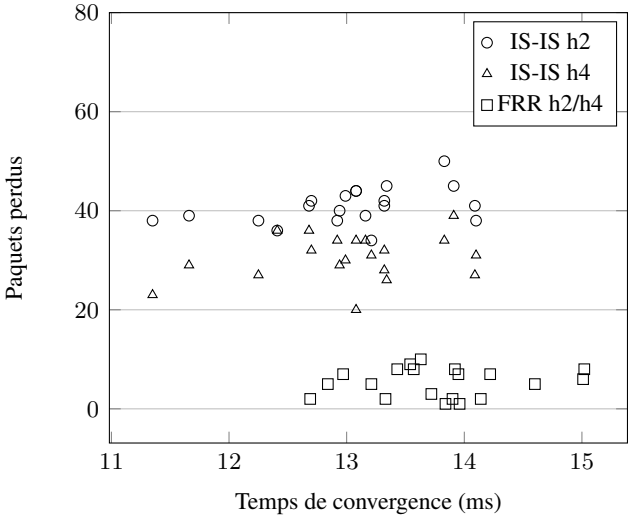


FIGURE 5.16 – Nombre de paquets perdus pour host2 et host4

convergence du protocole de contrôle. La figure 5.18 montre la perte de paquets des stations h2 et h4 en relation avec les temps de convergence des commutateurs B4 et B3; on observe que cette relation est plus représentative que celle montrée dans la figure 5.16 où le temps de convergence global a été utilisé.

Tableau 5.1 – Correlation entre paquets perdus et Tc

	B1	B2	B3	B4	Tc
IS-IS (h2)	0.14	0.59	0.07	**0.70**	0.42
IS-IS (h4)	0.58	0.00	**0.67**	0.14	0.27
FRR (h2,h4)	0.01	0.22	0.02	0.09	0.11

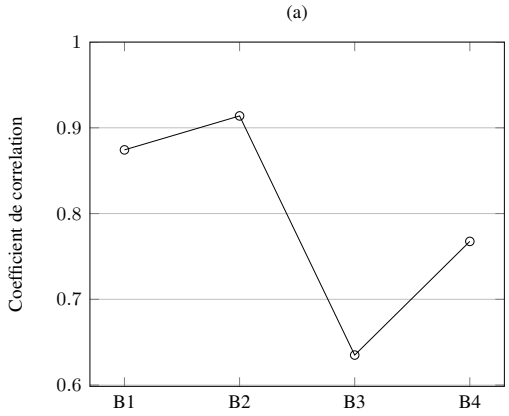


FIGURE 5.17 – Corrélation entre le temps de convergence des commutateurs et Tc

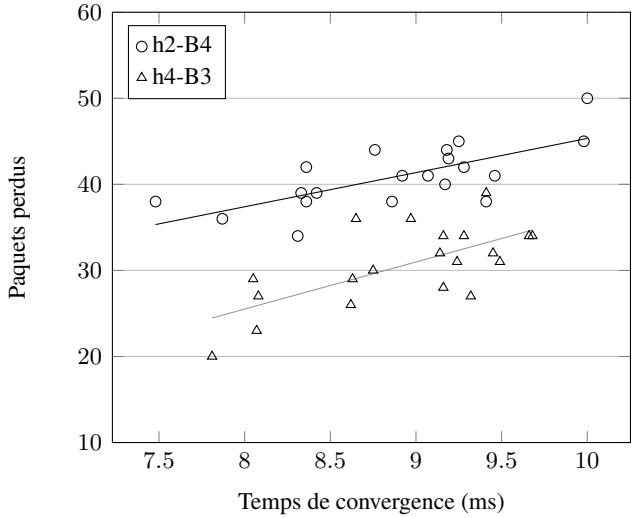
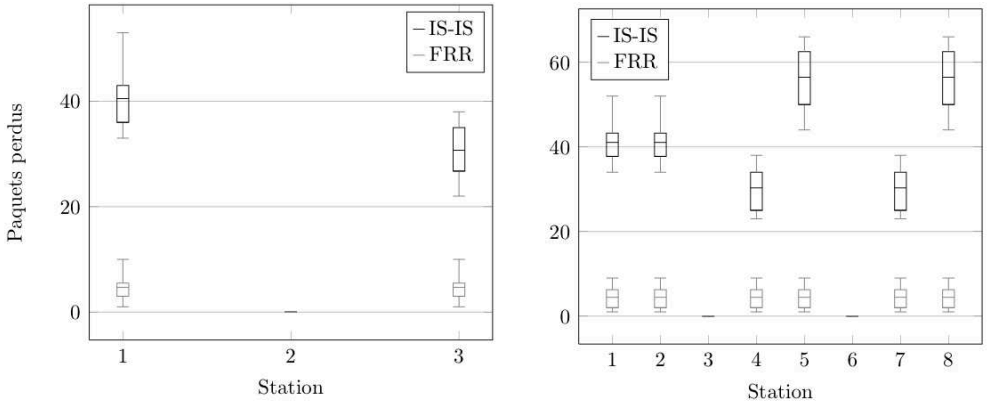
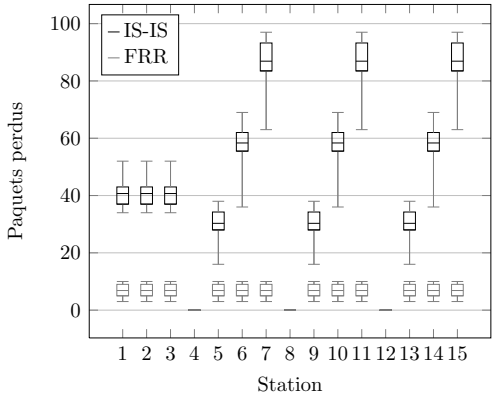


FIGURE 5.18 – Nombre de paquets perdus station 2 et station 4

La figure 5.19 montre le nombre total de paquets perdus pour chaque station. On peut observer la différence entre les réseaux qui utilisent le recouvrement du protocole *IS-IS* et les réseaux qui utilisent le recouvrement *FRR*. Pour toutes les stations dans les trois topologies, le nombre de paquets perdus est plus élevé dans le cas du recouvrement du protocole *IS-IS*, à l'exception des stations qui ne sont pas affectées par la défaillance, pour lesquelles le nombre de paquets perdus est zéro dans les deux cas. La figure 5.20 montre le nombre total de paquets perdus pour toutes les stations; dans la figure 5.20(a) le nombre total de paquets perdus et dans la figure figure 5.20(b) le pourcentage de paquets perdus par rapport au nombre total de paquets attendus. Dans les deux résultats, on observe que le nombre augmente de manière plus prononcée dans le cas du recouvrement réalisée par la convergence du protocole *IS-IS*.

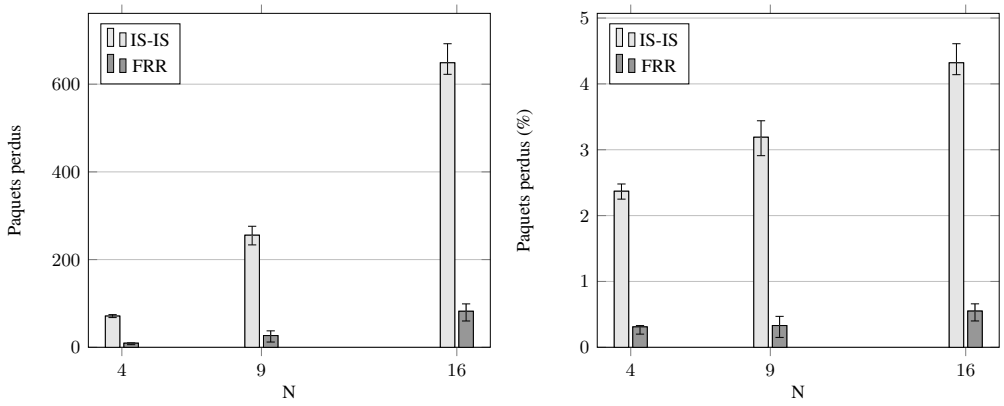


(a) N=4 (b) N=9



(c) N=16

FIGURE 5.19 – Nombre de paquets perdus pour chaque station



(a) Nombre total (b) Pourcentage

FIGURE 5.20 – Nombre total moyen de paquets perdus

5.4 Simulation III - Coût du réseau

La troisième simulation a comme objectif l'analyse du coût d'implémentation du réseau avec une capacité additionnelle. Dans cette simulation, nous mesurons le trafic dans un réseau qui utilise le mécanisme de recouvrement rapide et un autre réseau qui utilise le recouvrement à travers la convergence du protocole *IS-IS*. Cette simulation a comme objectif la comparaison du coût d'implémentation des réseaux protégés par une implémentation du mécanisme de routage rapide présenté dans la section 5.1.

5.4.1 Méthode d'analyse

La figure 5.21 montre la topologie utilisée; le réseau simulé est composé de 6 nœuds, 8 liens et 6 stations. Le protocole *IS-IS* simulé sur la plate-forme *OMNeT++* est le même que celui utilisé dans la simulation précédente. Le mécanisme de recouvrement rapide simulé réalise le détournement du trafic sur la boucle composé par les nœuds 1,2,3 et 4. On a observé le trafic sur chaque lien dans la situation normale du réseau (s=0) et la situation de défaillance sur le lien 2 (s=1). Pour chacune de ces situations le trafic sur les liens a été mesuré, d'abord pour le cas d'un réseau sans le mécanisme de recouvrement et après avec le mécanisme. Pour évaluer le coût du réseau pour différents volumes de trafic générés par les stations, on utilise la mesure du trafic maximal unidirectionnel sur chaque lien. Dans la simulation point-à-point, chaque paire de stations génère un flot de paquets CBR bidirectionnel (demande d). On a ainsi un total de 15 demandes (d=1,...,15). On a réalisé un total de 10 simulations avec un volume de demande (h) variable entre 1000 pps à 10000 pps, soit un incrément de 1000 pps à chaque simulation. Pour le calcul du coût, on a fait la conversion du volume de demande 1000 pps vers une unité de volume (DVU) arbitraire équivalent à 1 Gbps. Dans la simulation du trafic point-à-multipoint, chaque nœud génère une seule demande qui a comme destination tous les autres nœuds. Comme dans le cas du trafic point-à-point, le volume de la demande a été augmenté a chaque simulation.

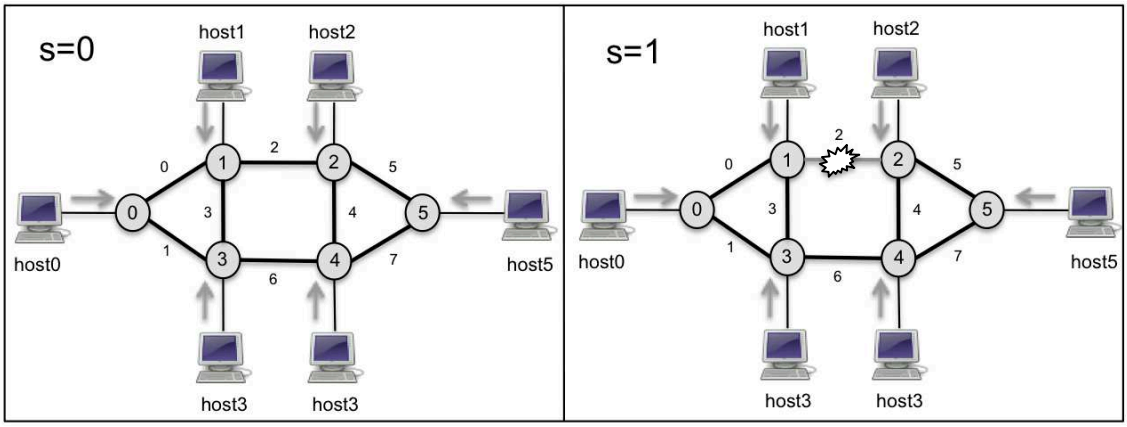


FIGURE 5.21 – Topologie utilisée pour la simulation

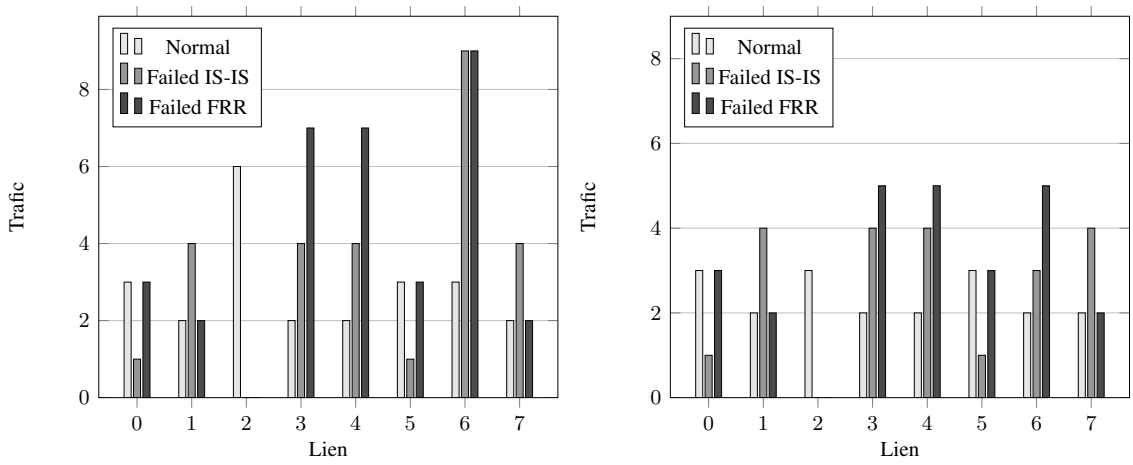
5.4.2 Analyses réalisées

Coût linéaire

Le coût linéaire est obtenu directement à partir de la mesure du volume du trafic sur les liens (équation (4.1)). Dans cette analyse on a pris les résultats obtenus dans chaque simulation. On a analysé les résultats dans les cas du trafic point-à-point et dans le cas du trafic point-à-multipoint.

Coût modulaire

Le coût modulaire a été obtenu à partir du problème de planification avec capacité discrète. Pour le calcul, on a choisi trois options de modules pour les liens: le premier avec capacité de 16 Gbps et coût unitaire de $8.000,00, le deuxième avec capacité de 80 Gbps et coût unitaire de $14.000,00 et le troisième avec capacité de 160 Gbps et coût $24.000,00 (les modules Gigabit Ethernet du commutateur Cisco Catalyst 6500 ont été utilisés comme base pour le coût et la capacité [72]). Le coût total est obtenu par la somme du coût unitaire de tous les modules, considérant que le module utilisé pour chacun des liens sera le moins couteux et qu'il possède la capacité suffisante. Dans cette analyse on a pris les résultats des simulations du trafic point-à-point et point-à-multipoint combinés.



(a) Point-à-point (b) Point-à-multipoint

FIGURE 5.22 – Trafic sur les liens

5.4.3 Résultats

La figure 5.22 montre la proportion de trafic sur chaque lien pour les cas de l'opération normale du réseau (s=0) et pour le cas de défaillance sur le lien 2 (s=1). Avec le recouvrement par la convergence du protocole *IS-IS* le volume total du trafic est plus petit puisque tout le trafic est détourné sur le meilleur chemin. Avec le recouvrement en utilisant le mécanisme *FRR*, seulement le trafic affecté par la défaillance est détourné, mais pourtant le trafic total est plus élevé à cause des retransmissions sur le même lien. Dans la figure on peut observer l'augmentation du trafic sur les liens qui font partie du cycle (liens 3, 4 et 6).

La figure 5.23 montre les résultats pour le coût linéaire; ils sont calculés sur le trafic total du réseau pour chaque volume de demande. La solution *FRR* présente évidement le coût le plus élevé à cause du plus grand volume de trafic. Dans la figure 5.24 on montre les résultats pour le coût modulaire, et dans ce cas on peut observer que le coût de la solution *FRR* ne présente pas forcément le coût plus élevé pour tous les volumes de demande (volumes de demande 1...5 et 10), tant que l'augmentation du trafic n'oblige pas l'acquisition de nouveaux modules. Pourtant, le taux d'occupation du réseau est plus élevé dans le cas où le coût de la solution *FRR* n'est pas supérieur à celui de la solution IS-IS.

Dans ce chapitre nous avons proposé un mécanisme de recouvrement rapide pour un réseau Ethernet contrôlé par un protocole à état de liens. Il existe de nombreux mécanismes pour offrir le recouvrement rapide sur les réseaux IP. Pourtant, les caractéristiques particulières des réseaux Ethernet exigent le développement de solutions pour contourner le blocage des paquets réacheminés et un traitement spécifique pour

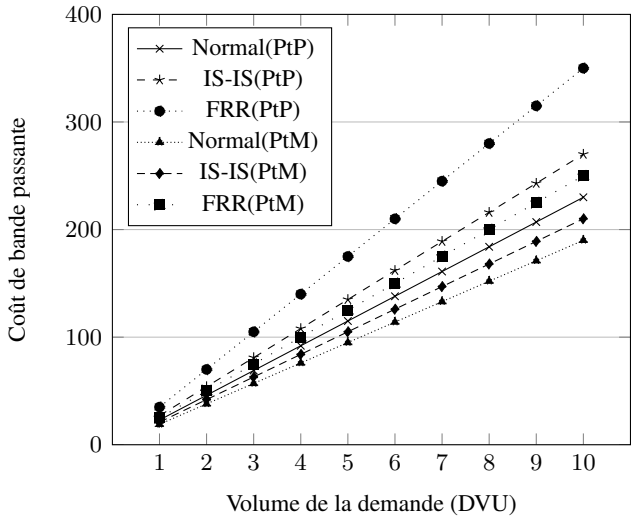


FIGURE 5.23 – Coût lineaire

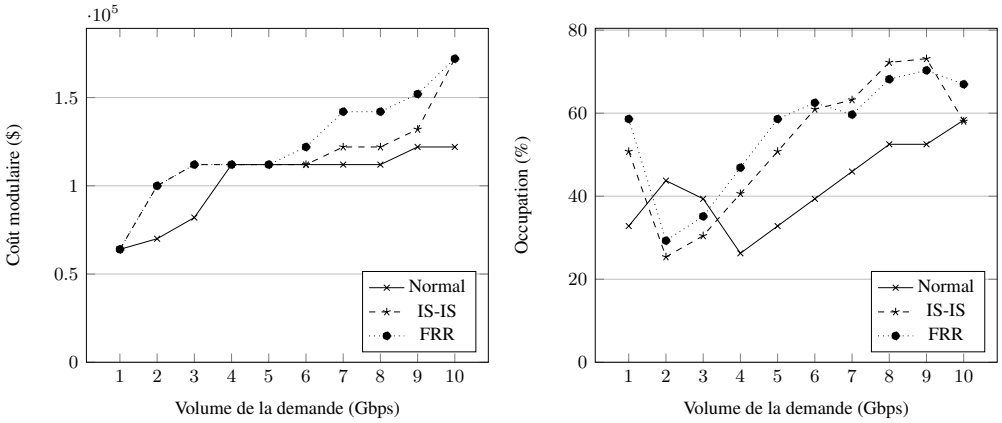


FIGURE 5.24 – Coût modulaire et taux d'occupation

les transmissions de type point-à-multipoint. Notre mécanisme est basé sur les solutions déjà existantes, comme l'utilisation de tunnels et les *p-Cycles*. L'analyse du mécanisme à travers des simulations a permis de montrer que le nombre de paquets perdus pendant le temps de recouvrement du réseau avec l'utilisation du mécanisme est considérablement inférieur au temps de recouvrement du protocole *IS-IS*. De plus, puisque le mécanisme offre un recouvrement local, le temps de recouvrement est indépendant de l'architecture du réseau. L'implémentation d'un mécanisme de robustesse exige l'utilisation d'une capacité additionnelle dans le réseau; cette capacité est plus élevée dans le cas du mécanisme *FRR*. Pour diminuer les retransmissions et, par conséquent, la capacité additionnelle, il faut utiliser un mécanisme amélioré. Toutefois, le mécanisme amélioré augmente la complexité des tables de routage dans les nœuds; comme on a vu dans la section 5.2 cette complexité est proportionnelle au nombre de nœuds et de liens à protéger. Avec l'utilisation des *p-*

cycles, le mécanisme pourra aussi être développé pour la protection de nœuds défectueux, cette approche a été utilisée pour la protection des réseaux WDM dans [73] et [74].

Chapitre 6

Conclusions

Dans ce travail on a présenté un état de l'art des technologies Ethernet de nouvelle génération et une étude sur le temps de recouvrement des réseaux Ethernet contrôlés par un protocole de contrôle du type état de liens. On a fait l'analyse, par voie de simulations, du temps de convergence des protocoles *RSTP* et *IS-IS*. À travers ces simulations on a observé que ce temps dépend de la topologie physique utilisée et qu'il varie selon le nombre de nœuds et la façon dont ces nœuds sont interconnectés. On a aussi proposé l'utilisation d'un mécanisme de recouvrement rapide pour ces réseaux. La simulation d'un tel mécanisme a montré une diminution du temps de recouvrement du trafic sur le réseau, pourtant il exige une augmentation de la capacité additionnelle et dans certains cas, du coût du réseau. Pour diminuer le coût, on peut utiliser une version améliorée du mécanisme, mais la complexité des nœuds est aussi augmentée selon la quantité d'éléments du réseau. L'utilisation du protocole *IS-IS* pour les technologies Ethernet de nouvelle génération est une solution intéressante à cause de sa robustesse, de sa simplicité et de sa facilité d'expansion. Malheureusement, la variation du temps de convergence dans les différentes topologies ne permettent pas l'établissement d'une garantie de recouvrement dans les valeurs attendues par les réseaux de classe opérateur. Afin de profiter de tous les avantages de la technologie Ethernet dans son utilisation pour les réseaux *Carrier*, il faudrait développer des mécanismes capables d'apporter des caractéristiques requises d'un tel réseau. Plusieurs de ces mécanismes sont déjà utilisés, comme c'est le cas des technologies qui utilisent les mécanismes *Q-inQ/MAC-in-MAC*, Ethernet *OAM*, et Synchrounous Ethernet. Toutefois, pour que la technologie Ethernet soit capable d'offrir les services avec la même qualité que celles des réseaux de transport traditionnels, il faut développer des améliorations, notamment dans le cas des mécanismes pour améliorer la robustesse. Le mécanisme proposé dans ce travail permet la réalisation de cette amélioration, mais son implémentation dans un réseau

présente quelques défis, par exemple la résolution du problème de mise à l'échelle causé par la complexité de la construction des tables de routage. Notre mécanisme offre un recouvrement local, donc seuls les nœds qui font partie du cycle sont impliqués dans la récupération; de cette façon il peut être utilisé même dans les topologies maillés plus complexes, sans impact sur le temps de recouvrement. Les technologies *TRILL* et *SPB*, par exemple, n'ont aucun mécanisme de protection rapide. Cela représente un désavantage en termes de fiabilité par rapport à d'autres technologies telles que le *RPR*, le *PBB-TE* et le *MPLS-TP*. Donc, notre mécanisme peut apporter des améliorations aux technologies qui utilisent un protocole à état de liens dans le plan de contrôle. L'amélioration de ce mécanisme est l'objectif de nos travaux futurs.

Bibliographie

[1] R. D. Doverspike, S. J. Phillips, and J. R. Westbrook, "Transport network architectures in an IP world," *IEEE INFOCOM*, 2000.

[2] J. F. Kurose and K. W. Ross, *Computer Networking: A top down approach*, 5th ed. Pearson Education, 2009.

[3] V. Puglia and O. Zadedyurina, "Optical transport networks: From all-optical to digital," *ITU-T Kaleidoscope Academic Conference*, 2009.

[4] A. Mcguire, G. Parsons, and D. Hunter, "Carrier scale Ethernet [guest editorial]," *Communications Magazine, IEEE*, vol. 46, no. 9, pp. 84–86, Sep. 2008.

[5] C. Hoogendoorn, K. Schrodi, M. Huber, C. Winkler, and J. Charzinski, "Towards carrier-grade next generation networks," in *Communication Technology Proceedings, ICCT. International Conference on*, vol. 1, April 2003, pp. 302–305 vol.1.

[6] A. Haider and R. Harris, "Recovery techniques in Next Generation Networks," *Communications Surveys Tutorials, IEEE*, vol. 9, no. 3, pp. 2 –17, quarter 2007.

[7] M. Pourzandi, I. Haddad, C. Levert, M. Zakrzewski, and M. Dagenais, "A new architecture for secure carrier-class clusters," in *Cluster Computing, 2002. Proceedings. 2002 IEEE International Conference on*, 2002, pp. 494–497.

[8] M. Khan, *Building Service-Aware Networks: The Next-Generation WAN/MAN*, ser. Networking Technology. Pearson Education, 2009.

[9] A. Meddeb, "Why ethernet WAN transport," *IEEE Communications Magazine*, Nov. 2005.

[10] A. Reid, P. Willis, I. Hawkins, and C. Bilton, "Carrier Ethernet," *Communications Magazine, IEEE*, vol. 46, no. 9, pp. 96 –103, Sep. 2008.

[11] A. Mcguire, G. Parsons, and D. Hunter, Eds., *IEEE Comunications Magazine: Series on Next-generation Carrier Ethernet transport technologies*, vol. 46, Mar. 2008.

[12] ETNA, "Ethernet transport networks, architectures of networking - final report," http://www.ict-etna.eu/ [Dernier accès le 2 décembre 2011].

[13] C. C. Solutions, "100 gigabit ethernet transport technologies," http://www.celtic-initiative.org/Projects/Celtic-projects/Call4/100GET/Project-default.asp [Dernier accès le 6 octobre 2011].

[14] R. Perlman, "Challenges and opportunities in the design of TRILL: A routed layer 2 technology," in *GLOBECOM Workshops*, Dec. 2009, pp. 1 –6.

[15] D. Allan, P. Ashwood-Smith, N. Bragg, J. Farkas, D. Fedyk, M. Ouellete, M. Seaman, and P. Unbehagen, "Shortest path bridging: Efficient control of larger ethernet networks," *Communications Magazine, IEEE*, vol. 48, no. 10, pp. 128–135, Oct. 2010.

[16] IEEE 802.1ad, "Virtual bridged Local Area Networks amendment 4: Provider bridges," Dec. 2005.

[17] IEEE 802.1ah, "Virtual bridged Local Area Networks amendment 7: Provider backbone bridges," Jun. 2008.

[18] IEEE 802.1ag, "Virtual bridged Local Area Networks amendment 5: Connectivity fault management," Sep. 2007.

[19] ITU-T Y.1731, "OAM functions and mechanisms for ethernet based networks," May 2006.

[20] M. Huynh, S. Goose, P. Mohapatra, and R. Liao, "RRR: Rapid ring recovery sub-millisecond decentralized recovery for ethernet ring," *Computers, IEEE Transactions on*, vol. PP, no. 99, p. 1, 2010.

[21] J. Farkas, C. Antal, L. Westberg, A. Paradisi, T. R. Tronco, and V. G. Oliveira, "Fast failure handling in Ethernet network," *Proc. of IEEE ICC'06*, 2006.

[22] D. Jin, Y. Li, W. Chen, L. Su, and L. Zeng, "Ethernet ultra-fast switching: a tree-based local recovery scheme," *Communications, IET*, vol. 4, no. 4, pp. 410–418, 5 2010.

[23] S. Sharma, K. Gopalan, S. Nanda, and T. Chiueh, "Viking: a multi-spanning-tree ethernet architecture for metropolitan area and cluster networks," in *INFOCOM 2004. Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 4, Mar. 2004, pp. 2283–2294 vol.4.

[24] IEEE 802.1D, "Media access control (MAC) bridges," Jun 2004.

[25] IEEE 802.1Q, "Virtual bridged Local Area Networks," May 2006.

[26] J. Sommer, S. Gunreben, F. Feller, M. Kohn, A. Mifdaoui, D. Sass, and J. Scharf, "Ethernet; a survey on its fields of application," *Communications Surveys Tutorials, IEEE*, vol. 12, no. 2, pp. 263–284, Second 2010.

[27] MEF, "Ethernet services definitions - phase 2," Apr. 2008.

[28] D. Allan, P. Ashwood-Smith, N. Bragg, and D. Fedyk, "Provider Link State Bridging," *Comm. Mag.*, vol. 46, no. 9, pp. 110–117, Sep. 2008.

[29] J.-d. Ryoo, H. Long, Y. Yang, M. Holness, Z. Ahmad, and J. Rhee, "Ethernet ring protection for carrier ethernet networks," *Comm. Mag.*, vol. 46, no. 9, pp. 136–143, Sep. 2008.

[30] R. Vaishampayan, A. Gumaste, S. Rana, and N. Ghani, "Application driven comparison of T-MPLS/MPLS-TP and PBB-TE - driver choices for carrier ethernet," *Proceedings of the 28th IEEE international conference on Computer Communications Workshops*, pp. 79–84, 2009.

[31] B. Niven-Jenkins, D. Brungard, M. Betts, N. Sprecher, and S. Ueno, "Requirements of an MPLS Transport Profile," RFC 5654 (Proposed Standard), Internet Engineering Task Force, Sep. 2009. [Online]. Available: http://www.ietf.org/rfc/rfc5654.txt.

[32] IEEE 802.1qay, "Virtual bridged Local Area Networks - amendment 10: Provider backbone bridge traffic engineering," Aug. 2009.

[33] E. Sfeir, S. Pasqualini, T. Schwabe, and A. Iselt, "Performance evaluation of Ethernet resilience mechanisms," in *High Performance Switching and Routing, 2005. HPSR. 2005 Workshop on*, May 2005, pp. 356–360.

[34] E. Khaled, C. A. L., and N. T. S. Eugene, "On count-to-infinity induced forwarding loops Ethernet networks," in *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, April 2006, pp. 1–13.

[35] J. Farkas and Z. Arató, "Performance analysis of Shortest Path Bridging control protocols," in *Proceedings of the 28th IEEE conference on Global telecommunications*, ser. GLOBECOM'09. Piscataway, NJ, USA: IEEE Press, 2009, pp. 4191–4196.

[36] H. Gredler and W. Goralski, *The Complete IS-IS Routing Protocol*. SpringerVerlag, 2004.

[37] J. Touch and R. Perlman, "Transparent Interconnection of Lots of Links (TRILL): Problem and Applicability Statement," RFC 5556 (Informational), Internet Engineering Task Force, May 2009. [Online]. Available: http://www.ietf.org/rfc/rfc5556.txt.

[38] A. Markopoulou, G. Iannaccone, S. Bhattacharyya, C.-N. Chuah, Y. Ganjali, and C. Diot, "Characterization of failures in an operational IP backbone network," *IEEE/ACM Trans. Netw.*, vol. 16, pp. 749–762, Aug. 2008.

[39] J.-P. Vasseur, M. Pickavet, and P. Demeester, *Network Recovery: Protection and Restoration of Optical, SONET-SDH, IP, and MPLS*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2004.

[40] G. Iannaccone, C.-N. Chuah, S. Bhattacharyya, and C. Diot, "Feasibility of IP restoration in a tier 1 backbone," *IEEE Network*, vol. 18, no. 2, pp. 13–19, 2004.

[41] P. Anelli and M. Soto, "Evaluation of the APS protocol for SDH rings reconfiguration," *IEEE Transactions on Communications*, vol. 47, pp. 1386–1393, 1999.

[42] M. Gjoka, V. Ram, and X. Yang, "Evaluation of IP fast reroute proposals," in *IEEE COMSWARE*, 2007.

[43] P. Francois, "Achieving sub-second IGP convergence in large IP networks," *SIGCOMM Comput. Commun. Rev*, vol. 35, p. 2005, 2005.

[44] P. Francois and O. Bonaventure, "Avoiding transient loops during IGP convergence in IP networks," 2005.

[45] M. Menth, M. Duelli, R. Martin, and J. Milbrandt, "Resilience analysis of packet-witched communication networks," *IEEE/ACM Trans. Netw.*, vol. 17, pp. 1950–1963, Dec. 2009.

[46] P. Pan, G. Swallow, and A. Atlas, "Fast reroute extensions to RSVP-TE for LSP tunnels," RFC 4090 (Proposed Standard), Internet Engineering Task Force, May 2005. [Online]. Available: http://www.ietf.org/rfc/rfc4090.txt.

[47] Y. Yao, Y. Zhang, C. Lu, Z. Zhang, Y. Zhao, and W. Gu, "An efficient shared-bandwidth reservation strategy for MPLS fast reroute," in *Proceedings of the 2009 First IEEE International Conference on Information Science and Engineering*, ser. ICISE '09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 1644–1647.

[48] A. Myers, T. E. Ng, and H. Zhang, "Rethinking the service model: Scaling Ethernet to a million nodes," *Third Workshop on Hot Topics in networks (HotNets-III)*, Mar. 2004.

[49] M. Marchese, M. Mongelli, and G. Portomauro, "Simple protocol enhancements of Rapid Spanning Tree Protocol over ring topologies," in *GLOBECOM 2010, 2010 IEEE Global Telecommunications Conference*, Dec. 2010, pp. 1–5.

[50] R. Pallos, J. Farkas, I. Moldovan, and C. Lukovszki, "Performance of Rapid Spanning Tree Protocol in access and metro networks," *Access Networks and Workshops. AccessNets '07*, pp. 1–8, 2007.

[51] A. Kern, I. Moldovan, and T. Cinkler, "Bandwidth guarantees for resilient ethernet networks through rstp port cost optimization," in *Access Networks Workshops. AccessNets '07. Second International Conference on*, Aug. 2007, pp. 1–8.

[52] J. Farkas, "Notes on IS-IS network convergence," www.ieee802.org/1/files/public/.../new-farkas-convergence-1110.pdf [Dernier accès le 13 juin 2011].

[53] T. S. E. Andy Myers, "Bridgesim," http://www.cs.cmu.edu/~acm/bridgesim/index.html [Dernier accès le 6 juin 2013].

[54] ETNA, "WP3 simulation package - guidance," http://www.ict-etna.eu/[Dernier accès le 11 août 2011].

[55] A. Medina, A. Lakhina, I. Matta, and J. Byers, "BRITE: Universal topology generation from a user's perspective," Boston University, Tech. Rep., 2001.

[56] L. Florit. Putting 50-ms in perspective. http://www.ethernetacademy.net/Ethernet-Academy-Articles/putting-50-milliseconds-in-perspective [Dernier accès le 16 août 2014].

[57] M. Shand and S. Bryant, "IP fast reroute framework," RFC 5714 (Informational), Internet Engineering Task Force, Jan. 2010. [Online]. Available: http://www.ietf.org/rfc/rfc5714.txt.

[58] A. Atlas and A. Zinin, "Basic specification for IP fast reroute: Loop-free alternates," RFC 5286 (Proposed Standard), Internet Engineering Task Force, Sep. 2008. [Online]. Available: http://www.ietf.org/rfc/rfc5286.txt.

[59] A. Atlas, "U-turn alternates for IP/LDP fast-reroute," Internet Draft, Internet Engineering Task Force, Feb. 2006. [Online]. Available: http://tools.ietf.org/html/draft-atlas-ip-local-protect-uturn-00.

[60] W. Grover and D. Stamatelakis, "Cycle-oriented distributed preconfiguration: ring-like speed with mesh-like capacity for self-planning network restoration," in *IEEE International Conference on Communications. ICC 98. Conference Record.*, vol. 1, Jun. 1998, pp. 537–543 vol.1.

[61] D. Stamatelakis and W. Grover, "IP layer restoration and network planning based on virtual protection cycles," *Selected Areas in Communications, IEEE Journal on*, vol. 18, no. 10, pp. 1938–1949, Oct 2000.

[62] J. Kang and M. Reed, "Bandwidth protection in MPLS networks using p-cycle structure," in *Design of Reliable Communication Networks. (DRCN 2003). Proceedings. Fourth International Workshop on*, Oct. 2003, pp. 356–362.

[63] G. K. Jing Fu, Peter Sjödin, "Loop-free updates of forwarding tables," *IEEE Transactions on Network and Service Management*, vol. 5, no. 1, Mar. 2008.

[64] S. Nelakuditi, Z. Zhong, J. Wang, R. Keralapura, Z. Zhong, and C.-N. Chuah, "Mitigating transient loops through interface-specific forwarding," *Computer Networks*, vol. 52, no. 3, pp. 593–609, Feb. 2008.

[65] K. Elmeleegy, A. L. Cox, and T. S. E. Ng, "Understanding and mitigating the effects of count to infinity in Ethernet networks," *IEEE/ACM Trans. Netw.*, vol. 17, pp. 186–199, Feb. 2009.

[66] M. Seaman, "Exact hop count," White Paper, Dec. 2006.

[67] Y. K. Dalal and R. M. Metcalfe, "Reverse path forwarding of broadcast packets," *Commun. ACM*, vol. 21, pp. 1040–1048, Dec. 1978.

[68] M. Zhang, "Reverse Path Forwarding Check under Multiple Topology TRILL," Working Draft, IETF Secretariat, Internet-Draft draft-zhang-trill-multi-topo-rpfc-00.txt., Aug. 2012.

[69] D. Allan and N. Bragg, *802.1aq Shortest Path Bridging Design and Evolution: The Architect's Perspective*. Wiley, 2012.

[70] M. Pióro and D. Medhi, *Routing, Flow, and Capacity Design in Communication and Computer Networks*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2004.

[71] R. Asthana, Y. Singh, and W. Grover, "p-cycles: An overview," *Communications Surveys Tutorials, IEEE*, vol. 12, no. 1, pp. 97–111, quarter 2010.

[72] Google, "Google shopping," http://www.google.com/search?hl=en&tbm=shop&q=cisco+catalyst+6500+ethernet+module [Dernier accès le 10 avril 2014].

[73] D. A. Schupke, "Automatic protection switching for p-cycles in WDM networks," *Opt. Switch. Netw.*, vol. 2, no. 1, pp. 35–48, May 2005.

[74] J. Doucette, P. Giese, and W. Grover, "Combined node and span protection strategies with node-encircling p-cycles," in *Design of Reliable Communication Networks, 2005. (DRCN 2005). Proceedings.5th International Workshop on*, Oct 2005, p. 9.

Annexe A

Algorithmes

A.1 Algorithme 1

Le but de cet algorithme est de réaliser le détournement des paquets sur le cycle.

```
#Arrivee du paquet F au port p_entrée
#TableTunnel: addr -> port
dst = F.AddresseDestination
src = F.AddresseSource
p_entrée = F.PortEntrée
Si (tunnel[F] == vrai):
     Si (dst == MonAdresse):
                    F = decapsuler[F]
                    dst = F.AddresseDestination
                    p_sortie = TableNormale[dst]
                    envoyer[F,p_sortie]
     Sinon :
                   p_sortie = TableTunnel[dst]
                   envoyer[F,p_sortie]
Sinon:
     Si (RPFtest[src,dst,p_entrée] == ok):
          p_sortie = TableNormale[dst]
          Si (défaillance[p_sortie] == oui):
                   src = MonAdresse
```

```
                dst = AdresseVoisin[p_sortie]
                Ftunnel = encapsuler[F]
                Ftunnel.AdresseSource = src
                Ftunnel.AdresseDestination = dst
                p_sortie = TableTunnel[dst]
                envoyer(Ftunnel,p_sortie)
        Sinon:
                envoyer(F,p_sortie)
    Sinon :
        écarter[F]
```

A.2 Algorithme 2

Le but de cet algorithme est de réaliser le détournement des paquets, sans causer la retransmission sur un même lien.

```
#Arrivee du paquet F au port p_entrée
#TableTunnel_in: addr -> p_entrée
#TableTunnel_out: addr -> port_sortie
dst = F.AddresseDestination
src = F.AddresseSource
p_entrée = F.PortEntrée
Si (Point-à-Point[dst] == vrai):
     Si (tunnel[F] == vrai):
           Fdec = decapsuler[F]
           p_sortie = TableNormale[dst]
                Si (dst == MonAdresse):
                      dst = Fdec.AddresseDestination
                      envoyer[F,p_sortie]
                Sinon (dst != MonAdresse):
                      src_orig = Fdec.AdresseSource
                      dst_orig = Fdec.AdresseDestination
                      Si (TableNormale[dst_orig] == TableTunnel[dst]):
                         p_sortie = TableTunnel[dst]
                         envoyer[F,p_sortie]
                      Sinon:
                         p_sortie = TableNormale[dst_orig]
                         envoyer[Foriginal,p_sortie]
     Sinon:
                Si (RPFtest[src,dst] == vrai):
                      Si (défaillance[p_sortie] == oui)
                          src = MonAdresse
                          dst = AdresseVoisin[p_sortie]
                          Ftunnel = encapsuler[F]
                          Ftunnel.AdresseSource = src
                          Ftunnel.AdresseDestination = dst
                          p_sortie = TableTunnel[dst]
                          envoyer(Ftunnel,p_sortie)
                      Sinon:
```

```
                        p_sortie = TableNormale[dst]
                        envoyer(F,p_sortie)
                Sinon:
                        écarter[F]
Sinon:
     Si (tunnel[F] == vrai):
          p_entréetunnel = TableTunnel.in[src]
          Fdec = decapsuler[F]
          src_orig = Fdec.AdresseSource
          p_sortieorigin = TableNormale[src_orig]
          Si (dst == MonAdresse):
               envoyer[Fdec,p_sortieorigin - p_entréetunnel]
          Sinon:
               p_sortietunnel = TableTunnel.out[dst]
               Si (TableNormale.in[src_orig] == TableTunnel.in[src]):
                   envoyer[F,p_sortietunnel]
                   envoyer[Fdec,p_sortieorigin - p_sortietunnel]
               Sinon:
                   p_entréetunnel = TableTunnel.in[src]
                        Si (dst == MonAdresse):
                            envoyer[Fdec,p_sortieorigin - p_entréetunnel]
                        Sinon:
                            envoyer[Fdec,p_sortieorigin]
                            envoyer[F,p_sortietunnel]
     Sinon:
           Si (RPFtest[src,dst] == vrai):
               p_sortie = TableNormale[dst]
               Si (défaillance[p_sortie] == vrai):
                       src = MonAdresse
                       dst = AdresseVoisin[p_sortie]
                       Ftunnel = encapsuler[F]
                       Ftunnel.AdresseSource = src
                       Ftunnel.AdresseDestination = dst
                       p_sortietunnel = TableTunnel[dst]
                       envoyer(Ftunnel,p_sortietunnel)
                       envoyer(F,p_sortie - p_sortietunnel)
               Sinon:
                       envoyer(F,p_sortie)
           Sinon:
```

```
écarter[F]
```