The Path Computation Element (PCE) for MPLS networks and its application to the Internet of Things (IoT)

Jean-Philippe Vasseur

ParisTech
INSTITUT DES SCIENCES ET TECHNOLOGIES
PARIS INSTITUTE OF TECHNOLOGY

TELECOM ParisTech

2013-ENST-0005

EDITE ED 130

Doctorat ParisTech

T H È S E

pour obtenir le grade de docteur délivré par

Télécom ParisTech

Spécialité “ Informatique & Réseaux ”

présentée et soutenue publiquement par

Jean-Philippe VASSEUR

le 18 janvier 2013

L’architecture Path Computation Element (PCE) pour les réseaux MPLS et son application à L’internet des objets

Directeur de thèse : **Maurice GAGNAIRE**

Jury

M. Marco Ajmone MARSAN, Professeur, IEEE Fellows, Politecnico di Torino, Italie	Rapporteur
M. Javier ARACIL, Professeur, Universidad Autónoma de Madrid, Espagne	Rapporteur
Mme Jaudelice de OLIVEIRA, Professeur, Drexel University-Philadelphia, USA	Examinateur
M. Gérard MEMMI, Professeur, Telecom ParisTech, France,	Examinateur
M. Jean-Louis LEROUX, Orange Labs.	Examinateur
M. Martin VIGOUREUX, Alcatel-Lucent	Examinateur
M. Maurice GAGNAIRE, Professeur, Telecom ParisTech, France	Directeur de thèse
M. Stefano PREVIDI, CISCO, Italie	Invité

T H È S E

Télécom ParisTech
école de l’Institut Mines Télécom – membre de ParisTech
46, rue Barrault – 75634 Paris Cedex 13 – Tél. + 33 (0)1 45 81 77 77 – www.telecom-paristech.fr

Acknowledgments

I would first like to warmly thank Professor Maurice Gagnaire for his supervision, guidance and support; throughout the years of collaboration, he has been a source of inspiration, as well as a reference in terms of scientific research.

I am also extremely grateful to Gerard Memmi, head of the Computer Science and Networks Department of Telecom ParisTech who kindly welcomed this research in his department.

Of course, I would also like to express my sincere thanks to both Professor Ajmone Marsan from the Politecnico di Torino-Italy and Professor Javier Aracil from the Universidad Autonoma de Madrid-Spain for having reviewed this thesis in detail, and to the committee members: Professor Jaudelice De Oliveira from the university of Drexel (USA), Jean-Louis Le Roux from Orange (France) with whom I closely collaborated over the past decade on a number of the topics addressed in this thesis, Martin Vigoureux from Alcatel (France) and Stefano Previdi, Cisco distinguished Engineer (Italy) for over 15 years of very close technical collaboration on a vast number of technical topics.

Content

Part 1. Background material

1 State of the art of MPLS Traffic Engineering and Fast Reroute **42**

1.1 Traffic Engineering LSP Attributes 42

1.2 The Traffic Engineering DataBase (TED) and IGP routing extensions 44

1.3 Signaling of a TE LSP 45

1.4 Reoptimization of the TE LSP 46

1.5 MPLS Traffic Engineering Fast Reroute (TE FRR) 46

1.6 Unsolved MPLS Traffic Engineering technical challenges 49

1.6.1 Computation of Inter-domain TE LSP 49

1.6.2 Bandwidth sharing between FRR tunnels protecting independent resources 52

1.6.3 Protecting ABR and ASBR nodes with FRR tunnel 52

1.6.4 Global optimization of TE LSP in intra-domain scope 52

1.7 Conclusion 53

2 The PCE Architecture and its protocols **55**

2.1 The Path computation Element (PCE) Architecture 55

2.1.1 Path Computation Element (PCE) 55

2.1.2 Architectural Building Blocks 56

2.1.2.1 TED synchronization 56

2.1.2.2 Request synchronization 56

2.1.2.3 PCE discovery and Load Balancing 56

2.1.2.4 Signaling protocol for PCC-PCE and PCE-PCE communication 57

2.1.2.5 Stateless versus Statefull PCE 58

2.1.2.6 Policy 59

2.2 PCEP: A new signaling protocol for PCC-PCE and PCE-PCE communication 60

2.2.1 Introduction and problem scoping 60

2.2.2 PCEP protocol messages 61
2.2.3 The PCEP Open and Keepalive messages and the Initialization Phase... 62
2.2.4 The PCEP Path Computation Request message (PCReq) 63
2.2.5 The PCEP Path Computation Reply message (PCRep) 67
2.2.6 The PCEP Notification and Error messages 68
2.2.7 The PCEP Finite State Machine 69
2.2.7.1 Set of Variables 70
2.2.7.2 State Description and transitions 70
2.3 The PCE Monitoring extension 73
2.4 Conclusion 75
3 Using a backward recursive algorithm to compute optimum constrained shortest inter-domain TE LSP **76**
3.1 The BRPC algorithm 76
3.2 Path Optimality 79
3.3 Reoptimization of an Inter-Domain TE LSP 80
3.4 Computation of inter-domain TE LSP across AS made of multiple areas 80
3.5 Extension to an arbitrary set of Autonomous Systems (AS) 81
3.6 Concurrent path computations 82
3.7 Encryption of Path-key to protect confidentiality 83
3.8 Performance analysis via simulation of the BRPC multi-PCE collaborative approach for inter-domain TE LSP computation 86
3.8.1 Set of assumptions 88
3.8.2 Performance metrics 89
3.8.3 Performance results for the second topology (SYM-CORE) 90
3.8.3.1 Path Cost 90
3.8.3.2 Signaling delays 92
3.8.3.3 CAC Failures 93
3.8.3.4 Conclusion 93

3.9 Modeling of the PCE request processing an inter-domain multi-PCE collaborative approach 94

3.10 Conclusion 96

4 PCE selection techniques and Stateless PCE 98

4.1 Introduction 98

4.2 Definition and computation algorithm of the PCR (PCE Path Computation Resource) variable 99

4.3 PCE selection algorithm 100

4.4 A bandwidth defragmentation algorithm reducing the blocking probability in a stateless PCE-based path computation architecture 101

4.5 Mathematical modeling 103

4.6 Conclusion and future work 103

5 A Distributed-PCE based technique to compute back-tunnel for Fast Reroute Node protection with bandwidth sharing 104

5.1 Introduction and Problem Statement 104

5.2 The Naïve CSPF-based Path Computation (NCPC) approach 106

5.3 Notion of bandwidth sharing between backup tunnels 107

5.4 The Facility Based Computation Model (FBCM) 109

5.4.1 Signaling backup tunnel with zero bandwidth 109

5.4.2 Bandwidth protection and bandwidth sharing with FBCM 111

5.4.3 Increasing the probability of backup tunnel success placement of FCBM 114

5.4.4 Required bandwidth to protect 115

5.4.5 Backup tunnel split 117

5.4.6 Extensions to multiple backup pools 118

5.4.7 Set of already signaled bypass tunnels 119

5.4.8 Bandwidth protection and QoS scheduling 120

5.5 Conclusion 120

6 Push-based packet and Event Inspecting (PPEI): A new PCE-based architecture for the Internet of Things **123**

6.1 The Internet of Things (IoT) or Low Power and Lossy Networks (LLNs) ...123

6.1.1 Introduction 123

6.1.2 A short historical background 124

6.1.3 IP Smart Object Network characteristics.....124

6.2 A new PCE-based architecture for the IoT.....126

6.2.1 From Smart object to Minimalist Connect Objects architectures 126

6.2.2 Functional blocks and architecture.....128

6.2.3 A Push-based Packet/Event Inspecting (PPEI) PCE-based architecture 131

6.3 Conclusion.....131

7 Routing and Traffic Engineering in PPEI PCE-based architecture..... **132**

7.1.1 Introduction 132

7.1.2 Use of PCE based computation in non MPLS networks.....132

7.1.3 Overview of RPL, the new routing protocol for the Internet of Things 133

7.1.4 Functional description of the DIA-R, a PPEI PCE used for routing in the Internet of Things 137

7.2 Traffic Engineering in PPEI-based networks.....141

7.3 Conclusion.....144

8 Conclusion and Future work..... **146**

8.1 Conclusion.....146

8.2 Perspectives.....147

List of publications 123

Glossary 126

Bibliography 132

Patents 139

List of Figures

Figure 1 *Traffic Load variation on a Service Provider link [Source: traffic monitoring of a link in a Service Provider in Japan]* 31

Figure 2 – *Illustration of The Well-known Fish Problem* 33

Figure 3 – *Solving the Fish Problem using MPLS Traffic engineering Label Switched Paths* 33

Figure 4 – *Auto-bandwidth approach used to dynamically adjust the TE LSP bandwidth according to measured traffic load* 43

Figure 5 – *Simulation results of the auto-bw dynamic TE LSP sizing strategy* 43

Figure 6 – *MPLS Traffic Engineering Fast Reroute Terminology* 47

Figure 7 – *FRR Local Protection and reroute onto a NNHOP backup tunnel upon a node failure* 48

Figure 8 – *Signaling MPLS TE FRR* 49

Figure 9 – *Routing of Inter-AS TE LSP with the per-domain routing approach* 50

Figure 10 – *The Path Computation Element Architecture* 56

Figure 11 – *Statefull versus Stateless PCEs* 59

Figure 12 – *PCEP messages flows* 62

Figure 13 – *Format of the PCEP PCReq and PCRep messages in Reduced Backus Form* 64

Figure 14 – *PCEP Finite State Machine* 69

Figure 15 – *The BRPC algorithm for inter-AS TE LSP path computation using collaborative stateless PCEs* 76

Figure 16 – *BRPC path computation across IGP areas and Autonomous Systems* 80

Figure 17 – *Use of Path-key to preserve confidentiality using the BRPC algorithm to compute shortest constrained inter-domain paths* 84

Figure 18 – *Network topology 1 (MESH-CORE) used for performance metric analysis using an event discrete simulator* 86

Figure 19 – *Network topology 1 (SYM-CORE) used for performance metric analysis using an event discrete simulator* 87

Figure 20 – *Distribution of the average path cost* 90

Figure 21 – *Distribution of the maximum path cost respectively* 91

Figure 24 – *The notion of "bandwidth Protection" during local reroute with FRR* 105

Figure 25 – *Illustration for the notion of bandwidth sharing between backup tunnels protecting independent resources* 108

Figure 26 – *Illustration for the notion of primary and backup pools* 110

Figure 27 - *Set of NNHOP backup tunnels computed by the PCE Node R2* 113

Figure 28 – *Illustration of bandwidth sharing between two NNHOP backup tunnels computed by the PCE Nodes R2 and R7 for two specific backup tunnels* 114

Figure 29 – *Increased probability of backup tunnel success placement with FCBM*...115

Figure 30 – *Bin-packing problem required in case of Bn with n>1 NNHOP backup tunnels protecting a path segment* 117

Figure 31 – *The Packet Delivery Rate of a low-speed IEEE 802.15.4 link* 126

Figure 32 – *Large-scale system complexity increase as a function of the number of supported functionality (algorithms, protocols)* 127

Figure 33 – *Classic IoT architecture interconnecting IP Smart object network to IP core networks via LBR* 128

Figure 34 – *Representation of a new PCE-based Architecture made of MCOs, DIAs (PPEI-PCEs) and CICs* 129

Figure 35 – *Illustration of the RPL non-storing mode of operation* 136

Figure 36 – *Signaling messaging in PPEI PCE architecture in LLNs to perform routing and traffic engineering* 141

Résumé

L'ingénierie de trafic (nommée ci-après TE pour « Traffic Engineering ») au sein des réseaux de données publics et privés fut sans nul doute l'un des sujets qui a fait preuve d'études très avancées au cours des 30 dernières années, tant au plan académique qu'industriel depuis l'invention même des réseaux de données. L'objectif de l'ingénierie de trafic s'articule autour de deux axes principaux :

1) L'optimisation des ressources réseaux afin de router le trafic dynamiquement au sein du réseau en fonction des ressources disponibles au niveau de la bande passante, prenant en compte l'ensemble des flux présents ainsi que leur variation.
2) La satisfaction de la qualité de service rendue aux applications (par ailleurs appelée QoS (« Quality of Service »), fussent elles de type continu ou élastique. Dans le premier cas, il s'agit principalement d'applications interactives ou unidirectionnelles de type voix ou vidéo. Ce type de trafic réclame des conditions de performance strictes relatives aux délais au sein du réseau (directionnel et bidirectionnel), à la gigue (variation du temps inter-paquets) et se doivent d'être bornés ainsi qu'à un faible taux de perte paquet. Le second type de trafic dit élastique concerne par exemple les transferts de données asynchrones. Ces flux sont généralement moins prédictibles en termes de bande passante requise en comparaison des flux continus de type voix ou vidéo pour lesquels les méthodes de codage permettent une estimation de la bande passante requise.

L'émergence d'applications requérant une qualité de service stricte au sein des réseaux de données, sans compter de nouvelles application industrielles imposant des contraintes de performance extrêmement strictes, se traduisent par une exigence de performance toujours plus accentuée, souvent appelée SLA (de l'anglais « Service Level Agrement »). Ces exigences de performance se traduisent par diverses métriques telles que la bande passante disponible, les bornes relatives au délai de transmission d'un paquet de données au sein du réseau ainsi que la gigue ou encore les taux de pertes paquets. Par ailleurs, il convient de souligner que le taux de disponibilité du réseau lui même est devenu une métrique majeure pour les réseaux fixes ou mobiles au même titre que les métriques de performance mentionnées ci dessus.

De nombreux protocoles, algorithmes et technologies réseaux ont été développés durant ces trente dernières années afin de répondre à de telles contraintes de performance et de disponibilité qui ont permis aux réseaux de données de transporter une large proportion des flux de données (voix, vidéo, données, flux de données industrielles, …). Depuis le début des années 1990, le protocole IP (« Internet Protocol ») s'est avéré être le protocole de choix, ce dernier étant transporté sur réseaux SDH sur fibre optique utilisant la technologie WDM (Wavelength Division Multiplexing) ou directement sur WDM. Par ailleurs, parmi le très large spectre de protocoles IP il convient de souligner le rôle majeur des

réseaux de type MPLS (« Multi-Protocol Label Switching ») et en particulier MPLS VPN (« Virtual Private Network ») et MPLS TE (« Trafic Engineering »).

Depuis la fin des années 1990, le trafic de type « donnée » étant devenu prédominant, les contraintes de couts liées à la technologie SDH offrant un niveau de synchronisation élevé ont amené à son remplacement par des technologies MPLS et notamment FRR (« Fast Reroute »). En effet, MPLS TE FRR a permis par ses évolutions multiples d'offrir des chemins alternatifs en cas de pannes de liens ou de nœuds dans les réseaux avec des temps de convergence similaires à SDH, sinon meilleurs, de l'ordre de quelques dizaines de millisecondes le long de chemins permettant même de garantir la bande passante, l'un des aspects traités en détail dans cette thèse.

Ainsi, MPLS TE fut utilisée non seulement pour fournir une qualité de services appropriée grâce à l'ingénierie de trafic mais aussi des taux de disponibilité du réseau inégalés.

La seule technologie d'ingénierie de trafic disponible avant l'émergence de MPLS TE consistait à adapter dynamiquement la métrique de routage (de type lien) de protocoles tels que OSPF [1] ou ISIS [2], ces derniers utilisant des algorithmes de calculs de plus courts chemins de type Dijkstra [3]. Il est a noter qu'une première tentative fut faite au sein de réseau ARPANET [4] dans les années 1980, consistant à ajuster la métrique de cout de lien en fonction du niveau de congestion de la file d'attente du lien, après ajustement de cette dernière grâce a un filtre passe-bas. Malheureusement cette tentative s'est avérée infructueuse de part l'impossibilité d'éviter des oscillations de routage dans le réseau notamment en présence de fortes variations de trafic. En conséquence, la plupart des protocoles de routage actuels font usage de protocoles de routage à métriques de liens statiques.

Cependant, plusieurs techniques furent développées afin d'effectuer un calcul « optimal » des métriques de liens prenant en compte les ressources réseaux (bande passante) et la matrice de trafic des flux. L'objectif d'optimalité peut se traduire sous des formes diverses telles que la minimisation des taux de charges maximum des liens dans le réseau, le bornage du délai maximum par classe de service dans le réseau pour ne mentionner que quelques uns des objectifs possibles. Ces derniers peuvant également prendre en compte l'état du réseau en présence de pannes de liens ou de nœuds. Ce type de problème étant NP-Complet un grand nombre d'heuristiques furent développées utilisant des techniques d'optimisation locales ou globales.

L'optimisation globale consiste à envisager le trafic (statique) dans la globalité utilisant des techniques inspirées de méta heuristiques de type « Simulated Annealing » (SA) ou « Tabu Search » (TS), toutes deux de type itératif permettant de parvenir à un quasi optimum et dont la convergence dépend fortement de la solution initiale et de la nature des perturbations appliquées à chaque itération. D'autres techniques dites « exactes » comme ILP (« Integer Linear Programming ») peuvent être aussi considérées. Le problème majeur tient cependant à la complexité

qui croit de manière exponentielle avec la taille du réseau et qui rend leur usage inapplicable aux réseaux dont la charge de trafic varie rapidement ou même leur utilisation au calculs de chemins alternatifs en présence de pannes réseau, les temps de calcul étant trop importants.

Sans nul doute, l'utilisation d'ingénierie de trafic IP fut très limitée de part leur manque de flexibilité et granularité (bien souvent le changement d'une métrique de lien peut amener à des modifications de chemins multiples affectant un large nombre de flux). Par ailleurs, les flux de données se sont révélés de plus en plus sporadiques amenant à une prédictibilité réduite face à un besoin de réactivité accru. De plus , l'apparition de techniques de codage adaptifs pour certains flux temps réels (voix, vidéo) a rendu la prédictibilité des ces flux d'autant moins certaine. La figure 1 illustre une variation de trafic typique pour des flux données et voix sur lien operateur.

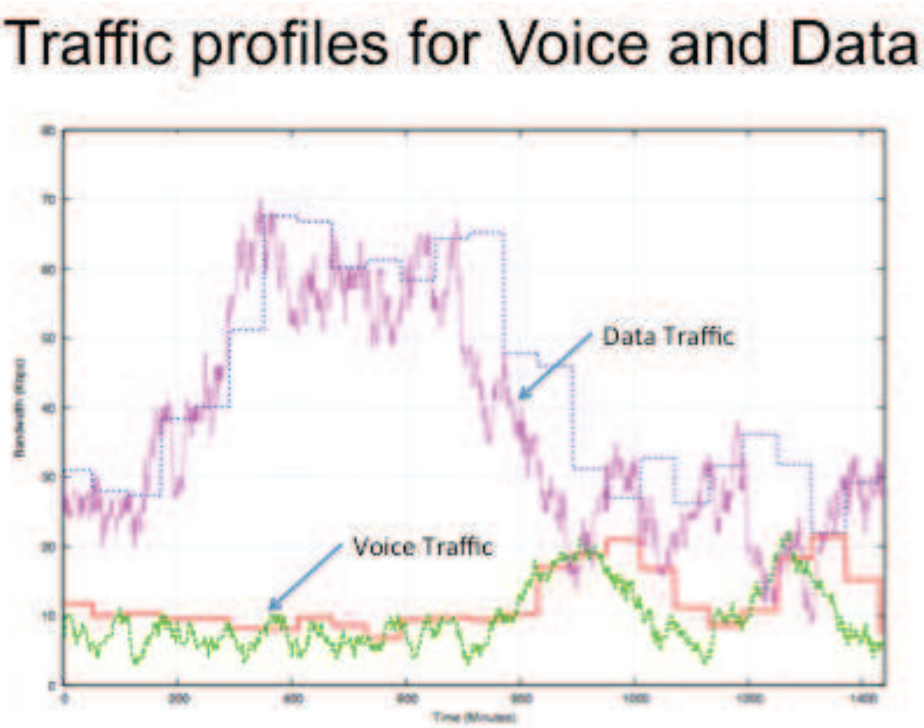


Figure 1 ***Illustration de la charge de trqfic sur un lien Operateur Telecom***

A la lumière de ces observations, il deviendrait nécessaire pour utiliser l'ingénierie de trafic IP de modifier les métriques de liens de manière régulière afin de s'adapter à la demande de trafic fluctuante. Une telle approche qui pourrait s 'opérer grâce au protocole SNMP (« Simple Network Management Protocol ») [5], se révèle non utilisable en pratique : en effet, elle présente de nombreux inconvénients tels que le manque de granularité. Comme précédemment mentionné, la modification d'une métrique a un effet global affectant potentiellement un large nombre de flux ; si une telle contrainte peut être prise en compte par l'algorithme de calcul de métriques modulo un accroissement de complexité de calcul, il convient de souligner que lors

d'une modification de plusieurs métriques, ces dernières ne peuvent s'opérer de manière synchrone, et la mise à jour des bases de données de routage prend elle même quelques dizaines de millisecondes, ce qui amène inéluctablement à la formation de boucles de routage partielles et temporaires mais qui impacte le trafic dans le réseau et donc la qualité de serveur afférente.

Ainsi depuis le début des années 2000, les réseaux IP ont vu l'adoption de nouvelles technologies d'ingénierie et de tolérance de pannes, comme MPLS TE (voir [6] et [7] pour le détail de ces technologies, dont le co-auteur est Jean-Philippe Vasseur, rédacteur de cette thèse). MPLS TE est une technologie riche offrant un vaste spectre de possibilités ; une illustration réductrice mais simple consiste à se référer au problème dit « fish problem », illustré en figure 2. Au sein des réseaux IP, chaque paquet de données est routé de nœud en nœud grâce à l'adresse de destination (et potentiellement d'autres attributs de classe de service par exemple) grâce aux tables de routages calculées par le ou les protocoles de routage ; ces protocoles de routage font usage d'algorithmes de calculs de plus courts chemins distribués sur la base de métriques de liens statiques. Ainsi, comme illustré sur la figue 2, les paquets envoyés par les routeurs R1 et R2 à destination de R8 seront tous routés le long du même chemin dès lors qu'ils auront atteint le routeur R3.

Un équilibrage de charge peut être obtenu grâce à une modification des métriques de liens afin d'obtenir un ensemble de chemins de couts identiques entre les routeurs R3 et R8 permettant ainsi de mieux répartir la charge de trafic (cette technique se nomme ECMP (« Equal Cost Multiple Path »). Malheureusement, même si ECMP peut être utilisée pour le cas trivial exposé en figure 2, son utilisation dans un réseau comprenant plusieurs millions de flux routés dans un réseau interconnectant des centaines de routeurs via plusieurs milliers de liens est très significativement plus complexe et n'offre pas la granularité requise pour la garantie de service évoquée précédemment, à fortiori lorsque les conditions de trafic sont changeantes.

Par ailleurs ECMP ne s'applique que dans le cas de chemins à couts identiques : l'équilibrage de charge le long de chemins asymétriques reste possible mais introduit des problèmes divers tels que l'apparition de boucles partielles, résolus grâce à des techniques d'examens des liens de provenance du trafic qui induisent une complexité et un cout supplémentaire.

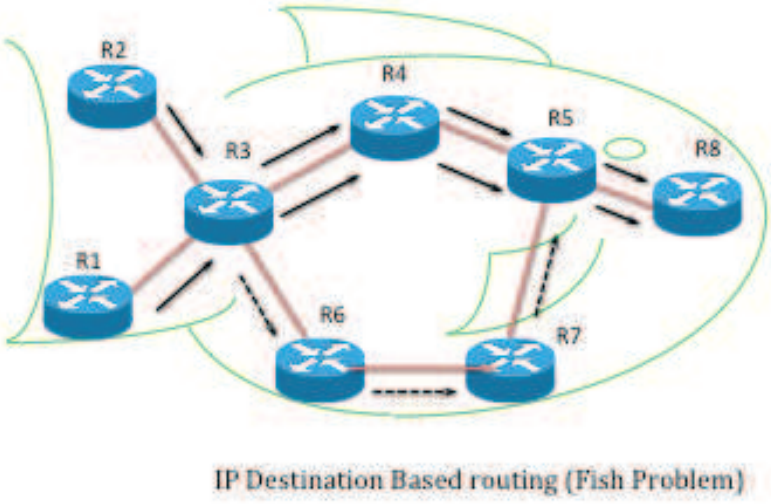


Figure 2 – *Illustration du " Fish Problem"*

MPLS TE utilise une approche radicalement différente et illustrée en Figure 3. Contrairement au routage IP brièvement décrit précédemment, MPLS TE fait usage de tunnels (nommés TE LSP (« Trafic Engineering Label Swtiched Paths ») entre paires de routeurs. Le trafic est ensuite encapsulé à l'intérieur de chaque tunnel dont le chemin est calculé pour répondre aux contraintes de qualité de service évoquées précédemment, et ce grâce à une technique de commutation de labels (appelée « Label Switching »).

Ainsi la décision de routage des flux n'est plus de type « hop-by-hop » et permet d'obtenir une granularité très élevée concordant avec les contraintes d'ingénierie de trafic adaptées au respect de qualités de service. MPLS TE couvre un ensemble de technologies complexes décrites en [6] et [7]. Le chapitre 1 de ce manuscrit fourni un aperçu des protocoles et algorithmes utilisés par MPLS TE.

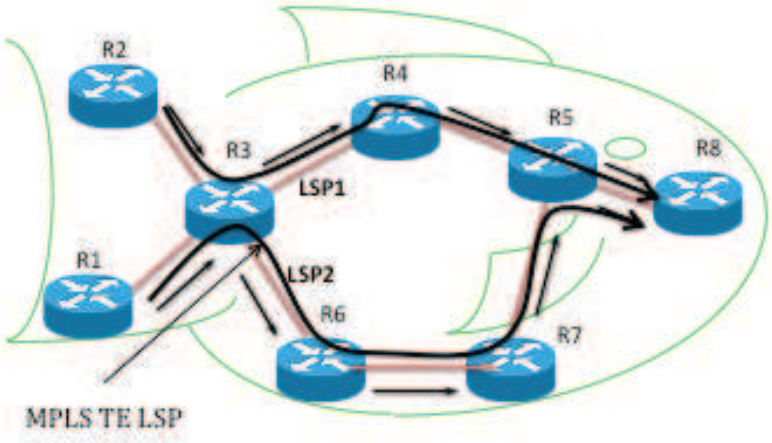


Figure 3 – *Illustration simple de l'utilisation de commutation de label par MPLS TE pour l'ingenierie de traffic*

Il est à noter que bien que l'ingénierie de trafic constitue un élément fondamental permettant une garantie de qualité de service et un taux de disponibilité élevé du réseau, diverses technologies furent développées aux cours des deux dernières décennies qui sont généralement utilisées de concert avec MPLS TE. Plusieurs modèles de qualité de services (QoS) ont été élaborés comme « Diffserv » et « Intserv » qui traitent du marquage de trafic (souvent appelé « trafic coloring »), d'algorithmes de gestion de files d'attente ou encore de techniques de contrôle d'admission (autrement nommées CAC pour « Call Admission Control »).

Contribution de cette thèse

Bien que MPLS TE apparut rapidement comme une technologie extrêmement performante pour l'ingénierie de trafic et le reroutage de trafic en présence de pannes de liens ou nœuds dans le réseau, l'essence de sa complexité réside essentiellement dans le calcul des tunnels utilisés pour le routage des flux ou le reroutage en cas de pannes d'éléments réseau.

L'algorithme le plus simple pour le calcul des (chemins empreintés par les) tunnels se nomme CSPF (« Constrained Shortest Path First »), et se base sur l'algorithme Dijkstra [3] : à chaque tunnel est associé à une série de contraintes qui s'exprime en termes de bande passante, champs de valeurs binaires (appelées affinités), priorités pour ne mentionner que certains d'entre eux utilisés pour retirer des chemins candidats les chemins qui ne répondent pas aux contraintes, et ensuite calculer le plus court chemin prenant en compte une ou plusieurs métriques statiques de liens. CSPF présente l'avantage d 'être peu gourmand en terme de ressources de calculs, ne réclamant aucune synchronisation globale entre les routeurs à l'origine des tunnels utilisés pour router les flux dans le réseau : CSPF est ainsi considéré comme un algorithme distribué.

Cependant CSPF ne peut être utilisé pour résoudre les problèmes suivants :

1) Le manque de synchronisation peut amener à une fragmentation de la bande passante dans le réseau qui présente le désavantage de ne pouvoir trouver un chemin potentiel pour un tunnel qu'au prix d'une réorganisation des tunnels en place.
2) La visibilité des ressources présentes au sein de réseaux distants et traversés par un tunnel inter-domaines n'étant pas accessible par le routeur à l'origine du tunnel, le calcul de ce dernier est effectué de manière sérialisée par chaque routeur d'entrée de domaine. Ce type de contrainte amène un risque de sous optimalité du chemin de bout en bout mais aussi des risques d'échec d'établissement du tunnel, amenant à des reroutages successifs potentiellement couteux en temps.

3) Le calcul de chemin des tunnels de secours utilisés lors de pannes de liens ou de nœuds est rendu particulièrement difficile sauf à fortement sous optimiser les ressources réseaux dédiées au chemin de secours.

L'objectif de cette thèse est de proposer un changement de paradigme consistant à effectuer le calcul des chemins pour les tunnels primaires mais aussi ceux de secours sur un élément de réseau dédié appelé PCE (« Path Computation Elément »). Cette nouvelle architecture également appelée PCE est le résultat de plusieurs années de recherche, implémentation et déploiement de ces technologies à large échelle.

Cette recherche a débuté en 2003 alors que je faisais partie de l'équipe d'ingénierie de Cisco Systems à Boston. Ces années de recherche et design de protocoles et algorithmes ont amené à l'architecture PCE, désormais déployée dans un large nombre de contextes, comme l'ingénierie de trafic, le routage de flux inter-domaines ou encore les réseaux à tolérance de pannes proposant un reroutage rapide en cas de pannes de liens ou de nœuds, avec ou sans garantie de qualité de services.

Il me fut donné l'opportunité de collaborer avec plusieurs universités comme l'université de Drexel (Professeur Jaudelice De Oliveira), l'université de UT Dallas (Professeur Andrea Fumagalli), l'université de Ghent (Professeur Piet Demeester), le MIT (Professor Neil Gershenfel), ou encore l'université de UMass. Plusieurs articles de recherche ont été publiés (IEEE Infocom, IEEE Globecom, IEEE Communication magazines, IEEE networks, Computer networks).

Nombre de technologies, algorithmes et protocoles novateurs ont résulté de ces années de recherches et non moins de 150 brevets ont été déposés. Par ailleurs, il est impératif de souligner le rôle essentiel de la standardisation pour l'émergence et l'adoption de nouvelles technologies qui elles mêmes enrichissent la science et amènent à de nouveaux projets de recherche. A cet effet, j'ai publié environ 30 standards de protocoles ou extensions de protocoles existants auprès de L'IETF (« Internet Engineering Task Force ») durant ces dix dernières années, qui sont relatifs a MPLS et a l'architecture PCE (au cœur de cette thèse).

Il est par ailleurs intéressant de noter que de nouvelles applications du PCE continuent d'émerger et s'accompagnent de nouveaux algorithmes et protocoles standardisés au sein du groupe de travail PCE de l'IETF que je cogère depuis sa création en 2004.

L'architecture PCE a amené à la spécification d 'un nouveau protocole de signalisation appelé PCEP (« Path Computation Elément Protocol ») standardisé par l'IETF dans le document RFC5440 dont je suis éditeur et co-auteur.

Par ailleurs plusieurs algorithmes ont résulté de cette recherche comme l'algorithme BRPC (« Backward Recursive Path Computation ») permettant le calcul optimal de chemins inter-domaines contraints, et encore plusieurs algorithmes distribués pour le calcul de tunnels de secours minimisant la capacité requise pour

les tunnels de secours dans un réseau IP/MPLS, l'ensemble de ces algorithmes et protocoles étant détaillés dans cette thèse.

Par ailleurs il est incontestable que l'Internet des objets qui a commencé à émerger il y a quelques années constitue l'une des révolutions les plus notables des réseaux de données, consistant à connecter via le protocole IP plusieurs milliards d'objets fortement contraints en ressources (mémoire, calcul, bande passante) de type capteurs et actuateurs.

L'interconnexion de ces objets aux réseaux IP publics (Internet) et privés permettra l'émergence de vastes domaines d'applications comme les réseaux électriques intelligents, les réseaux industriels, les véhicules connectés, bâtiments intelligents ou encore les villes intelligentes pour ne mentionner que quelques unes de ces applications émergentes. Il est apparu que l'architecture PCE pouvait, modulo plusieurs adaptations spécifiées dans cette thèse, jouer un rôle central dans l'Internet des objets. La dernière partie de cette thèse est consacrée à l'usage du PCE dans l'Internet des objets au travers de plusieurs adaptations architecturales, protocolaires et algorithmiques.

Structure et résumé du manuscrit

Ce manuscrit est organisé en trois parties.

La première partie fournit les bases nécessaires à la compréhension de l'architecture et des protocoles relatifs à la technologie MPLS TE, élément essentiel à l'architecture, les algorithmes et protocoles spécifiés dans cette thèse. Le premier chapitre introduit également les notions fondamentales sous jacentes à FRR (« Fast Reroute ») pour le reroutage de flux en présence de pannes d'éléments réseaux de types lien ou nœud.

La deuxième partie est consacrée au PCE et comprend les chapitres 2, 3, 4 et 5.

L'architecture PCE repose sur un modèle client-serveur, le client étant un routeur initiant un ou plusieurs tunnels (appelé « PCC : Path Computation Client «) et le serveur étant le PCE (« PCE : Path Computation Elément) en charge du calcul des chemins pour les tunnels. Tout routeur agissant en tant que PCC communique avec un ou plusieurs PCEs via des requêtes qui déterminent les contraintes relatives aux tunnels (bande passante, affinités, priorités, …) ; le PCE effectue les calculs nécessaires et retourne le chemin calculé pour chaque tunnel au PCC. La signalisation entre PCC et PCE (requête, réponse, erreur, redirection, …) s'effectue grâce au protocole PCEP (« Path Computation Elément Protocol).

Le chapitre 2 « The PCE architecture and its protocols » est dédié à l'architecture PCE détaillant les blocs fonctionnels de cette dernière, illustrée Figure 4.

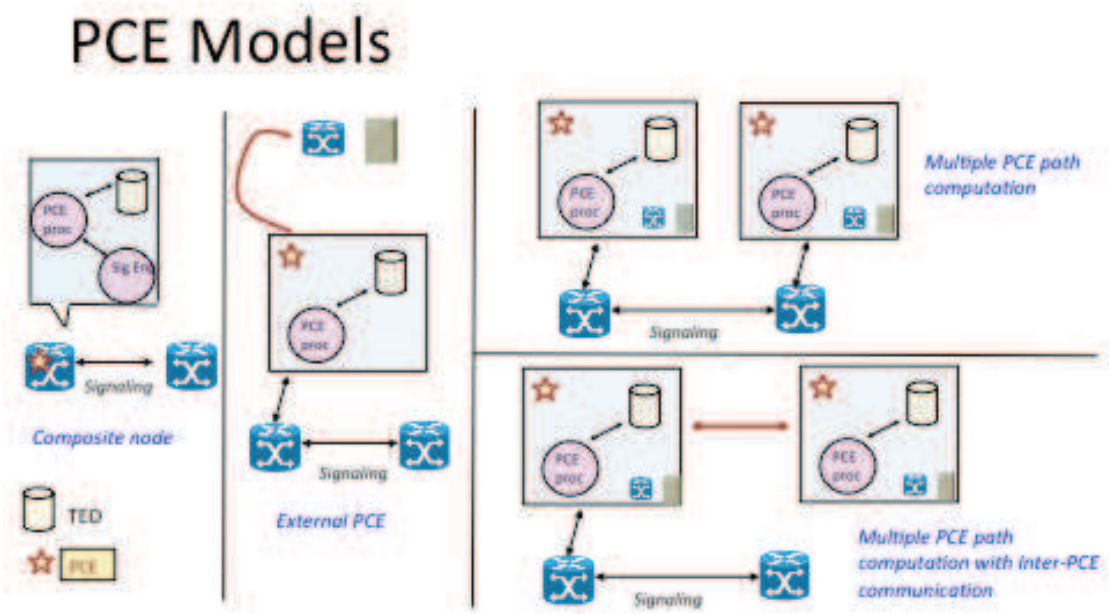


Figure 4 – *L'architecture "Path Computation Element"*

Plusieurs mécanismes de découverte automatique du PCE par les éléments clients (PCC) sont proposées au travers d'extensions de protocoles de routage de type OSPF et ISIS, ainsi que les mécanismes de partage de charge de calculs en présence de plusieurs PCE servant un ensemble de clients.

Il est à noter que deux catégories de PCE sont spécifiées (voir figure 5): avec ou sans mémoire. Dans le premier cas, le PCE mémorise l'ensemble des chemins pour chaque tunnel opérationnel dans le réseau : ainsi le PCE n'effectue pas « simplement » une tache de calcul du tunnel mais répertorie les tunnels pour lesquels le chemin a été calculé et qui sont opérationnels, et ce grâce a une base de données. Bien que plus couteuse en terme de nécessité de synchronisation, ce type de PCE permet la prise en compte de tunnels existants lors d'une requête de calcul de tunnel ; dans l'hypothèse où une requête ne peut être satisfaite il devient possible pour le PCE d'effectuer une réorganisation des tunnels en place afin de satisfaire la nouvelle requête, ce qui induit un cout significatif en terme de contrôle mais permet de diminuer le taux d'échec de calculs notamment au sein de réseaux dont le taux d'utilisation est élevé.

Par ailleurs, le PCE avec mémoire peut procéder à une optimisation globale au regard d'une fonction d'optimisation déterminée, prenant en compte l'ensemble des tunnels existants.

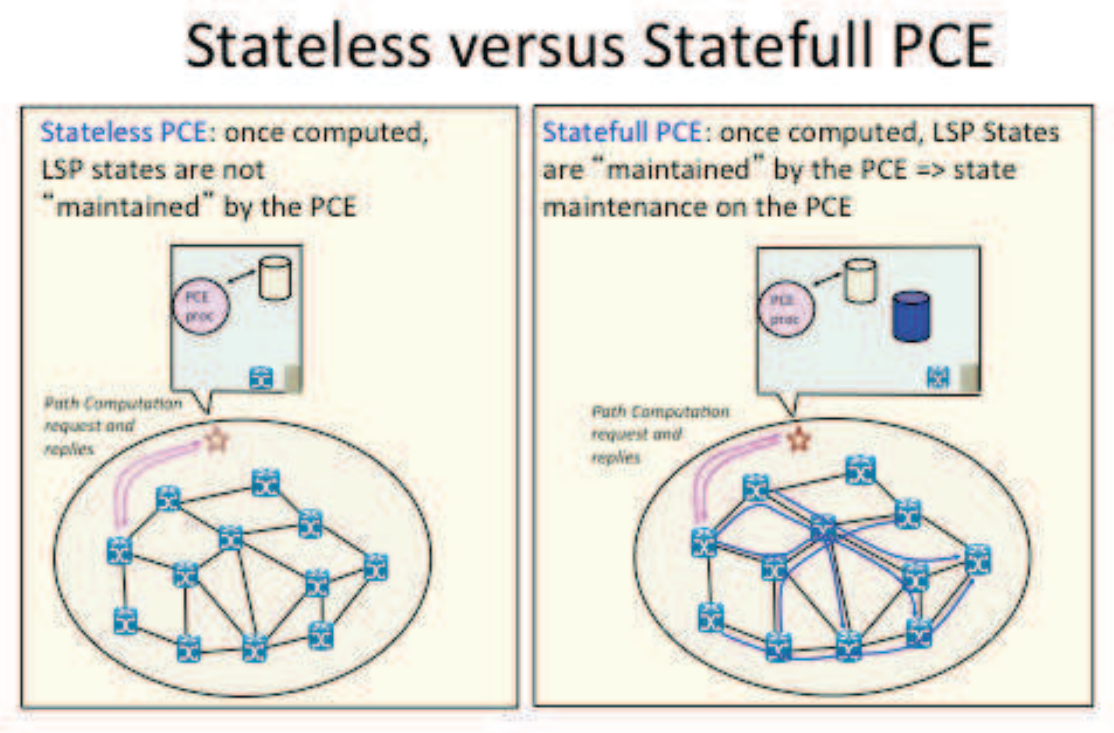


Figure 5 – *Statefull versus Stateless PCEs*

En revanche, les PCE à mémoire restent couteux en ressources et cout de contrôle. Le temps de calcul s'en trouve également augmenté avec un impact sur le temps de réponse. Ces dernières contraintes sont exacerbées en présence de plusieurs PCEs à mémoire, réclamant un degré de synchronisation potentiellement couteux. Il s'avère néanmoins important de signaler leur applicabilité aux problèmes d'optimisation inter-couches (par exemple, WDM et IP).

Par contraste, les PCE sans mémoire traitent chaque requête de manière individuelle, à l'exception de requêtes corrélées. Ainsi, par opposition au cas avec mémoire, l'efficacité en termes d'optimalité de calcul est moindre au bénéfice cependant d'une architecture simplifiée et donc plus efficace en terme de cout de contrôle et donc de temps de réponse par exemple.

Les échanges entre PCC et PCE s'opèrent via le protocole PCEP, basé sur le protocole de transport TCP. Comme brièvement représenté sur la Figure 6, PCEP comprend quatre classes de messages utilisés pour effectuer des requêtes, obtenir le ou les chemins pour un ou plusieurs tunnels contraints, afin d'effectuer des notifications bidirectionnelles (du PCC au PCE ou du PCE au PCC) mais aussi pour les besoins du protocole lui même (gestion des erreurs).

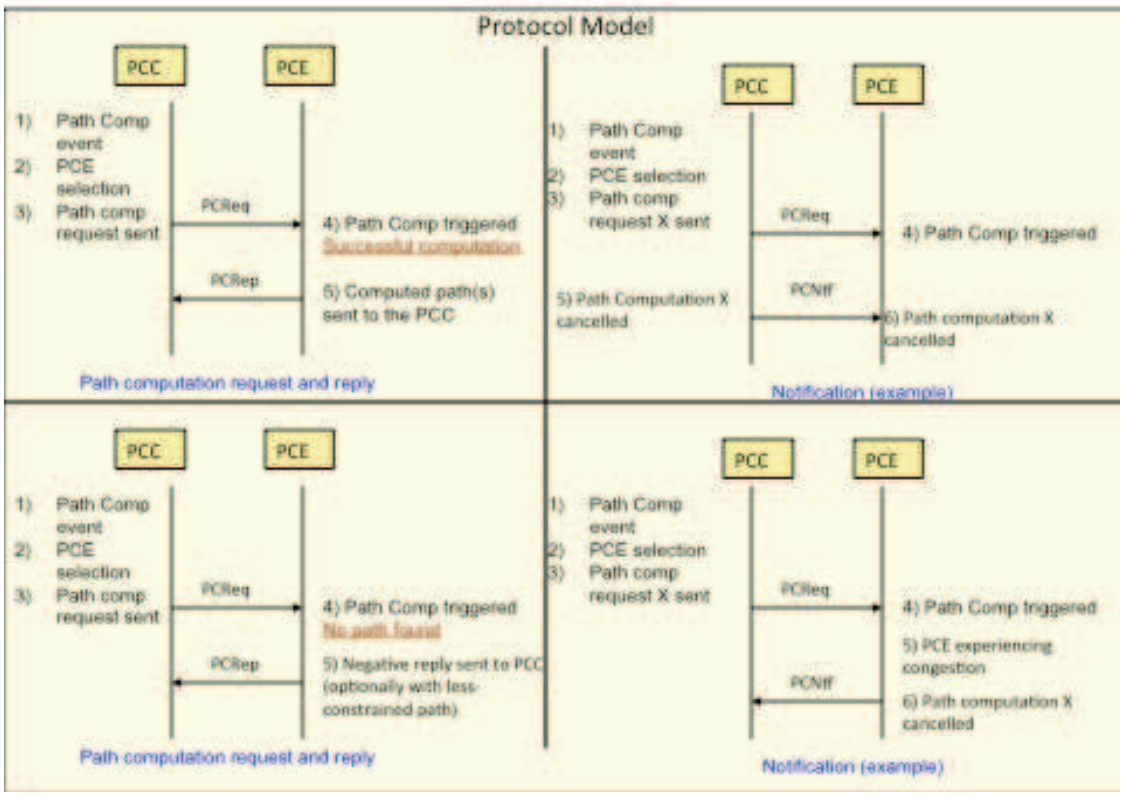


Figure 6 – *PCEP messages flows*

PCEP est un protocole avancé supportant un large éventail de fonctionnalités décrites dans le chapitre 2, et architecturé pour accommoder de multiples extensions.

PCEP supporte la notion de corrélation et synchronisation de requêtes, notamment disponibles pour le calcul de tunnels par des PCEs sans mémoire. Ainsi, un ensemble S de requêtes peut être corrélé pour diverses raisons :

- Optimalité du calcul des chemins,
- Dépendance de chemins : par exemple lorsqu'un PCC requiert le calcul de tunnels n'ayant pas d'éléments réseau (type lien ou nœud) en commun pour du partage de charge afin de minimiser l'impact lié à une panne unique),
- Afin de maximiser la probabilité de trouver un chemin contraint : le PCC peut demander le calcul d'un ensemble de tunnels dont la somme des bande passantes est spécifiée, le nombre de ces derniers bornés, et la bande passante minimale de chacun d'entre eux supérieure à une valeur spécifiée dans la requête).

Il est à noter que les contraintes peuvent être soit impératives (la non satisfaction d'une contrainte impérative menant à un échec de calcul) ou optionnelle.

De plus, dans l'hypothèse d'échec de calculs (une ou plusieurs contraintes ne peut être satisfaites) le PCE peut reporter dans le message la valeur la plus proche pour la contrainte non satisfaite qui aurait mener au succès du calcul.

Par ailleurs, dans le cas (non exclusif) des tunnels inter-domaines, comme spécifié dans le chapitre 3, il peut être nécessaire d'effectuer un calcul partiel du chemin. Dans ce cas, la réponse du PCE contient une clé utilisée pour recouvrer la totalité du chemin et ainsi maintenir la confidentialité des chemins au delà du domaine dans lequel se trouve le PCC.

Le chapitre 3 spécifie un nouvel algorithme appelé BRPC (Backward Recursive Path Computation) illustré en Figure 3 et destiné au calcul optimal des chemins contraints inter-domaines ou un domaine est soit une aire de routage ou un système autonome (AS : « Autonomous System »).

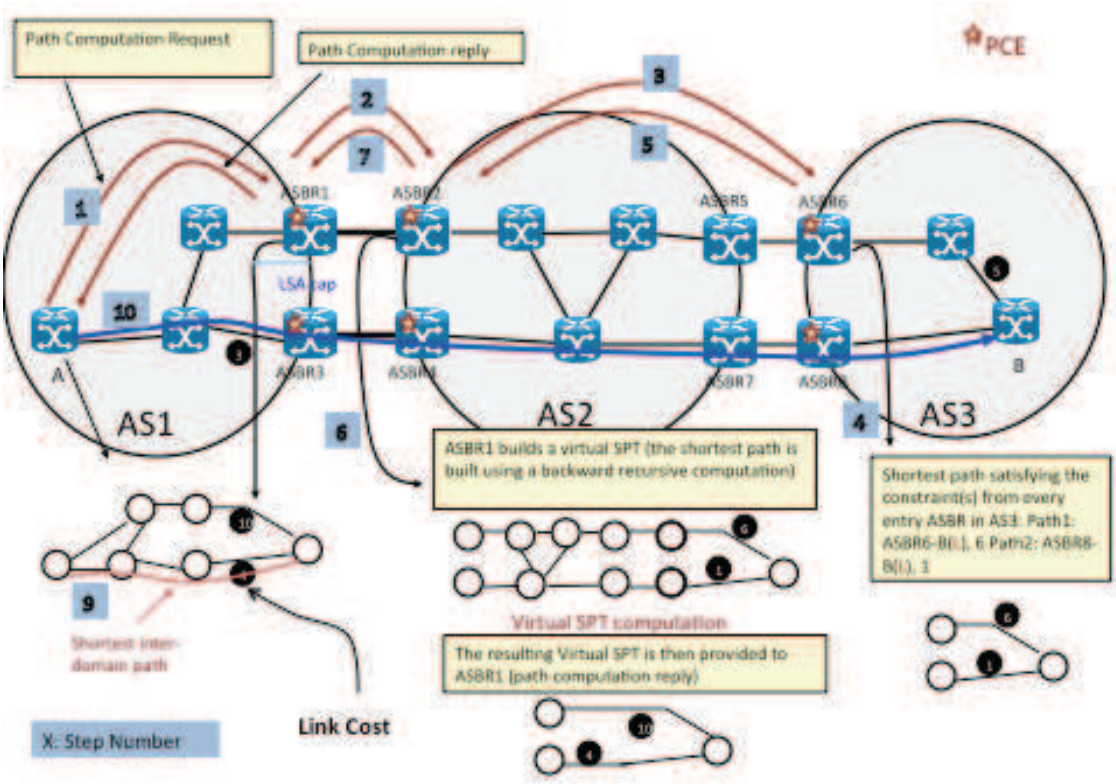


Figure 7 – *Illustration de l'algorithme BRPC pour le calcul de tunnels inter-domaines utlisant plusieurs PCE sans-memoire*

La seule approche auparavant disponible pour le calcul de tunnel inter-domaines consistait à sérialiser le calcul de chaque segment, un segment étant défini comme le chemin suivi par le tunnel au sein d'un même et unique domaine. Ainsi, le routeur initiant le tunnel calculait le plus court chemin contraint jusqu'au routeur d'entrée du prochain domaine (par exemple ASBR 1 Figure 7), puis chaque routeur effectuait

le calcul du chemin au sein de son domaine et ce jusqu'à atteindre la destination finale.

La première contrainte majeure de ce type d'approche est liée au risque de ne pas sélectionner le point de sortie de domaine optimal menant à calculer un tunnel inter-domaines satisfaisant les contraintes mais sous-optimal au regard du meilleur chemin disponible. D'autre part, il était possible qu'un routeur de sortie d'un domaine N sélectionne un routeur d'entrée du domaine N+1 n'ayant pas de chemin possible satisfaisant les contraintes ; dans ce cas, la procédure de signalisation échouait, nécessitant de trouver une combinaison de routeurs d'entrée et sortie de domaines alternative, sachant que cette dernière impliquait des délais potentiellement inacceptables tout en ne pouvant garantir l'optimalité du chemin de bout en bout.

Plusieurs optimisations ont été proposées dans la littérature (comme par exemple la signalisation de ressources agrégées inter-domaines) qui se sont avérées non utilisables en pratique.

Afin de circonscrire les limites des modèles de calcul par domaine, l'algorithme BRPC, basé sur l'utilisation de plusieurs PCE, a été proposé. BRPC (détaillé dans le chapitre 3) est un algorithme distribué impliquant un ensemble de PCE sans-mémoire localisés dans plusieurs domaines et permettant un calcul de chemin contraint optimal.

BRPC implique le calcul sérialisé ou parallèle de segments par plusieurs PCE sans mémoire comme brièvement illustré Figure 8. La requête de calculs est propagée le long d'une série de PCE dynamiquement découverts où débute le calcul des segments entre chaque paire de routeur d'entrée et de sortie de chacun des domaines. La réponse est ensuite propagée en retour où s'effectue le calcul d'un arbre des plus courts chemins dont la source est la destination du tunnel.

Lors du chemin inverse (de la destination vers la source) l'arbre des plus courts chemins est calculé de manière récursive jusqu'à obtention du chemin contraint inter-domaines le plus court. Comme illustré Figure 8, lors de la phase de signalisation, l'utilisation de clés chiffrées fournies lors du calcul des segments permet de recouvrer chaque segment, préservant ainsi la confidentialité de ces derniers.

L'algorithme exact est détaillé au sein du chapitre 3 ainsi que l'évaluation de sa performance. Il est montré grâce à l'utilisation de simulations d' évènements discrets que l'algorithme BRPC affiche une nette supériorité par rapport aux approches de calculs par domaine et ce considérant un large ensemble de métriques comparatives : optimalité du chemin (cout), échec de signalisation et délai de signalisation. Le chapitre inclut également une analyse faisant usage de modèles mathématiques utilisant des modèles de files d'attente de type M/D/1, et qui montre des résultats comparables à nos modèles de simulations à évènements discrets.

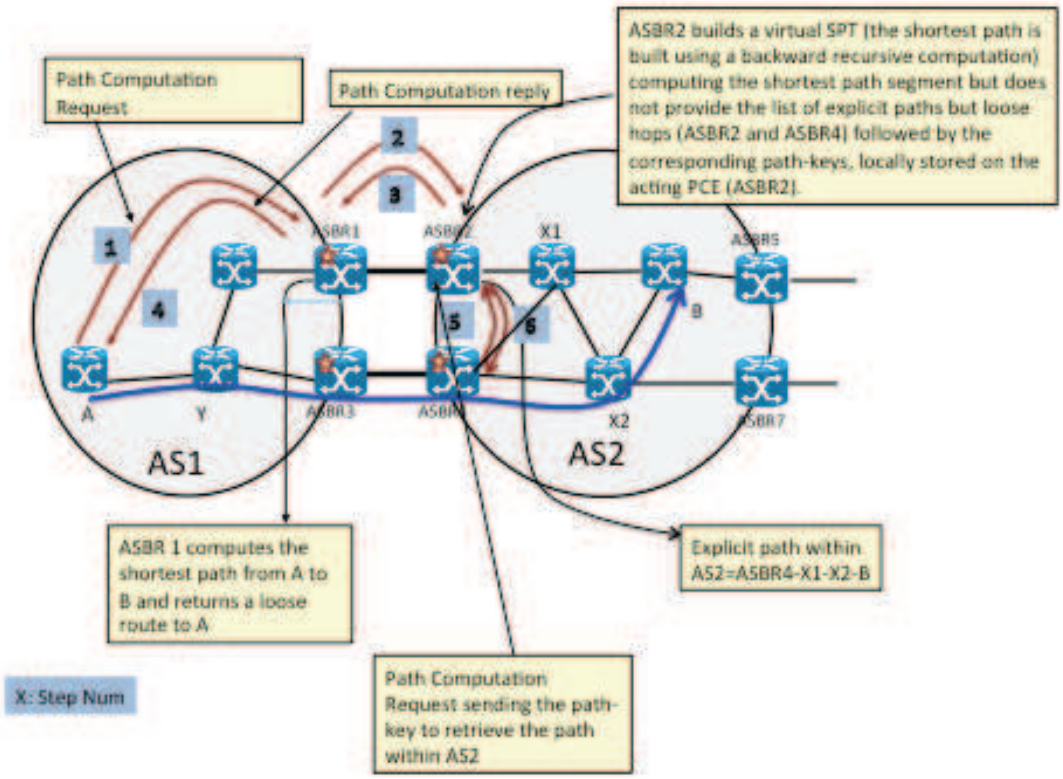


Figure 8 – *Utilisation de cle par l'algorithme BRPC pour la preservation de confidentialite pour les chemins constraints inter-domaines*

Le chapitre 4 (« PCE Selection techniques and Stateless PCE ») est dédié aux algorithmes de sélection de PCE. Ce chapitre traite également d 'une technique de défragmentation de la bande passante afin de réduire la probabilité de blocage au sein de réseaux dont le taux de réservation devient élevé.

Le chapitre 5 présente un nouvel modèle PCE dédié au calcul des tunnels de secours pour FRR (« Fast Reroute ») tout en garantissant une bande passante (et donc une qualité de service équivalente aux flux reroutés en présence de pannes) tout en minimisant la quantité de ressources réseaux dédiées au tunnel de secours.

FRR est une technologie de reroutage local opérant en mode protection : les chemins de secours sont calculés à priori, et dès lors qu'une panne de lien ou de nœud est détectée, les tunnels éligibles au reroutage sont encapsulés dans des tunnels de secours dont la sélection est également effectuée à priori. FRR permet ainsi de rerouter le trafic impacté par la panne en quelques dizaines de millisecondes.

Outre la contrainte sur le temps de reroutage, il s'avéré nécessaire de garantir la non dégradation de performance en reroutant le trafic le long du chemin de secours en offrant des ressources équivalentes par exemple en terme de bande passante. En effet, comme illustré Figure 9, l'approche initiale consistait à rerouter le trafic le

long d'un tunnel de secours n'offrant pas de garantie de service dans l'attente (non déterministe) d'un reroutage du tunnel primaire par le routeur source de ce dernier.

Par ailleurs, il pouvait se produire qu'aucun chemin ne soit disponible pour rerouter le tunnel primaire : le cas échéant, le tunnel primaire restait utilisé et localement rerouté le long du tunnel de secours, en attendant la disparition de la panne.

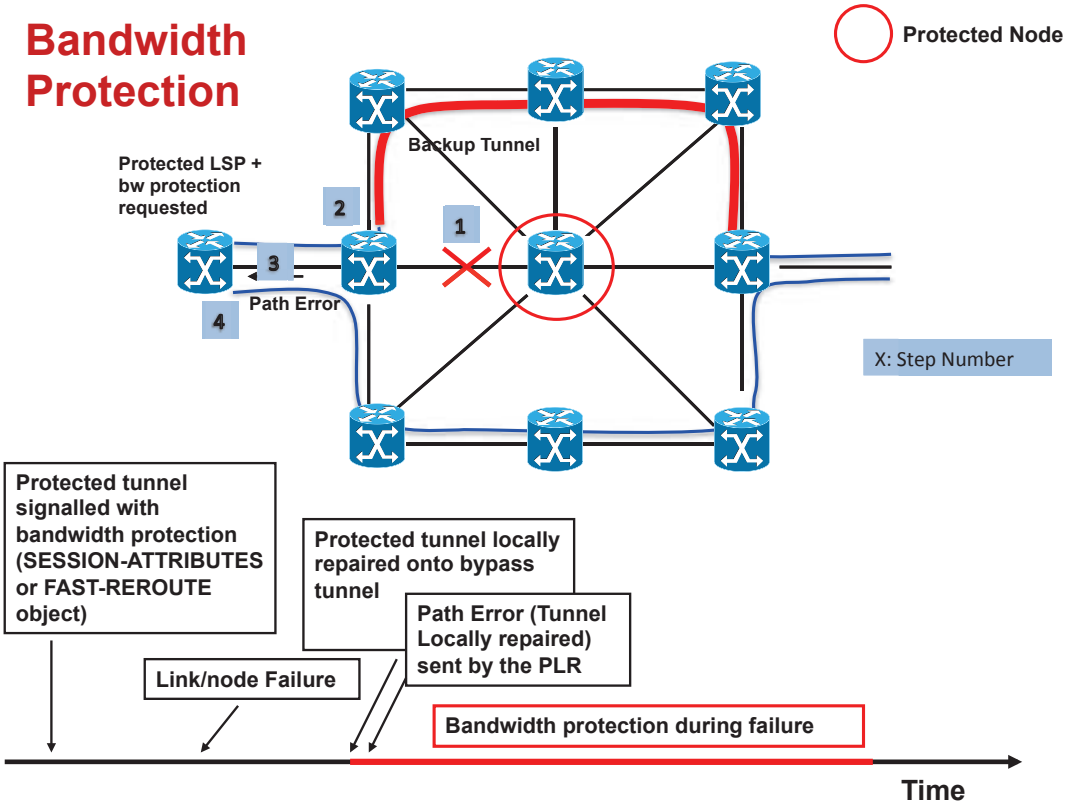


Figure 9 – *Notion de garantie de bande passante après reroutage local*

La première approche naïve permettant de garantir des tunnels de secours offrant une qualité de service non dégradée a tout d'abord consisté à effectuer une signalisation des tunnels de secours ayant pour leur part une bande passante équivalente à la somme des tunnels primaires reroutés sur ces derniers. Bien que fonctionnelle et simple, une telle approche est sans nul doute très couteuse en ressource réseau.

L'objectif du chapitre 5 de ce manuscrit est de proposer une nouvelle approche basée sur des PCE distribués afin de garantir la qualité de service des flux reroutés tout en minimisant la quantité de ressource réseau requise. Cette approche nommée FBCM (« Facility Based Computation Model ») consiste à ce que chaque routeur agisse en tant que PCE pour les calculs des tunnels de secours utilisés par chacun de ses voisins en cas de panne du routeur lui-même.

Ces tunnels de secours sont alors calculés par le routeur lui-même et ne peuvent partager la bande passante : en d'autres termes le PCE calcule une série de tunnels de secours en fonction de la bande passante disponible sur le réseau et dédiée à cet effet.

Le second paradigme clé du modèle FBCM repose sur l'hypothèse de pannes de nœuds non simultanées, hypothèse parfaitement valide. Ainsi, la bande passante dédiée au secours utilisée par les tunnels de secours protégeant des nœuds distincts peut être partagée puisque l'hypothèse est faite que ces derniers ne seront pas simultanément en situation de panne (voir Figure 10).

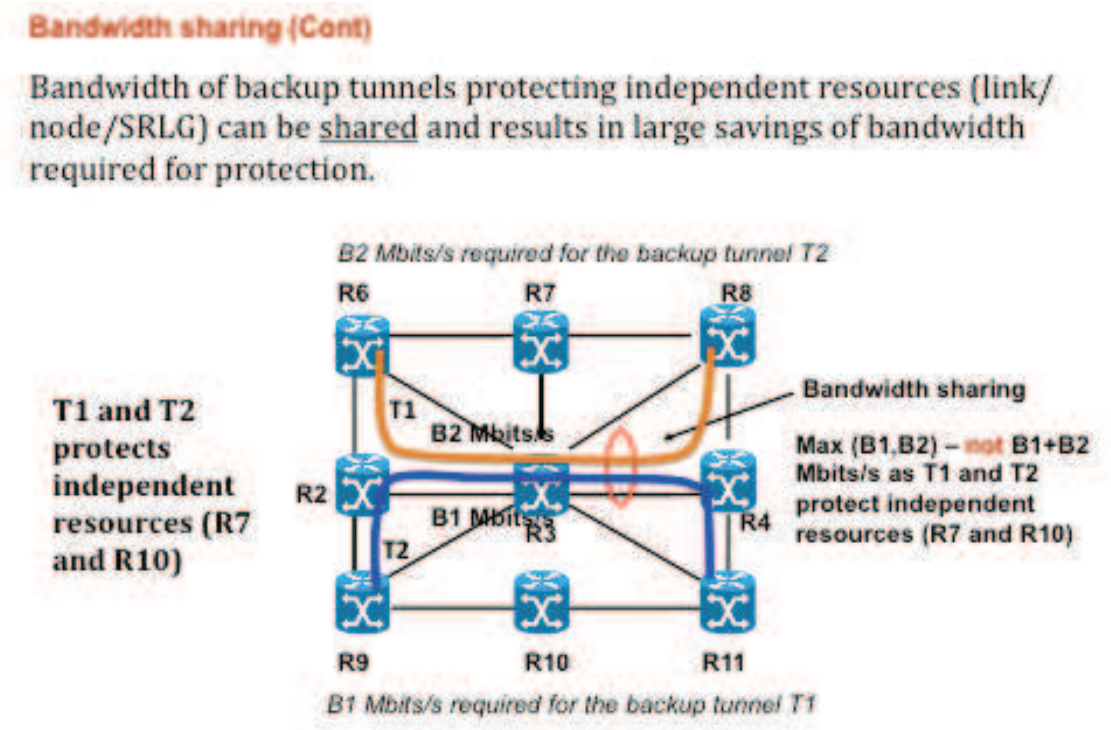


Figure 10 – *Illustration de la notion de partage de la bande passante entre tunnels de secours protegeant des resources independantes*

Comme mentionné précédemment, il devient possible pour chaque routeur d'agir en tant que PCE, utilisant la totalité de la bande passante attribuée aux tunnels de secours pour tous tunnels de secours calculés par ce dernier (illustré Figue 11) ; l'absence de synchronisation entre PCEs (tous les nœuds) se traduit par un partage de la bande de secours entre tunnels protégeant des entités distinctes.

Cette solution permet ainsi, sans qu'il soit nécessaire d'effectuer une modification de la signalisation (les tunnels utilisent une signalisation RSVP standard avec pour valeur de réservation 0), d'obtenir un partage optimal de la bande passante de secours entre tunnels protégeant des entités distinctes. Par contre les tunnels destinés à protéger plusieurs tunnels primaires en cas de panne d'un routeur R

étant calculés par ce même routeur R (agissant en tant que PCE) partagent la bande passante.

Il convient par ailleurs de souligner que le modèle FBCM ne requiert aucune extension de signalisation RSVP, ni même de contrôle d'amission : en effet, d'autres approches ont été proposées dans la littérature qui se fondaient sur la signalisation explicite de la ressource protégée par le tunnel de secours. Ainsi il convenait de modifier le contrôle d'amission afin de partager la bande passante entre tunnels protégeant des entités distinctes. Le modèle FCMP reposant sur une signalisation nulle des tunnels de secours, le contrôle d'admission est non modifié. Une légère modification du protocole de routage est proposée afin de découvrir dynamiquement les ressources dédiées au secours (dénommée « Backup Overlay Network » en Figure 11).

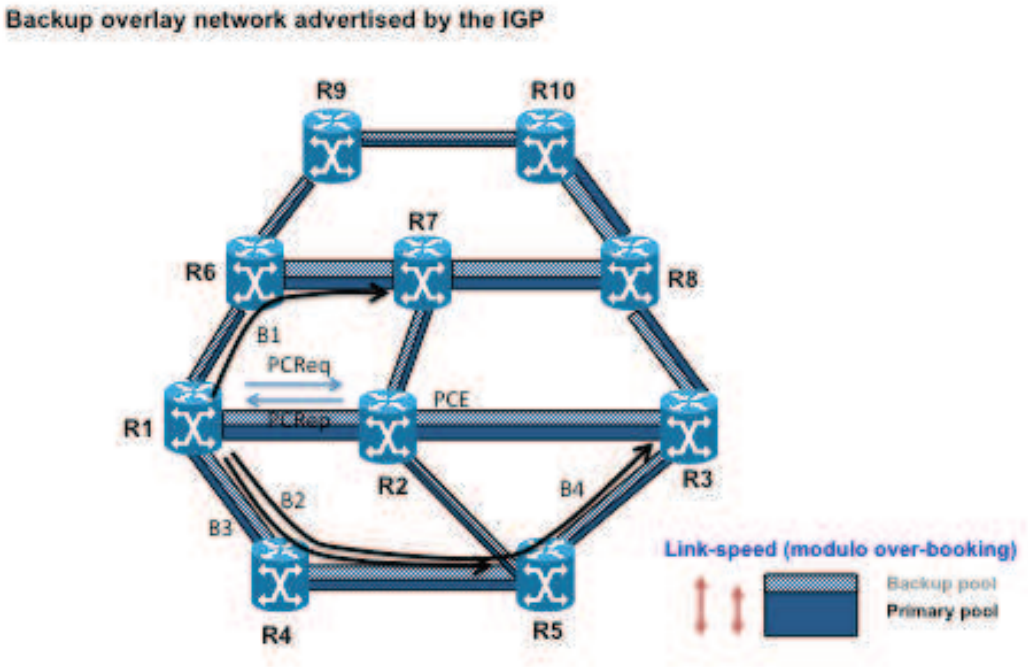


Figure 11 - *Set of NNHOP backup tunnels computed by the PCE Node R2*

Enfin, le modèle FBCM fut étendu au cas des liens bidirectionnels mais aussi des groupes de risque à lien partagé (appelés « SRLG : Shared Risk Link Group), concept particulièrement important au sein de réseaux de données utilisant la technologie WDM.

La troisième et dernière partie de cette thèse (« Applicability of the CE architecture to the Internet of Things ») traite de l'applicabilité du PCE à l'Internet des objets au sein des chapitres 6 et 7, mettant en exergue plusieurs problèmes critiques qui trouvent leur résolution grâce à l'utilisation d'une architecture PCE.

Le chapitre 6 introduit un nouveau modèle de PCE appelé PPEI (« Push-based packet and Event Inspecting ») destiné à résoudre plusieurs problèmes relatifs au routage et à l'ingénierie de trafic au sein de l'Internet des objets (autrement appelé IoT « Internet of Things »).

Il convient de rappeler que l'IoT introduit un vaste ensemble de contraintes liées à la nature des objets connectés et brièvement introduits précédemment : liens à très bas débits, instabilités des liens avec des taux de pertes paquets (souvent imprédictibles et variables), puissance de calcul sur les nœuds fortement limitée, mémoire restreinte ou encore énergie limitée lorsque les objets sont alimentés par batterie (cette partie du réseau est souvent appelée LLN « Low power and Lossy Network »). Ce type de réseau représenté en Figure 10 doit être capable d'opérer à très grande échelle avec plusieurs dizaines de millions voire milliards d'objets.

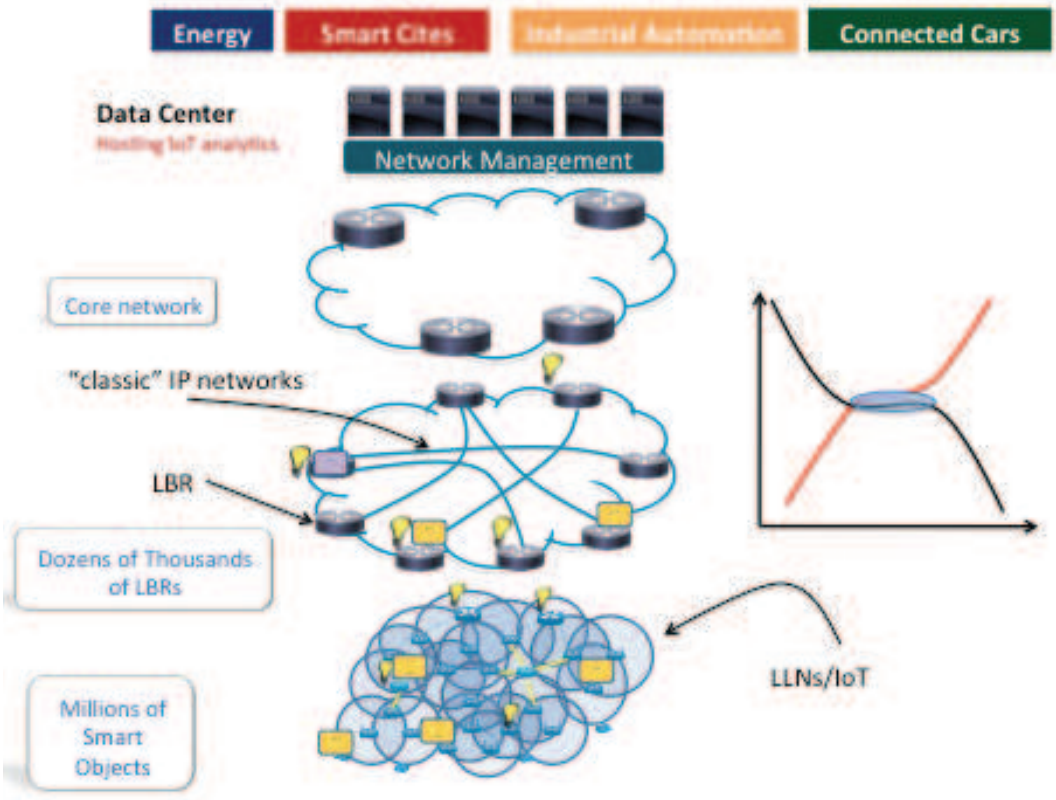


Figure 12 – *Architecture de L'Internet des Objets*

Le chapitre 6 propose un modèle PCE novateur adapté à l'IoT au sein duquel les objets dit « intelligents » deviennent minimalistes (MCO : « Minimalist Connected Object ») reléguant le traitement de taches plus complexes telles que le routage ou l'ingénierie de trafic à des agents équipés d'intelligence distribuée (appelés DIA « Distributed Intelligence Agent ») et agissant en qualité de PCE appelés PPEI-PCE.

D'autres DIA peuvent par ailleurs supporter d'autres fonctions telles que l'activation de modèle de qualité de service prenant en compte dynamiquement les métriques caractérisant la performance du réseau, la corrélation d'alarmes ou encore la fusion de données.

Enfin un troisième type d'élément réseau est défini nommé CIC (« Central Intelligence Controller ») en charge de taches plus complexes nécessitant d'importantes ressources de calcul et mémoire telle que l'optimisation globale, la gestion de règles plus complexes ou encore l'agrégation de données à large échelle. Cette nouvelle architecture PCE dédiée à l'Internet des objets est illustrée en Figure 11.

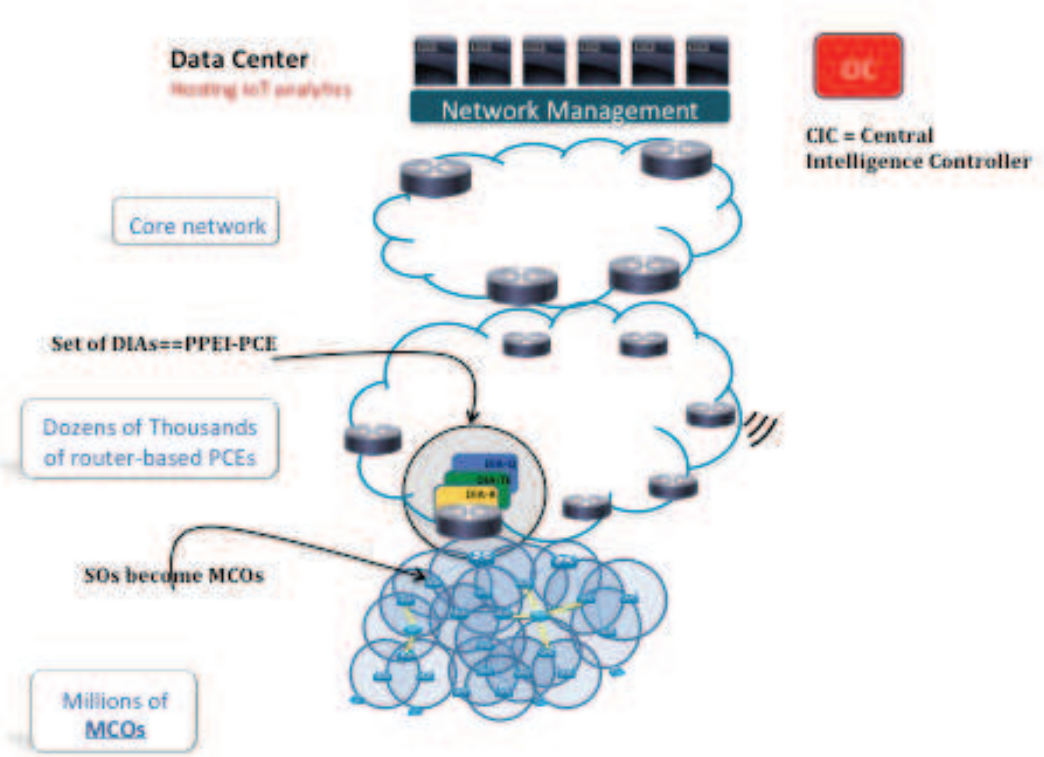


Figure 13 – *Representation of a new PCE-based Architecture made of MCOs, DIAs (PPEI-PCEs) and CICs*

Le dernier chapitre de cette thèse spécifie plusieurs algorithmes basés sur l'architecture précédemment exposée afin de traiter les fonctions de routage et ingénierie de trafic grâce au modèle PPEI-PCE.

La première partie de ce chapitre propose un routage mixte distribué et implémenté sur les objets de type MCO ; ce type de protocole de routage qui peut être inspiré du protocole de routage RPL vise à effectuer un calcul de routes sous-optimal et léger

en termes de ressources de calcul et plan de contrôle. Les chemins obtenus sont alors optimisés de manière incrémentale par le ou les PPEI-PCE selon l'algorithme spécifié dans ce chapitre prenant en compte plusieurs sources d'informations telle que la topologie LLN et évènements divers collectés par le PCE à la lumière des objectifs de qualité de service.

Il est à noter que dans ce cas le PCE ne reçoit pas de requête explicite de calcul de chemin mais intervient de manière proactive (sans attente de requêtes), comme illustré en Figure 12.

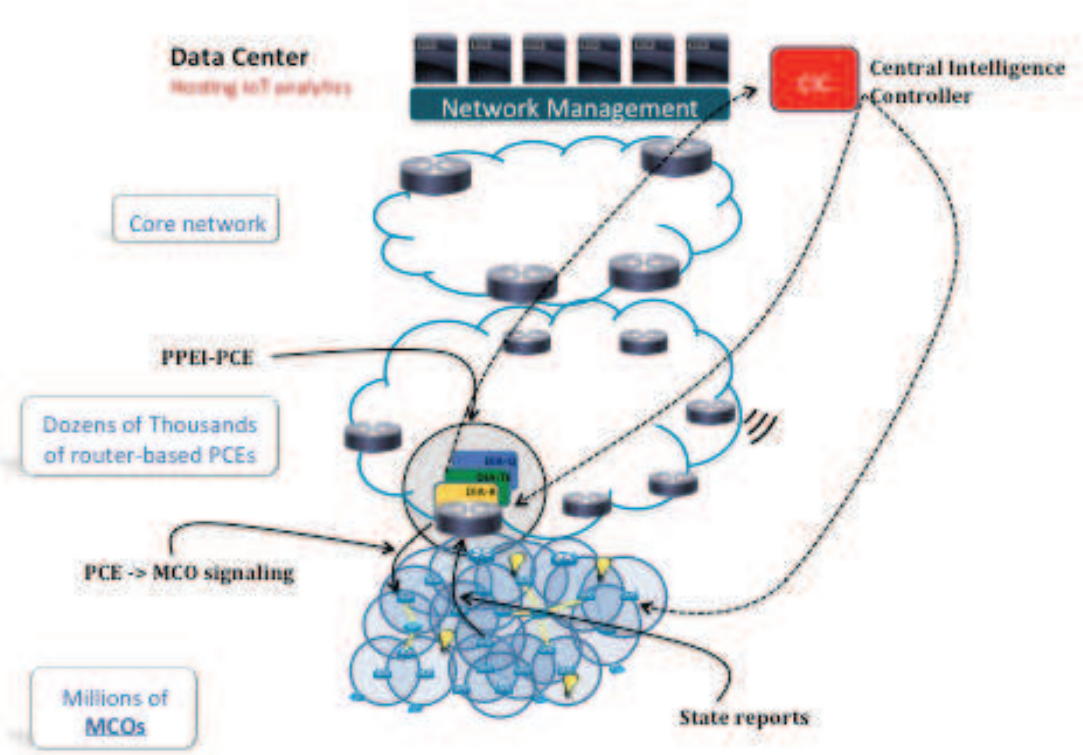


Figure 14 – *Messages de signalization entre MCO et PPEI-PCE*

Le second algorithme spécifié dans ce chapitre est relatif à l'ingénierie de trafic au sein de réseaux IoT (LLN). En effet, ce type de réseaux étant fortement contraint il n 'est pas envisageable de faire usage de MPLS TE telle que actuellement défini. Cet algorithme effectue tout d'abord un calcul de la matrice de trafic au sein du réseau LLN grâce à une technique reposant sur l'inspection de trafic (appelée DPI « Deep Packet Inspection »). Cet algorithme utilise la topologie de routage ainsi que d'autres informations telle que la bande passante dynamique afin de déterminer les liens potentiellement congestionnés en vue de procéder à l'ingénierie de trafic.

Introduction

Context

Traffic Engineering (TE) in public data networks has been one of the most prevalent topics of research and engineering for the last three decades. Two main objectives are dedicated to TE. The first one consists in routing dynamic traffic in order to optimize network resources utilization. The second one aims to satisfy the Quality of Service (QoS) requirements of the various types of connections that may be transported onto the network. Two main classes of traffic are considered: stream and elastic. The former refers mainly to interactive voice/video connections or to unidirectional voice/video diffusion from dedicated servers. This type of traffic typically requires upper bounded round-trip-times (RTT) and inter-packet delay jitters. It also necessitates low average packet loss. The later refers to asynchronous file transfers between remote nodes. This type of traffic mainly requires a minimum guaranteed bandwidth. Stream traffic bit rate is more or less predictable since it is determined by the nature of the coding/decoding technique used at the transmitter/receiver. Elastic traffic bit rate is a priori more fluctuant and unpredictable than stream traffic bit rate.

The support of strict Service Level Agreements (SLAs) has become an obligation. The QoS parameters inherent to these various types of data flows have been specified in terms of RTT, jitter and packet loss. It is worth pointing out that these packet-based networks do carry application with very stringent QoS requirements in support of industrial applications. In addition, high reliability in case of network component failure has become an absolute priority for all public packet-oriented networks, either for fixed of mobile users. Several factors have motivated for the design and development of successive generations of networking technologies, architectures, and protocols.

Between the end-users demanding for enriched services, the equipment vendors, the operators, and the performance of electronics, a form of virtuous loop has favored during these last twenty years the emergence of successive generations of transmission and switching/routing technologies, protocols and network architectures. Since the year 1990, IP traffic encapsulated into SONET/SDH frames over Wavelength Division Multiplexing (WDM) has emerged as the technology of choice for the deployment of multi-service highly reliable public networks. IP protocols consist in a large protocol suite that has been enhanced with the MultiProtocol Label Switching (MPLS) control plane. MPLS provides the ability to support Virtual Private Networks (VPN) and Traffic Engineering (TE).

By the end of the years 1990s, data traffic began to become predominant over voice traffic. In this new context, two main drawbacks of SONET/SDH have been underlined. SONET/SDH assumes a high level of synchronization between network nodes. The achievement of this synchronization is costly. In addition, SONET/SDH that is circuit-oriented, only provides connections with a very coarse granularity

that is ill-suited to most of the IP traffic flows. This was one of the motivations for the support of fast protection and restoration functionalities in the MPLS specifications. The main challenge today satisfied consisted in providing dynamic traffic recovery of MPLS paths upon link and node failures in a few tens of milliseconds, as it was possible with SONET/SDH. MPLS Traffic Engineering (TE) has undoubtedly emerged as the traffic engineering technology of choice in IP/MPLS not only for the sake of traffic engineering but also to provide extremely high network reliability.

Prior to the emergence of MPLS Traffic Engineering (MPLS TE), the only tool available to engineer traffic was the routing protocol. By tuning the link costs, it was thus possible to influence path computation in the network performed by distributed routing protocols such as OSPF [1] or ISIS [2]. Both OSPF and ISIS use the well-know Dijkstra [3] algorithm with minor adaptations. They make use of the network topology and the associated link costs to compute shortest paths in the network. An attempt was made in the ARPANET [4] in the early 80' to make the link metrics dynamics so as to reflect the level of link congestion. For that purpose, the average link load was controlled in monitoring the queuing scheduler by means of a low-pass filter. It then became possible to dynamically adapt the routing behavior to the offered load. Unfortunately despite a number of attempts, it turned out to be too challenging to stabilize the network behavior, especially in Wide Area Networks (WAN). By increasing link costs according to link load and other parameters such as the averaged queuing delays, the networks quickly exhibited routing oscillations leading to lack of stability. Thus today's networks strictly make use of static routing link metrics.

Still, this does not preclude from performing traffic engineering in the network thanks to off-line tools optimizing the link metrics. Such tools require a fine-grained knowledge of the traffic matrix and offered load, network topology and available resources so as to compute the "optimum" link costs. From this knowledge, it is possible to optimize network resources utilization according to an objective function. Various objective functions can be adopted such as minimizing the maximum load, bounding the maximum delays on a per Class of Service (CoS) basis according to pre-defined SLAs to name a few. These optimization techniques are activated during normal operation and failures. Such problems are known to be NP-Complete and a number of heuristics have been worked out over the past two decades to engineer IP traffic by links costs tuning. In this matter, two classes of optimization must be distinguished: global and local. The global optimization techniques consider the whole set of traffic demands and tries to route them onto the network under the assumption of static traffic. Most of the proposed heuristics for the static context are inspired from the Simulated Annealing (SA) or the Tabu Search (TS) meta-heuristics. The SA and TS meta-heuristics are two iterative techniques that can drive to near optimal solutions. Meanwhile, their speed of convergence strongly depends on the quality of the considered initial solution and the nature of the applied perturbation at each iteration. Exact techniques based on an Integer Linear Programming (ILP) formulation of the objective function are also

possible. The problem is that the number of variables to take into consideration in such formulations grows exponentially with network size. This imposes unacceptable computation delays for dynamic traffic routing. In practice, the ILP approach applied to traffic engineering in WANs may even be intractable in many cases due to memory overflow during the computation. That being said, both the approximate and the exact techniques provide poor granularity and cannot quickly adapt to fast network conditions changes.

With no doubt the use of IP Traffic Engineering is complex and of little flexibility and granularity in WAN environment. Traffic flows tend to quickly vary in terms of offered load. The relative bit rate predictability and stability inherent to stream traffic in the years 1990s that we underlined above is no longer valid today because of the emergence of adaptive-coding techniques (for instance used for video). Figure 1 shows a typical traffic variation for voice and data traffic on a Service Provider link.

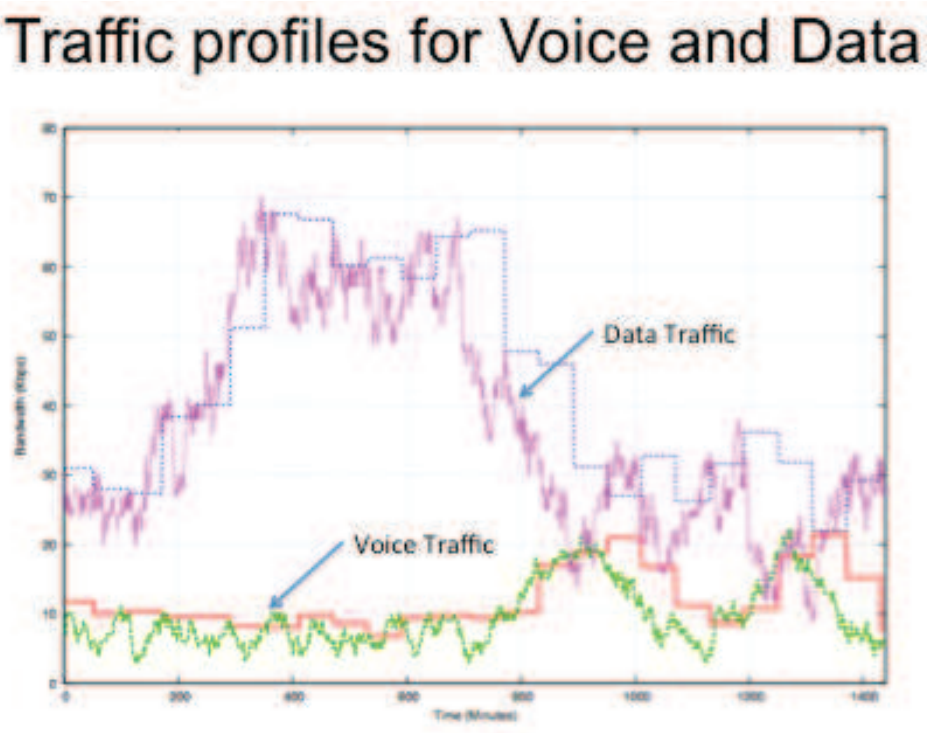


Figure 15 ***Traffic Load variation on a Service Provider link***
[Source: traffic monitoring of a link in a Service Provider in Japan]

Thus, it may be required for the off-line IGP link cost computation engine to update the link cost in the network several times during the course of the day to re-optimize traffic routing according to the traffic demand. The tool would consequently re-configure the links cost on-the-fly thanks to a command sent to each router in the network or a Simple Network Management Protocol (SNMP) [5] Management Information Base (MIB). Although feasible, the level of granularity is

low since the modification of a single link metric in the network ineluctably has an impact on the overall traffic routing operation in the network. This parameter must be taken into account by the link cost path computation. In practice, all new link cost adjustment cannot be performed strictly simultaneously and link cost updates take several (dozens) of milliseconds in the network for the IGP to update the Link State Database (LSDB). As a consequence, this may lead to temporary routing loops and congestion in the network, yet another parameter to take into account.

In the early 2000s the emergence of a new technology called MPLS Traffic Engineering allowed for the support of a number of new services and functionalities in data networks such as Virtual Private networks and Traffic Engineering (more details on MPLS TE can be found in [6] and [7], co-authored by the author of this thesis). A simple way to describe how MPLS TE can be used to route traffic efficiently in the network according to the traffic demand and network resource is to illustrate the well-known "fish problem", depicted in Figure 2.

In IP networks, each packet is individually routed according to the IP destination present in the IPv4 or IPv6 header at each hop, using the routing table populated by the routing protocol. In most cases, the routing protocol makes use of the Dijsktra algorithm to compute shortest paths using static link costs.

Referring to Figure 2, packets originated by the routers R1 and R2 and destined to router R8 would follow the same path, leading to a potential congestion on the North path and leaving unused resources on the South path (dashed arrows), also referred to as the "Fish problem". IGP link costs could be modified in order to provide equal cost paths between R3 and R5, thus allowing for Equal Cost Path Multiple path (ECMP) in order to load balance the traffic along a set of N paths. Unfortunately, although the problem is easy to solve in a such a simple network topology, existing WANs are significantly more complex, and optimizing link cost to load balance traffic with millions of flows and thousands of links in a time varying traffic load environment quickly becomes cumbersome and does not provide the required high level of granularity as discussed in the previous sections. Note also that in order to load balance traffic across a set of paths, these paths must provide equal cost. Asymmetrical load balancing may lead to routing loops with hop by hop IP routing that would eventually be resolved at the cost of extra complexity in forwarding and potentially longer paths than required.

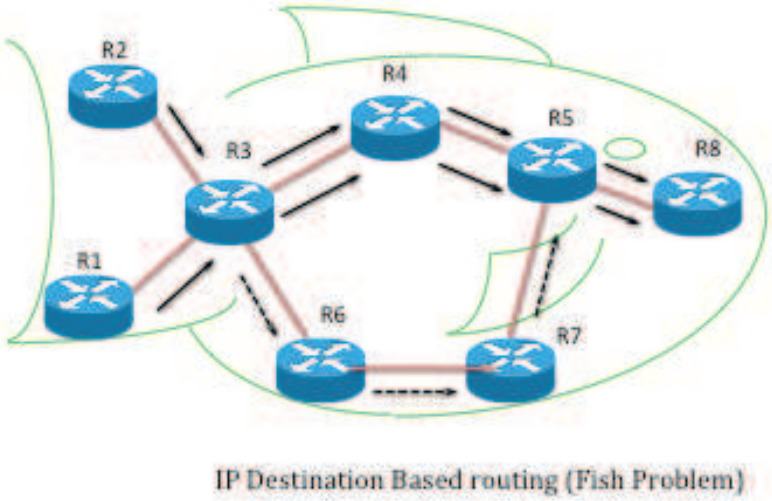


Figure 16 – *Illustration of The Well-known Fish Problem*

By contrast, MPLS Traffic Engineering consists of computing TE LSPs or called tunnels (T1 and T2 on Figure 2) between each pair of LSRs. The traffic is then encapsulated in TE LSPs as shown in Figure 3, without any further destination-based hop-by-hop routing in the network, thus simply following the TE LSP thanks to label switching.

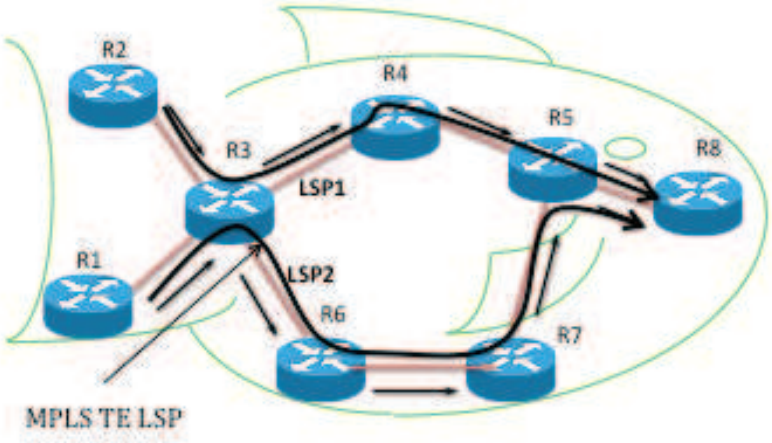


Figure 17 – *Solving the Fish Problem using MPLS Traffic engineering Label Switched Paths*

The organization of this manuscript is described in a few pages. At this step, let us just mention that Chapter 1 provides an overview of the functional components, algorithms and protocols of MPLS TE.

In addition to engineering the traffic, a key component of any IP/MPLS network to meet the SLA and provide guaranteed QoS, it is imperative to provision the network with QoS-related algorithms and techniques (that are outside of the scope of this thesis). Several research papers have been published that discuss in great details the number of technologies required in packet networks such as buffer management, capacity assignment (thus traffic engineering), flow and capacity assignment (which by itself covers a number of technologies developed to that end). For instance, reference [8] covers these aspects in details discussing the mapping between SLA and network as well as networks' performance constraints. There is a large number of papers in the literature covering various aspects of QoS such the Qos architectural model (Difserv versus Intserv), call admission control (CAC), bandwidth reservation, congestion avoidance, queuing disciplines to mention a few. It is worth mentioning reference [9] due to Prof. Marco Ajmone Marsan *et* alter that specifies a framework related to admission control and path allocation in Diffserv enabled networks. In this paper, the authors consider the overbooking probability thanks to a worse case mathematical model.

In reference[10], A. Cuadra *et alter* discusses in great details the need for efficient traffic monitoring in order to assure QoS and propose a detailed architecture.

Motivation and contributions

It quickly became apparent that one of fundamental technical challenges of MPLS TE lied in the complexity of computing paths for the TE LSP or tunnels. TE LSPs are both used for steering traffic in the network while optimizing the network resources in order to meet the SLA of each activated connection. MPLS TE also provides high reliability thanks to TE LSPs (backup tunnels) used to quickly reroute traffic in the network upon the occurrence of a network component failure.

The simplest algorithm known as CSPF (Constrained Shortest Path First) for MPLS TE consists in taking into account the characteristics of the TE LSP (bandwidth, affinities, priority, …), the available network resources distributed by the extended routing protocols and then prune the links that do not satisfy the TE LSP's requirements before running a shortest path computation algorithms such as Dijkstra [3]. The CSPF algorithm is undoubtedly extremely appealing for its simplicity in terms of computational running times and can be used in a distributed fashion on all nodes in the network for the computation of their TE LSPs, with no need for synchronization. Unfortunately, the distributed CSPF algorithm does not allow for solving several critical problems, and this for three reasons:

1. The lack of synchronization may lead to bandwidth fragmentation in the network and the inability of placing a TE LSP although such a path exists, which can be solved by re-organizing globally the TE LSPs' placement in the network ,

2. Visibility of the network resources may be partial when computing inter-domain TE LSPs, thus forcing to use per-domain path computation that are highly sub-optimal,

3. The use of TE LSPs for protection of the traffic upon network element failure requires sophisticated strategies to avoid requiring too much network backup capacity in the network.

So far, the only choices were either to use CSPF as a distributed path computation technique or the Network Management System (NMS). The NMS-based approach has been used in more static environments such as SONET/SDH over the past two decades and did not fulfill the requirements of dynamic path computation. Our objective is to propose a computation paradigm shift by off-loading the nodes of the network from the computation of the TE LSP.

This new architecture is called the Path Computation Element (PCE) and is the result of years of research, implementations and deployment in large-scale networks. My research activity started in 2003 where I was part of Cisco Engineering in Boston, MA. During the course of this intense research work activity, I managed to investigate a number of technologies, protocols and architectures that led to this new breakthrough architecture called the PCE. The PCE is now used in a number of contexts such as traffic optimization, inter-domain routing but also inter-layer routing and optimization across layers.

I got a chance to collaborate with a number of universities such as UT Dallas (Professor Andre Fumagalli), Drexel University in Philadelphia (Professor Jaudelice de Oliveira), the University of Ghent (Professor Piet Demeester), the MIT (Professor Neil Gershenfel - in the context of embedded systems) and the University of UMass to mention a few.

Several research papers have been published in conferences (e.g. IEEE Infocom and IEEE Globecom) and journals (e.g. IEEE Communication magazines, IEEE networks, Computer networks).

A number of novel technologies, algorithms, protocols and architectures arose from this research that have been patented (more than 150 patents) over an almost 8 year-period of research. Furthermore, stating the obvious, standardization is absolutely critical for a new technology to emerge and get deployed, leading to further research. To that end, I published about 30 standards at the Internet Engineering Task Force (IETF) during the past 10 years that are related to IP/MPLS and the PCE architecture, the core component of this thesis.

It is worth mentioning that the PCE architecture keeps evolving and its scope of applications gets extremely wide; new algorithms and protocols are being proposed for standardization in the IETF PCE Working Group that I have been co-chairing since 2004, opening the door to a wide range of new applications.

During these past 9 years my research contributions has mostly been on the emergence of a new architecture for the computation of intra and inter-domain paths that are detailed in this thesis.

The PCE architecture led to the specification of new protocols such as the PCEP (Path Computation Element signaling Protocol) protocols (that I designed in partnership with a few other contributors) and the specification of new algorithms such as the BRPC (Backward Recursive Path Computation) algorithm discussed later in this thesis. The BRPC algorithm allows for the optimum computation of inter-domain MPLS TE paths. In addition we specify a new distributed algorithm for the computation of a set of backup paths while minimizing the required amount of back-up capacity in a IP/MPLS network.

Last but not least, one of the next waves of the Internet and public IP networks, also known as the “Internet of Things (IoT)”, consists in connecting a wide range of new smart (small footprint) devices to IP networks called smart object networks (e.g. sensors, tags and actuators). The connection of these smart objects to IP networks opens the door to a vast number of new applications ranging from Smart Grid networks, Industrial automation, Connected vehicles, smart cites, home/building automation to mention a few. It clearly appears that the PCE architecture, with some adaptations proposed in this thesis, is very well suited to the IoT. We conclude this thesis by proposing various architectures and algorithmic enhancements so to use the PCE in support of IoT based networks.

Thesis outline

This manuscript is organized in three main parts.

Part 1 entitled "Background material" provides in a single chapter the basic knowledge in IP/MPLS protocols and traffic engineering on which relies the contribution of this thesis. **Chapter 1** entitled "State of the art: MPLS TE and Fast Reroute" recalls briefly how Traffic Engineering has been progressively introduced in core networks, more particularly with the emergence of service integration.

Part 2 entitled " The Path Computation Element and its extensions" is itself made of Chapters 2, 3 4 and 5. The PCE architecture is based on a client-server model that allows for off-loading TE LSP computation on a functional path computation engine called the PCE. In this new architecture, Label Switch Router called clients (Path Computation Client – PCC) interact with PCE(s) for the computation of TE LSPs (thanks to a new reliable protocol called the Path Computation Client Protocol (PCEP) used to convey TE LSP’s attributes and constraints) and get in return TE LSPs. Before the emergence of the PCE architecture, TE LSPs were computed using a distributed algorithm called Constrained Shortest Path First (CSPF) where each LSR computes the shortest constrained path across the network using the Dijkstra algorithm, taking into consideration the TE LSP’s constraints such as the bandwidth, priority, affinity to mention a few parameters, the network topology and available resources advertised by a link state routing protocol such as OSPF or IS-IS.

Chapter 2 entitled " The PCE architecture and its protocols " describes in detail the Path Computation Element (PCE) architecture as well as the role of each of the building block involved in PCE. Concerning the traffic engineering database (TED), the concept of request synchronization is introduced in order to show its ability to mitigate the issue of TED lack of synchronization. The PCE discovery and the load balancing problems are investigated. We then describe the signaling protocol known as the Path Computation Element Protocol (PCEP) that allows Path Computation Clients (PCCs – routers) to request path computation from a PCE. The PCEP protocol is based on TCP, with several messages used to send a path computation requests, receive path computation results and notify of errors, or even send notifications related to the PCC or PCE states. PCEP is a sophisticated protocol, designed to be extremely flexible in order to support future message types, and other extensions required for a wide set of scenarios. PCEP supports requests packing, bundling of correlated and/or synchronized requests distributed across a set of PCEs: a PCC can pack a set *S* of requests, indicate that a set of requests must be processed in a synchronized fashion for a variety of reasons (optimization, dependency between the requests such as the computation of a set of diversely routed TE LSPs), provide a number of indications on whether constraints are optional or mandatory. In return, the PCE can provide a path (route) that can be partial or complete, with path-keys when confidentiality across multiple domains must be preserved. The PCE may also indicate a “closest match”, thus suggesting constraints’ values when the request cannot be satisfied. We also study in detail the Finite State Machine of the protocol and we conclude this chapter with various monitoring tools. The two variants of PCE known as Stateless PCE and Statefull PCE are depicted and compared. The dynamics of PCEP is studied in detail in terms of protocols, signaling behavior and state machine.

Chapter 3 entitled "Backward recursive algorithm". Path computation of TE LSP across multiple domains, where a domain may either be a routing area or an Autonomous System, has always been a strong requirement and a major technical challenge. Indeed, in traditional systems whereby the head-end LSR computes the TE LSP using the CSPF algorithm, it becomes impossible to compute an optimum end-to-end path because of partial visibility of the network, which is the case of inter-domain. Thus the only available approach prior to the emergence of the PCE-based computation algorithm exposed in this thesis, consisted in computing TE LSP thanks to the use of a per-domain approach. With this technique, the PCC computes the TE LSP path up the next selected border router, that itself computes the path segment in the adjacent domain, and the process repeats up to the destination. Per-domain path computation cannot guarantee to find the most optimum shortest constrained path across domains simply because an entry border router has to select the next hop border router without knowing whether that border router can provide the best path to the destination. Several optimizations have been proposed in the literature consisting in advertising the aggregated resource information across domains that were all proven to poorly scale and barely increase the computed path quality. Furthermore, such a per-domain path computation approach may become extremely problematic in congested networks since a

selected next hop border router may not be able to find any path satisfying the constraints. This would lead to a call set up failure, and by consequence, an attempt to find a border router offering a path to the destination in using for example crankback mechanisms at the cost of extra-signaling and increased path computation time.

In chapter3, we introduce a new algorithm called the Backward Recursive Path Computation (BRPC) that makes use of a multi-PCE collaborative approach. This technique involves a chain of dynamically discovered PCEs that allows for the computation of optimal (shortest constrained) inter-domain TE LSPs. The BRPC algorithm involves parallel and serialized computations. It guarantees the selection of the shortest constrained inter-domain TE LSP, while minimizing the signaling overhead. Thanks to the usage of cryptography techniques, the BRPC algorithm preserves the confidentiality of the paths traversing multiple domains. The last section of Chapter 3 is dedicated to the performance evaluation of the BRPC algorithm. A large set of WAN configurations is considered ranging from small size networks with 17 nodes, to very large mesh networks with 83 nodes and 167 links. We show through discrete event simulations that the BRPC algorithm outperforms in comparison to existing per-domain path computation. Our performance metrics is based on three parameters: the path cost, the CAC failures (we mean by CAC failure the fact a TE LSP has to be rerouted when the call setup duration goes beyond a given admissible threshold), and the signaling delays. An approximate analytical modeling of the BRPC algorithm based on M/D/1 queue is commented. In spite of its simplicity, this analytical model used with identical network configurations and traffic scenarios shows comparable results with our discrete event simulations.

In **Chapter 4** entitled "PCE selection techniques and Stateless PCEs". we specify several algorithms for PCE selection. Other mechanisms dealing with path re-optimization are considered. A bandwidth defragmentation algorithm is proposed in order to reduce the blocking probability in a PCE-based path computation architecture.

Chapter 5 entitled " Distributed PCE for node protection and bandwidth sharing " is the last chapter of Part 2. Another critical problem known as “Bandwidth protection” refers to the ability to provide fast recovery along backup paths offering non-degraded quality of service. As we pointed out earlier in this section, network availability is critical. Upon network element failure such as a link or node failure, the Point of Local Repair (PLR) that is the node immediately upstream to the failure locally reroutes traffic onto backup paths (pre-computed backup tunnels). This allows for fast rerouting times of the order of a few dozen milliseconds, if not less. Although such backup paths may be used for a short period of time, it became required to provide non-degraded services, thus offering equivalent Quality of Service (QoS) during reroute. The first naïve approach consists in providing equivalent bandwidth along backup paths; unfortunately such an approach is extremely costly and it became critical to optimize the amount of resources dedicated to backup, also referred to as to the network backup capacity. In this

chapter, we propose a new approach based on the PCE-architecture so as to provide bandwidth protection while minimizing the required backup capacity in the network thanks to a distributed PCE-based architecture. By allowing each node to serve as a PCE for the computation of next-next-hop backup tunnels with bandwidth protection headed at each immediate neighbor, such back-up tunnels can be computed in order to share bandwidth between backup tunnels that protect independent resources. Resources are said to be independent when the probability of simultaneously failures of these resources is negligible (also called the single failure assumption).

The aim of this chapter is to propose a new approach for the computation of backup tunnels in order to quickly reroute traffic thanks to local protection along backup paths guaranteeing equivalent bandwidth (bandwidth protection) while minimizing signaling and backup network capacity. **Chapter 5** specifies a new model called the Facility Based Computation Model (FBCM): each node acts a PCE to compute the Next-next-hop (NNHOP) backup tunnels for its neighbors in case of its own failure. All of these tunnels cannot share bandwidth since in case of failure of that node they would all become simultaneously active. By contrast, because each node acts as an independent PCE, they use all of the backup resources of the overlay backup network: consequently, the backup tunnels computed by separate PCE implicitly share bandwidth, which is precisely the objective if we make the very reasonable assumption that nodes are unlikely to fail at the exact same time. This allows for providing a very efficient distributed model whereby bandwidth protection is provided while maximizing the degree of sharing of backup tunnels protecting independent resources. FBCM does not require additional RSVP signaling, Call Admission Control modifications and complex/heavy IGP extensions. The model is expanded to cover the case of bi-directional links but also overlapping Shared Link Risk Group (SRLG) thus expanding the scope of applicability of these protocols to the bandwidth protection of all network elements.

Part 3 entitled " Applicability of the PCE architecture to the Internet of Things " covers in two chapters (**Chapter 6** and **Chapter 7**) the applicability of the PCE architecture to the Internet of Things (IoT). This research outlines new technical problems that have become solvable thanks to the PCE architecture and its protocols. The IoT where large-scale IP networks connecting highly constrained objects such as sensors, RFID tags and actuators via low power/low speed and instable links led to the emergence of the wide variety of hard problems to solve that will benefit from PCE-based architecture.

In **Chapter 6** entitled " Push-based packet and Event Inspecting (PPEI): A new PCE-based architecture for the Internet of Things", we extends the PCE architecture for its applicability to the Internet of Things (IoT). The IoT corresponds to IP networks for which human decisions at the origin of the majority of path establishments are replaced by autonomous sensors. In a first section, we recall the main characteristics of the IoT. We then propose a new PCE architecture called the Push-based Packet/Event Inspecting (PPEI) PCE suited to the IoT environment. The PPEI

PCE interacts with highly constrained objects in order to solve a wide range of new challenging problems where the number of constraints in the network is potentially extremely large (limited bandwidth, unstable networks, limited processing power, memory and energy on end devices etc.).

Chapter 7 entitled " Routing and Traffic Engineering in PPEI PCE-based architecture" focuses on the routing problem in the context of IoT, showing how an adapted PCE-based architecture can be used for assisted routing in the IoT.

We provide our general conclusions of this thesis in **Chapter 8**. In this same chapter, we propose a few possible perspectives of investigation in the continuation of the original protocols and algorithms introduced in this manuscript.

NOTA:

After Chapter 8, we provide in a chronological order the list of our publications classified in conferences, journals, standards and books. An alphabetic list of the acronyms used in this manuscript is then provided. It has to be noticed that in the remaining pages of this document, all the references followed by the star “*”symbol refer to publications in which we are author or co-author.

Part 1 – Background Material

1 State of the art of MPLS Traffic Engineering and Fast Reroute

In this chapter, we provide an overview of the state of the art techniques used to engineer and protect traffic in data networks using MPLS Traffic Engineering (MPLS TE). In a first step, we briefly describe the MPLS protocols, algorithms and modes of operation. We recall the principle of MPLS Traffic Engineering LSP attributes and database with the associated signaling aspects. The concepts of Traffic Engineering reoptimization and Fast Reroute are also summarized. In a second step, we deal with complex aspects of MPLS, especially for the inter-domain environment. We review two major technical challenges related to the computation of inter-domain routing path computation of Traffic Engineering Label Switched Path (TE LSP) and backup tunnels for which no efficient solution was available prior to the emergence of the technology specified in this thesis.

The aim of this chapter is to provide a high level overview of MPLS TE; more details can be found in [6] and [7]*(co-authored by the author of this thesis).

1.1 Traffic Engineering LSP Attributes

The aim of this short section is to provide a non-exhaustive list the MPLS TE attributes, more details can be found in [7].

A Traffic Engineering Label Switch Path (TE LSP) is characterized by a series of attributes:

- Destination of the TE LSP identifying the node where the TE LSP terminates, the source and the destination of the TE LSP being called the Head-end Label Switch Router (LSR) and the Tail-end LSR respectively

- The bandwidth of the TE LSP specifies how much of bandwidth is required for the TE LSP. The bandwidth of a TE LSP is explicitly signaled using the RSVP-TE protocol ([11]). Determining the required bandwidth of a TE LSP is a daunting task and a wide set of complex strategies has been specified, ranging from simple approaches consisting of sizing the TE LSP to the peak of traffic during the course of the day multiplied by some over-under-booking multiplier, to a more dynamic approach illustrated in Figure 4 where the head-end LSR measures the actual traffic sent onto a TE LSP at regular intervals and dynamically recomputes the TE LSP that may or may not follow the same path (this technique is called “Auto-bw”).

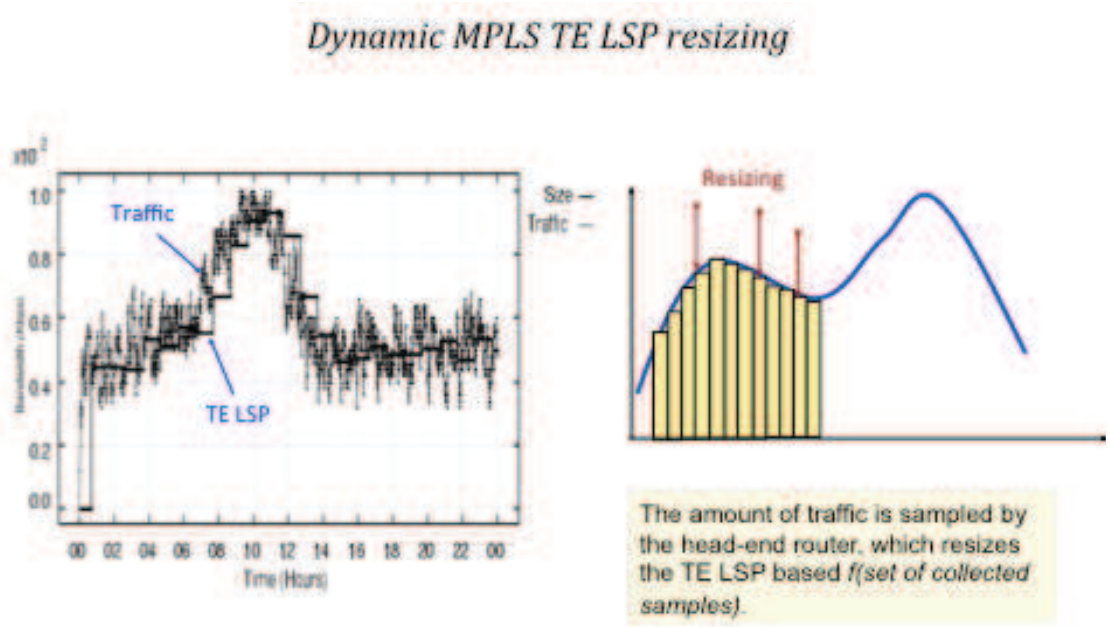


Figure 18 – *Auto-bandwidth approach used to dynamically adjust the TE LSP bandwidth according to measured traffic load*

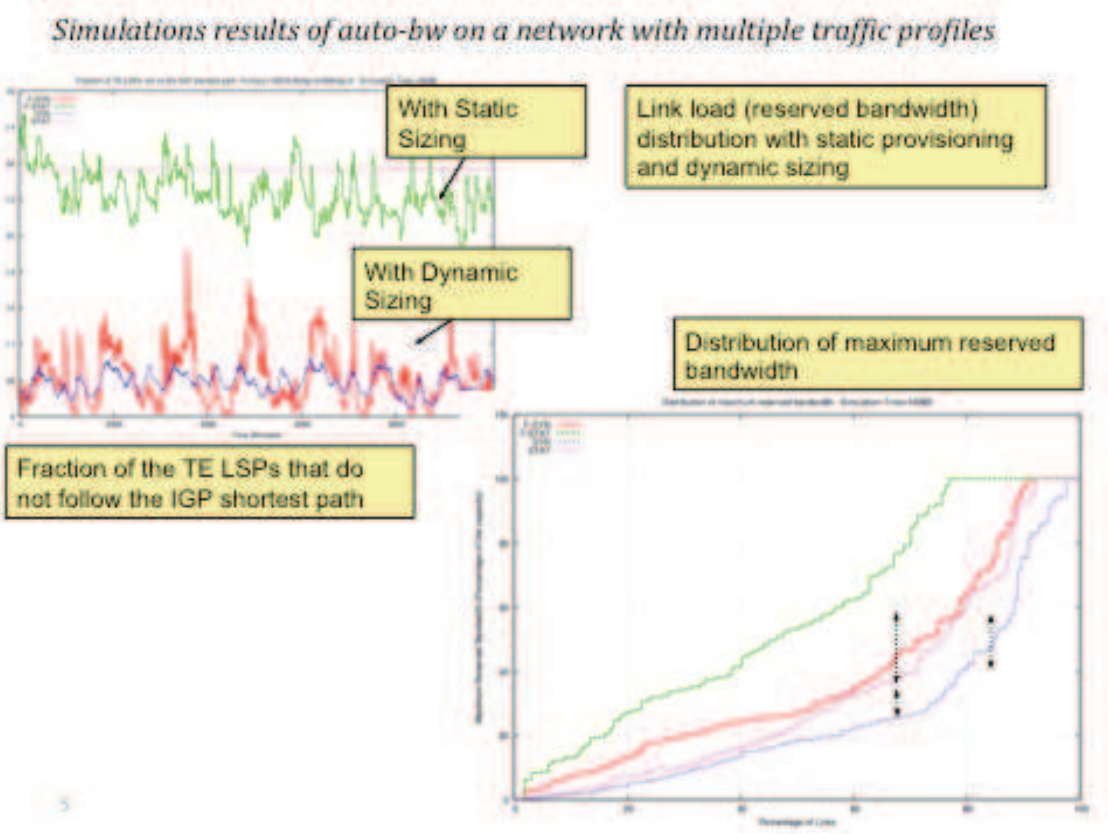


Figure 19 – *Simulation results of the auto-bw dynamic TE LSP sizing strategy*

Detailed simulations have been performed in [12]*, [13]* and [14]* that evaluate the degree of efficiency of various auto-bandwidth strategies in terms of degree of bandwidth booking, impact of overbooking on the number of TE LSPs that do not follow the shortest path because upper bounds of bandwidth have been taken into account to name a few performance metrics that were studied in these research papers.

- **Affinities** are a simple coloring scheme (using a 32-bit field flag) allowing for the assignment of attributes to links according to a user defined schema so as to explicitly request a TE LSP to include or exclude links matching specific criteria (for example, "Include Blue links" where the color blue is used to mark a link as a protected link)

- **Preemption**: MPLS TE supports the notion of preemption according to a 7-level priority scheme. Consequently a TE LSP with priority *X* (where a lower number indicates a higher priority) can preempt a set of TE LSPs of priorities *Pi* if and only if *X*<*Pi*. The decision on the set of TE LSPs that must be preempted should a new TE LSP require freeing up bandwidth allocated to lower priority TE LSP is not specified by MPLS TE and is implementation-specific. A number of strategies can be implemented with different objective functions (e.g. minimize the number of preempted TE LSPs, maximize the priority of the preempted TE LSPS, find the lower priority TE LSPs set that frees up a bandwidth *X*' as close as possible to *X*, …). These strategies have been studied in great details in [15]*.

- TE LSP **metrics**: links have different properties such as the bandwidth but also the propagation delay and bit error rate to mention a few. In addition to the IGP metric, a second metric called the TE metric has been specified that can be used to find the shortest constrained TE LSP according to a metric different than the IGP metric. The TE metric can be used to reflect the delay where the IGP metric can be used for bandwidth. As discussed in Chapter 2, the Path Computation Element Protocol (PCEP) used in the PCE architecture specified in this thesis allows a client to request the computation of the shortest constrained path while indicating the metric to optimize the path cost against. Optionally, a second metric can be specified to indicate an upper bound and can be used as another constraint for the TE LSP.

1.2 The Traffic Engineering DataBase (TED) and IGP routing extensions

Link state routing protocols such as OSPF and ISIS have been extended in order to flood TE link characteristics ([16], [17]). Such characteristics are either static (e.g. affinity) or dynamic (e.g. amount of reserved bandwidth per priority). The collection of the TE link characteristics for the whole network forms a topology map with the corresponding set of available resource and TE related link attributes and is called

the Traffic Engineering Database (TED), by contrast with the Link State Database (LSDB) that is used by the IGP for IP routing.

Dynamic TE link attributes such as the amount of reserved bandwidth per priority is not refreshed each time a new TE LSP is signaled in the network. Indeed, some links may carry a very large number of TE LSPs and a newly signaled (or torn down) TE LSP may marginally affect the amount of reserved bandwidth; in this case, it is preferable not to trigger an IGP update, which preserves the IGP scalability. Instead it is preferable to rather refresh the TED if the reserved bandwidth has changed significantly. Note that most of the implementations make use of a non-linear thresholds mechanisms, where IGP updates are triggered more frequently as the level of reserved bandwidth gets closer to the total bandwidth pool allocated for the priority. The downside of this approach is that the TED may not accurately reflect the actual available bandwidth at each hop, which may lead to call set up failures.

1.3 Signaling of a TE LSP

For several years two protocols were available for the signaling of TE LSPs, namely CR-LDP and RSVP-TE ([18] and [19]) before the decision of the Internet Engineering Task Force to abandon CR-LDP and elect the exclusive use of RSVP-TE as the signaling protocol for TE LSP. RSVP-TE extends the RSVP protocol specified in [20] in order to allow for the signaling of TE LSPs. Several new messages are specified such as the Path, Resv, Path and Resv Error messages that travel in different directions that each comprises a series of objects and in particular the TE-related object to signal TE LSP attributes such as bandwidth, affinities, priority, requirement for Fast Reroute protection to mention a few.

In a nutshell, RSVP Path messages travel in the downstream direction (from the head-end to the tail-end LSR) and signals the TE LSP attributes according to the computed TE LSP; the path is encoded using an object named the Explicit Route Object (ERO). By contrast, Resv messages travel in the opposite direction to confirm the proper signaling of the TE LSP and allows for label allocation and the population of the label tables at each hop also called the LFIB (Label Forwarding Information Base).

Once a TE LSP has been successfully established, the traffic steered onto TE LSPs are label switched in the network, thus following the TE LSP, which may not be identical to the IGP path. Since RSVP is a soft state protocol, states are refreshed at regular intervals thanks to Path and Resv messages, with jitter to avoid global synchronization. Refresh reduction techniques (and reliable messaging) have been elaborated to minimize the protocol overhead ([21]). Note also that TE LSP are by default unidirectional although extensions have been proposed in [22] to set up bidirectional TE LSPs, which are required for several packet and non packet TE LSPs.

1.4 Reoptimization of the TE LSP

TE LSP are reoptimized; reoptimization triggers may be timer-based (in which case the head-end LSR simply re-computes a TE LSP after the expiration of a local timer and reroute the TE LSP if a better path can be found in the network) or because of the occurrence of a specific event such as a topology change (e.g. restoration/addition of a link). It is worth pointing out that TE LSP reoptimization is performed using a Make-Before-Break (MBB) technique: when a TE LSP that used to follow a path *path_1* is reoptimized to follow a path *path_2* where Cost(*path_2*)< Cost(*path_1)*, then the TE LSP is signaled along *path_2* without double bandwidth accounting on the links shared between *path_1* and *path_2* thanks to the property of bandwidth sharing provided by the RSVP protocol. Once the TE LSP has been established along the new path, the traffic is switched to the new TE LSP and the old TE LSP is torn down, without any traffic loss, making the MBB procedure totally non-disruptive.

1.5 MPLS Traffic Engineering Fast Reroute (TE FRR)

Network availability is undoubtedly one of the most critical network performance characteristics, considering the number of critical applications that are based on the IP/MPLS protocol suite. To that end a number of technologies have been designed at all layers (optical, SONET/SDH, IP and MPLS) over the past two decades that are covered in great details in [23]*, and hundreds if not more of research papers have been published proposing fast recovery techniques.

Among these technologies, MPLS Traffic Engineering Fast Reroute specified in [24]*, also simply referred to as FRR, is undoubtedly one of the prevalent fast recovery techniques deployed so far in existing IP/MPLS networks. FRR is a local protection recovery technique whereby TE LSPs are locally rerouted onto backup tunnels that are pre-provisioned (referring to the term “protection” by contrast with “restoration” used by IP Fast Reroute for example) as shown in Figure 6:

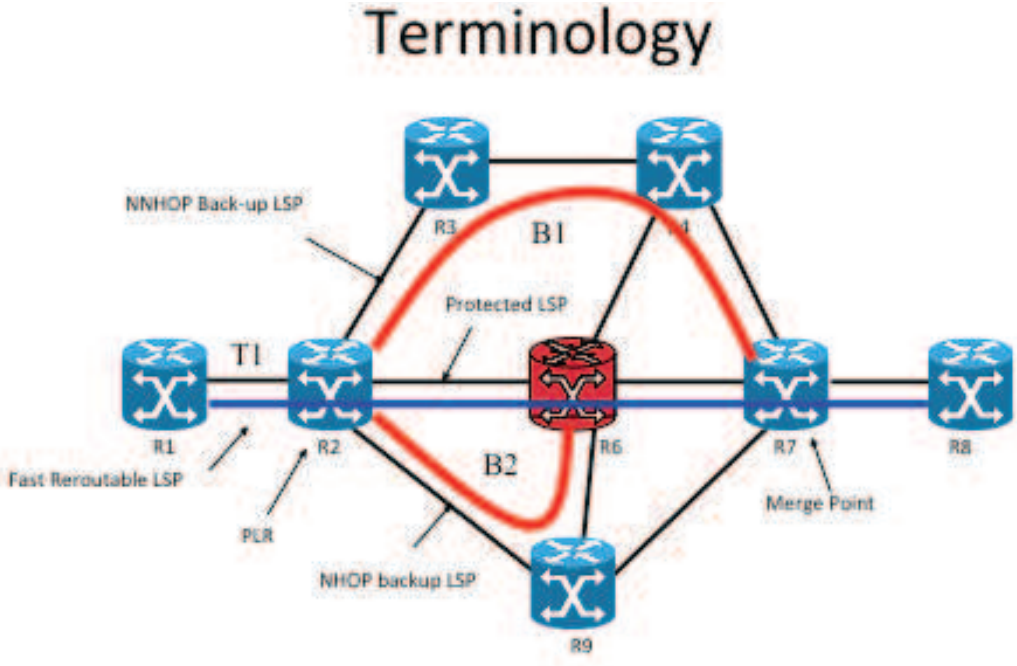


Figure 20 – *MPLS Traffic Engineering Fast Reroute Terminology*

Upon the detection of a failure thanks to low-layer interrupt messages (e.g. SDH or Optical) or fast keepalive mechanisms such as Bidirectional Forwarding Detection (BFD) (see [25]), the Point of Local Repair (PLR) locally reroutes each TE LSP onto the pre-selected backup tunnel and pushes an additional label that is removed prior to reaching the Merge Point (MP). Note that the backup tunnel used for link protection is a Next-Hop backup tunnel (e.g. B2 is a NHOP backup tunnel protecting against a failure of the link R2-R6) whereas for node protection the backup tunnel is a Next-Next-hop (NNHOP) backup tunnel (e.g. B1 is a NNHOP backup tunnel protecting against the failure of the node R6).

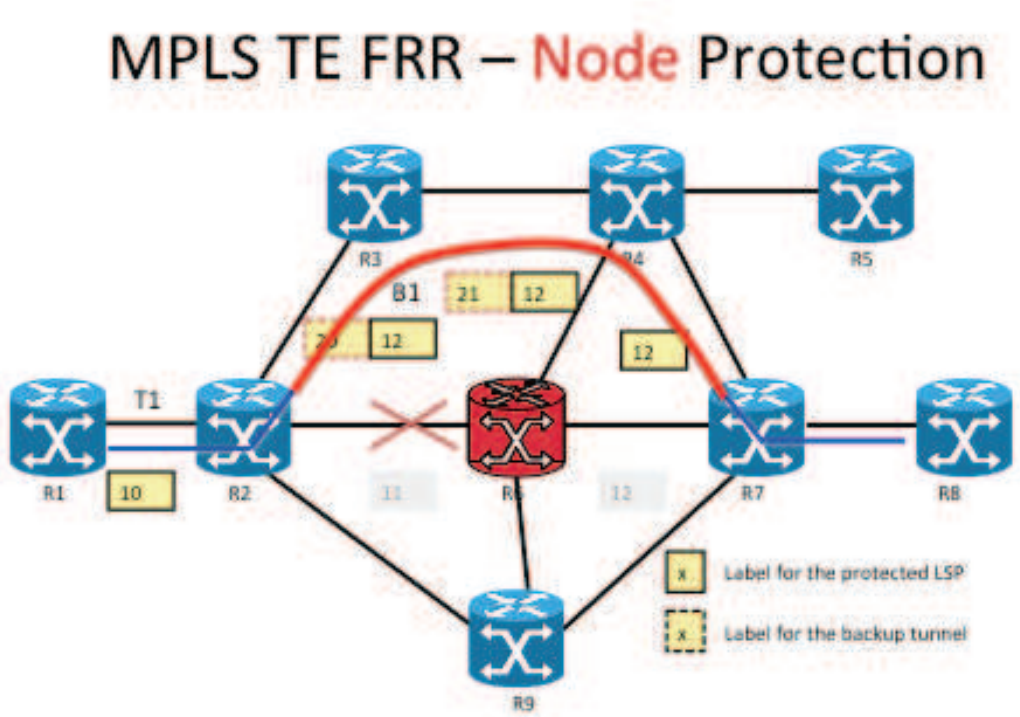


Figure 21 – *FRR Local Protection and reroute onto a NNHOP backup tunnel upon a node failure*

For example, as shown in Figure 7, once the failure of the node R6 has been detected by the PLR R2, the TE LSP T1 (from R1 to R8) is locally rerouted (encapsulated) onto B1. The PLR performs a label swapping so as to insert the labels expected for T1 by the MP R7 and the last hop LSR along the backup tunnel (R4) removes the outer label.

At this point, the PLR (node R2) notifies the head-end LSR of the local reroute that in turn gracefully reoptimizes the rerouted TE LSP T1 using the MBB procedure (see Figure 8).

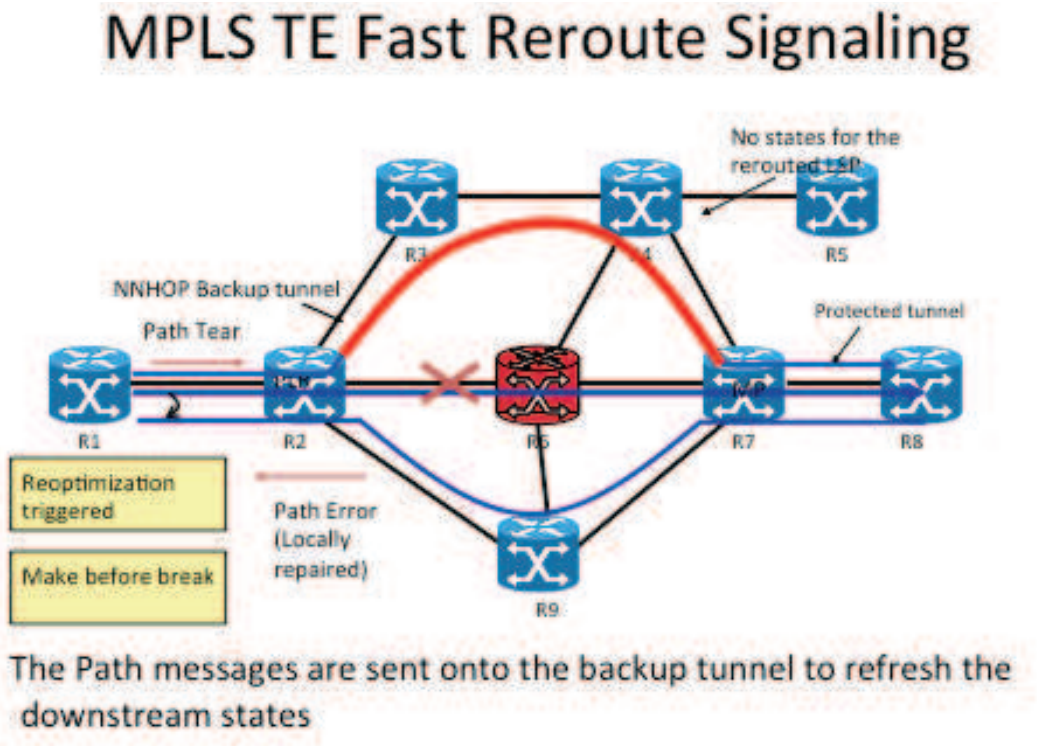


Figure 22 – *Signaling MPLS TE FRR*

Note that FRR is a non-revertive local protection mechanism: upon the restoration of the failed link or node, the PLR does not switch back the rerouted TE LSPs onto their original path. It is the responsibility of the head-end LSR to reroute/reoptimize the rerouted TE LSPs to follow the previously used path using the MBB procedure.

1.6 Unsolved MPLS Traffic Engineering technical challenges

In this section, we highlight several key technical challenges that remained unsolved by MPLS Traffic Engineering and led to the Path Computation Element architecture specified in this thesis, and that is now deployed in a number of operational networks (in large enterprises and Service Providers backbones).

1.6.1 Computation of Inter-domain TE LSP

Prior to the existence of the PCE-based architecture resulting from several years of research and exposed in this thesis, there was no solution to the computation of optimal paths for inter-domain TE LSPs. Link state routing protocols such as OSPF and ISIS fundamentally rely on the ability for each router that belongs to a routing area to share enough information with all of the routers in the area thanks to Link State Advertisement (LSA). As the number of routers increases, the LSDB comprising the LSA originated by the routers in the area also grows, so does the computation time of SPF (although not the limiting factor). Thus Service Providers and large Enterprises usually split their routing domain in areas. Traffic across areas is routed through Area Border Router (ABR) using a “distance vector” approach: for

example, the traffic from a router R1 in area 1 to router R2 in area 2 would transit via three areas: the area 1, the backbone area (connecting all areas in a hub and spoke fashion). The ABR would typically announce prefix reachability and a router with traffic destined to a remote area would select the closest ABR to route its traffic.

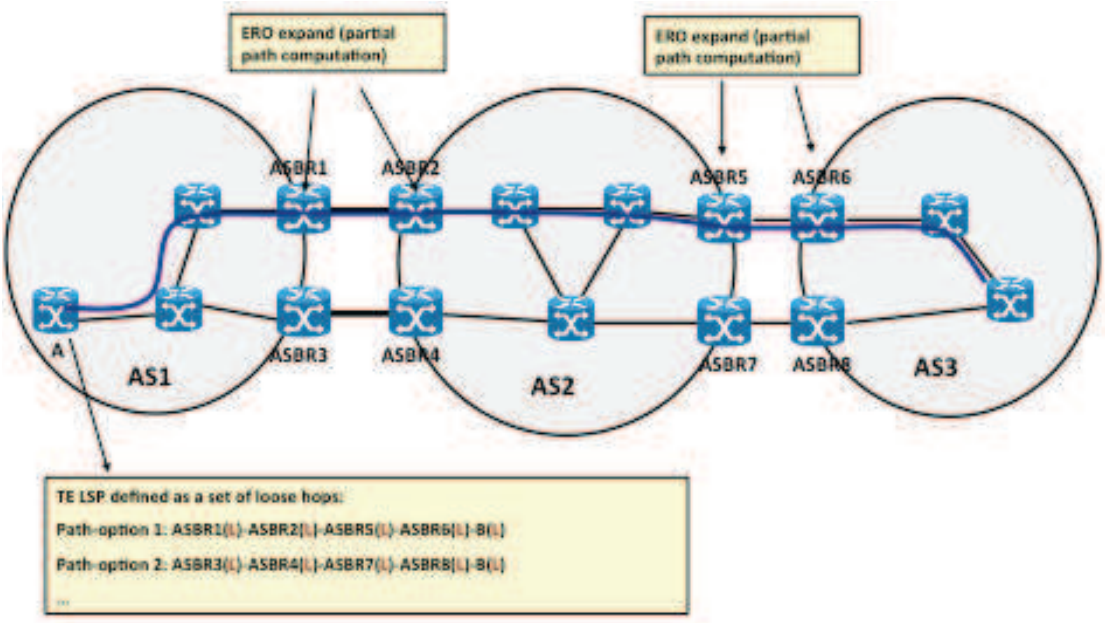


Figure 23 – *Routing of Inter-AS TE LSP with the per-domain routing approach*

Thus before the emergence of the novel PCE architecture, the only solution available and documented in [26]* was to follow a per-domain routing approach. The per-domain routing approach was the only technique available for inter-area and inter-AS TE LSP computation: figure 9 illustrates this approach in the context of an exemplary network composed of several Autonomous Systems. A TE LSP from a LSR A in Autonomous System (AS) AS1 to a LSR B is routed up to the next hop ASBR providing an exit point from the AS and advertising reachability for the destination address of LSR B. Such an ASBR would usually be selected according to the shortest constrained path cost to reach the ASBR. Upon reaching the ASBR, the signaling message (RSVP Path message) is then forwarded to the next hop ASBR (ASBR 2 in this example) at which point a CSPF (Constrained Shortest Path First) path computation is triggered on ASBR 2 to determine the closest ASBR providing connectivity to an ASBR advertising the destination and the process repeats until reaching the ASBR connected to the AS where the destination resides.

Such a simple (and naïve) approach unfortunately suffers from two major issues:

Issue 1: sub-optimal inter-domain TE LSP

The stitching of path segments between arbitrary chosen ASBR unfortunately does not provide the shortest constrained inter-domain TE LSP; indeed, since a node *N*

along the chain selects the next-hop ASBR *N+1* according to the shortest constrained path for the said TE LSP to that ASBR without visibility on the path cost from the node *N+1* to the next ASBR (or the final destination), it is not possible to guarantee that the end-result will provide the shortest constrained **end-to-end** inter-domain TE LSP. Indeed, back to the exemplary network depicted in Figure 9, if ASBR2 selects ASBR5 as the next-hop ASBR because the path to ASBR5 is of a lower cost compared to the path to ASBR7, and the path from ASBR6 to LSR B is very high or simply path_cost(ASBR2->ASBR5)+ path_cost (ASBR5->ASBR6) + path_cost (ASBR6->LSRB) > path_cost(ASBR2->ASBR7)+path_cost(ASBR7->ASBR8)+path_cost (ASBR8->LSRB) ***and*** path_cost (ASBR2->ASBR5) < path_cost (ASBR2->ASBR7), the decision of ASBR2 would lead to a sub-optimal constrained path computation.

Unfortunately such a per-domain approach cannot guarantee to find the shortest constrained path end-to-end, because the next-hop ASBR selection algorithm is performed without any knowledge/visibility of the topology and network resource in remote downstream domain(s).

Issue 2: Call set up failure and increased re-routing times in case of failure

The second issue may be even more problematic and relates to call set up failures. Indeed, when the RSVP Path message reaches a Node *N*, the node triggers a CSPF path computation according to signaled TE LSP parameters (bandwidth, preemption, destination, TE LSP attributes). If no path satisfying the constrained exists, a call set up failure would unavoidably take place, and the node failing to find a path satisfying the constraints would then trigger an error (RSVP Path Error message) sent in the upstream direction to the TE LSP originator. In this case, two approaches can be adopted:

- The signaled error can be sent back to the originator (head-end LSR of the TEP LSP) that may in turn select another next-hop ASBR; an optimization may consist in recording the set of visited ASBRs and piggyback the information in the signaled error,

- The previous ASBR processing the error message may simply intercept the message and trigger a CSPF computation after pruning the visited ASBR from the list of exit candidates. That technique has been experimented in other signaling protocol such as Private Network-to-Network Interface (PNNI) [27] for Asynchronous Transport Mode (ATM) networks and is also referred to as the crankback technique ([28]).

Unfortunately, none of the techniques above guarantees to find the shortest constrained path and although one can prove that they eventually converge, it can be shown that the set up time in case of call set failure may be quite large. Furthermore although the crankback technique usually performs slightly better in terms of call set up time in highly congested networks in terms of bandwidth reservation, call set up failure rates may still dramatically increase the time to establish an inter-domain TE LSP. This may be particularly problematic in case of TE

LSP failures not protected by local protection mechanism such as MPLS TE FRR since traffic would be dropped until the inter-domain TE LSP gets re-established.

It will be shown in Chapter 3 that the PCE-based path computation approach proposed in this thesis solves this issue and allows for the computation of optimum (shortest) constrained inter-domain TE LSP with reduced call setup failure rates compared to the per-domain path computation approach.

1.6.2 Bandwidth sharing between FRR tunnels protecting independent resources

The notion of bandwidth sharing between backup tunnels used by FRR and protecting independent resources is explored in depth in Chapter 6 when exploring how this research led to the adoption of an efficient bandwidth sharing approach using a distributed PCE-based approach. In short though, the issue with a naïve approach whereby each PLR independently computes its own set of (NHOP or NNHOP) backup tunnels with non-zero bandwidth to protect a pool of bandwidth is such that a large amount of backup capacity is required in the network (both for the primary TE LSPs and the backup tunnels), without the ability for backup tunnels protecting independent resources (i.e. resources that do not fail simultaneously) to share bandwidth according to the assumption that both tunnels are not active at the same time. A solution is specified in this thesis that relies on PCEs.

1.6.3 Protecting ABR and ASBR nodes with FRR tunnel

As briefly described in this chapter, MPLS TE FRR is a local protection technique that has been widely deployed to protect traffic from the failure of links and nodes and locally reroute TE LSPs onto back tunnels in a matter of a few dozens of milliseconds. Protecting inter-area TE LSP from the failure of an ABR (and similarly inter-AS TE LSP from the failure of an ASBR) quickly became a major objective since all inter-area traffic traverse ABRs (and ASBRs in the case of Inter-AS TE LSPs), thus requiring mechanisms to compute inter-domain backup tunnels. In a sense, an inter-domain backup tunnel is no different than a regular inter-domain TE LSP except that it must avoid the protected node. Thus, the computation of inter-area NNHOP backup tunnels used for FRR node protection is similar to the computation of an inter-area TE LSP.

1.6.4 Global optimization of TE LSP in intra-domain scope

Last but not least, the optimum placement of a set of primary (by contrast with backup tunnels) TE LSPs within a single routing domain has been studied for years, and is known as being NP-Complete. A number of algorithms and heuristics have been proposed in research papers over the past four decades, even prior to the emergence of MPLS TE, since this problem is similar to the “bin-packing” problem. As shown in the next chapter, one of the major added-value of the PCE-based architecture lies in the ability to provide a dynamic architecture allowing Path Computation Client (PCC) such as Label Switch Router (LSR) to dynamically interact with one or more PCEs for the global optimization of TE LSPs with an intra-domain

scope. Such a problem can be tackled by using statefull PCEs with a dynamic interaction between PCCs and PCEs, according to the network states, level of optimization required for the network, rate at which new requests cannot be satisfied, in a significantly more dynamic fashion than with existing models using off-line predictive and traffic analysis tools combined with off-line path computation server.

1.7 Conclusion

In this chapter, we first reviewed the fundamentals of traffic engineering in IP networks with a particular focus on MPLS Traffic Engineering, which provides the ability to compute constrained shortest paths, signals TE LSPs along these paths and make use of label switching techniques by contrast with IP routing. The main objective of MPLS TE is to use more efficiently network resources according to dynamic traffic demands and the required quality of service.

Then we provided a high level summary of one of the most prevalent fast recovery technique known as MPLS TE Fast Reroute that relies on local protection of TE LSPs thanks to back-up tunnels.

Both techniques are central to this thesis since the Path Computation Element architecture was designed for the computation of primary and backup TE LSPs. Note that the PCE architecture is not limited to MPLS path computation and is also applicable to IP routing and other path computation problems.

Finally, we provided a summary of several well-known unsolved technical problems such as the computation of shortest constrained inter-domain TE LSPs or the efficient computation of backup tunnels with bandwidth sharing to mention a few examples. Such problems are now solved thanks to the PCE architecture and the set of algorithms and protocols designed in this thesis and are explored in detail in the following chapters.

Part 2 – The Path Computation Element and its extensions

2 The PCE Architecture and its protocols

This chapter represents the core of this thesis resulting from several years of research (specified in [29]): a new routing paradigm for computing MPLS Traffic Engineering LSP, new algorithms such as the Backward Recursive Path Computation Algorithm for inter-domain TE LSP, a new signaling protocol called PCEP along with other protocols and algorithms that are described in the following sections.

Note that an excellent overview of the PCE in inter-layer and multi-layer contexts, using single, multi and hierarchical PCE architectures can be found in [30].

2.1 The Path computation Element (PCE) Architecture

2.1.1 Path Computation Element (PCE)

Fundamentally, the PCE is a path computation engine capable of computing a path for a TE LSP satisfying several constraints such as the TE LSP bandwidth, destination, preemption level (priority), and other attributes of various sorts, in light of the set of available network resources for a given topology or set of network topologies. No assumption is made on the nature of the algorithm used for path (segment) computation. The path computation algorithm may either simply based on CSPF (a modified version of the Dijkstra algorithm after pruning the set of links that do not satisfy constraint) or more sophisticated path computation techniques used for global optimization or multi-constraints optimization problems based on heuristics (these algorithms being known as NP-Complete).

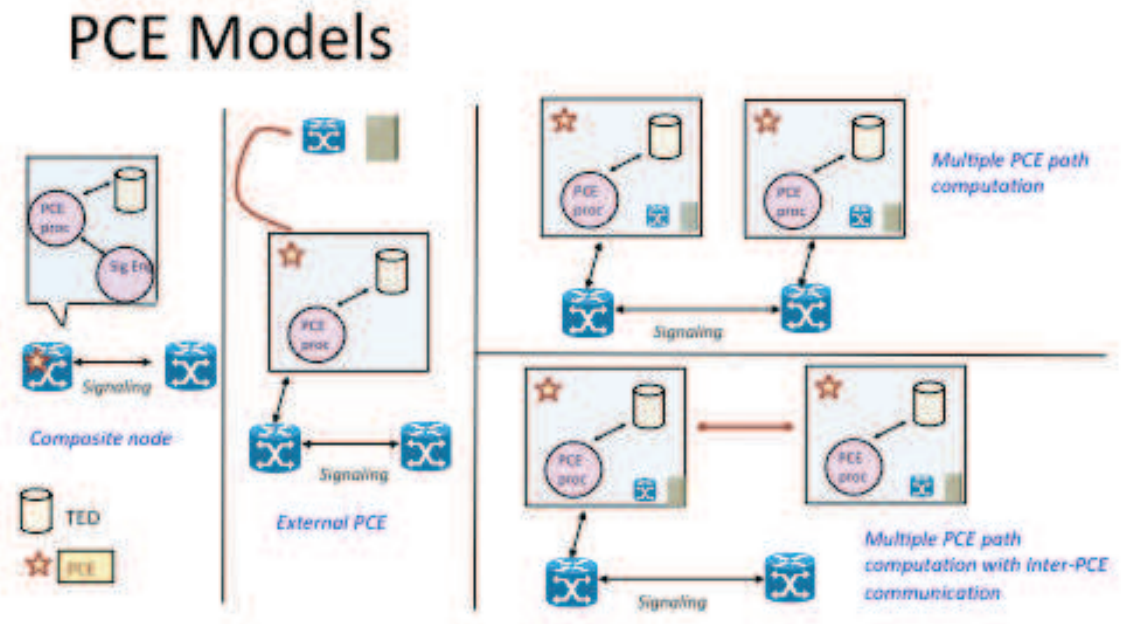


Figure 24 – *The Path Computation Element Architecture*

The overall PCE-based architecture is depicted in Figure 10.

Path computation may be centralized in which case a single PCE is used to compute all TE LSPs; conversely, a set of PCEs may be involved in the computation of the TE LSP in which case we refer to a distributed path computation model or a collaborative path computation model (referred to as "Multiple PCE path computation" in figure 10). Both models will be described in this chapter. Centralized and distributed path computation models are not exclusive and may be mixed in the same network. As shown later in this chapter, intra-domain TE LSP may use a centralized PCE whereas inter-domain TE LSP may be computed using a collaborative approach between a set of PCEs such as the BRPC procedure described in chapter 3 (one of the key component of this thesis).

As illustrated in Figure 10, the PCE is a functional entity and no assumption is made on the nature of the device hosting the PCE, which may either be a process running on an LSR or an application hosted on a standalone computer.

2.1.2 Architectural Building Blocks

In this section we briefly introduce the main building blocks of the PCE architecture; each of them leads to the specification of new protocols and algorithms in the context of this thesis that are covered in details in the next chapters. At this point, the objective is to briefly describe each of these building blocks and functional element of this architecture.

2.1.2.1 TED synchronization

The PCE architecture does not mandate a specific mechanism to retrieve the TED, required for path computation. The PCE can retrieve the TED thanks to a routing adjacency (directly, or via a Generic Routing Encapsulation (GRE) tunnel), which is by far the most common mechanism used in practice. The TED may alternatively be gathered using an out-of band mechanism, a manual operation or interaction with a Network Management System (NMS) or more recently using BGP extensions as described in [31].

2.1.2.2 Request synchronization

As discussed later in this chapter, the PCE architecture and the signaling protocol (PCEP) specified in this thesis support multi-request synchronization. For example, a PCC may send a request for the computation of a set of *N* correlated paths such as *N* diverse paths that are link/node/SRLG diverse used for 1:1 protection or simply to load balance the traffic between a pair of node along paths that cannot all be affected by a single failure.

2.1.2.3 PCE discovery and Load Balancing

First we made the architectural decision to unicast path computation request to a single PCE that may in turn invoke other PCEs to perform the path computation, by

contrast with other approaches whereby requests would be flooded to all PCEs in the network. Although static configuration of the PCE address on PCCs is a viable approach, we decided to specify a discovery mechanism to automatically determine the set of available PCEs. To that end, several approaches have been envisioned such as defining a new protocol or extend existing protocols.

Although several candidate protocols have been studied for dynamic PCE discovery, the most obvious candidate was the IGP itself considering that PCC would have to be part of the routing domain in most cases, in order to gather the TED. To that end, we extended both OSPF and ISIS (see [32]* and [33]*) piggybacking PCE-related information thanks to the specification of a new Type-Length-Value (TLV), called the PCED (PCE Discovery TLV) that itself comprises a set of sub-TLVs, each providing information related to the PCE, such as:

- PCE Address: IPv4/IPv6 address used to reach the PCE (Mandatory)
- PCE Path Scope: intra-area, inter-area, inter-domain or interlayer capability of the PCE to compute path along with a degree of preference for each path computation scope ranging from 0 to 7 and than can be used for weighted load balancing requests (Mandatory)
- PCE Domain: specifies the area/AS where the PCE has visibility and through which the PCE can compute paths.
- Neighbor PCE Domain: neighbor PCE domain towards which a PCE can compute paths.
- PCE Capability: this optional array of 32-bit fields allows for advertising PCE capabilities.

The PCED TLV is then carried within a Router Capability TLV in ISIS, which is itself leaked between ISIS levels according to the PCE path computation scope, and within an OSPF opaque LSA of type 10 and 11 of area and AS scope. Note that the PCED TLV could also be carried within BGP but such an approach has not been standardized.

2.1.2.4 Signaling protocol for PCC-PCE and PCE-PCE communication

The signaling protocol used by a PCC to send path computation requests and by the PCE to return computed paths to PCCs along with control planes messages to report error, notifications of various sorts, is a fundamental piece of the PCE architecture, and a key work item of this thesis.

Similarly to the PCE discovery covered in the previous sections several approaches have been studied namely: extending an existing protocols such as RSVP (which was initially proposed), LDAP [34], BGP [35] and other protocols, or alternatively define a new protocol for PCC-PCE and PCE-PCE signaling. It is the later approach that was chosen considering that none of the existing protocols met the PCE architecture signaling requirements nor would they have added a sufficiently low overhead for the PCC and PCE. That new protocol called PCEP is covered in detail in the following section.

2.1.2.5 Stateless versus Statefull PCE

During the early stages of this research we decided to specify two fundamentally distinct modes of operation for the PCE, namely the stateless versus statefull PCE.

- **Stateless** PCEs process path computation requests independently of each other without having to "remember" past path computation requests and established TE LSPs. A stateless PCE use as an input both the network topology, the available network resources (ie. the TED) and the requested TE LSP attributes. Consequently, a PCE may allocate the same resources to two non- synchronized path computation requests, which may lead to call set up failure should the available network resources not be sufficient to satisfy both requests. Still the PCEP protocol may allow a PCC to synchronize a set of *N* path computation requests (which are sometimes called "correlated" requests) in which case the set of requests is processed as a batch as opposed to individual requests; in a way this allows for temporary statefull computation of a limited set of TE LSPs.

- By contrast, a **statefull** PCE augments its knowledge of the network states by maintaining another database referred to as the LSP database, in addition to the TED. The LSP database comprises all of the established TE LSPs. Using both databases when processing path computation requests allows statefull PCE to perform more efficient network optimization.

  For example, a statefull PCE may find out that a request can be satisfied if a set of existing TE LSPs are displaced, thus effectively "de-fragmenting" the network bandwidth. Moreover, existing TE LSPs may be taken into account in order to make use of more sophisticated objective functions such as computing the shortest constrained path for a new TE LSP while minimizing the number of TE LSPs to displace or to preempt (also referred to as the "minimum perturbation problem").

  The major challenges with Statefull PCE lies in the increased complexity and overhead required for state synchronization in the network, implying non-trivial modes of operation and signaling protocol overhead. For example, although a new path computation may be satisfied by displacing a wide set of TE LSPs, rerouting these TE LSPs prior to signaling the new TE LSPs might be a costly operation. Furthermore, should a failure occur in the network during the rerouting operation and prior to signaling the new TE LSPs, the statefull PCE may have to revisit the whole TE LSP displacement at the cost of re-signaling a number of TE LSPs, sending a large number of synchronized messages and potentially disrupting traffic while TE LSPs are being rerouted if a "make-before-break" mode of operation is not possible, or the displacement of these TE LSPs imposes to first re-signal these TE LSP with 0 bandwidth to avoid a dead-lock (furthermore, such operation may lead to temporary congestion on various links in the network).

Moreover, should the path computation requests be load balanced across a set of statefull PCEs, a specific mechanism becomes required for state synchronization between the statefull PCEs performing concurrent path computations. Yet another issue may arise if a statefull PCE rejects a path computation request because the required resources have been allocated to a previous request and the corresponding TE LSP is not successfully signaled. Such a situation can occur when statefull PCEs use an out-of-sync TED or the head-end LSR is not able of signaling the TE LSP.

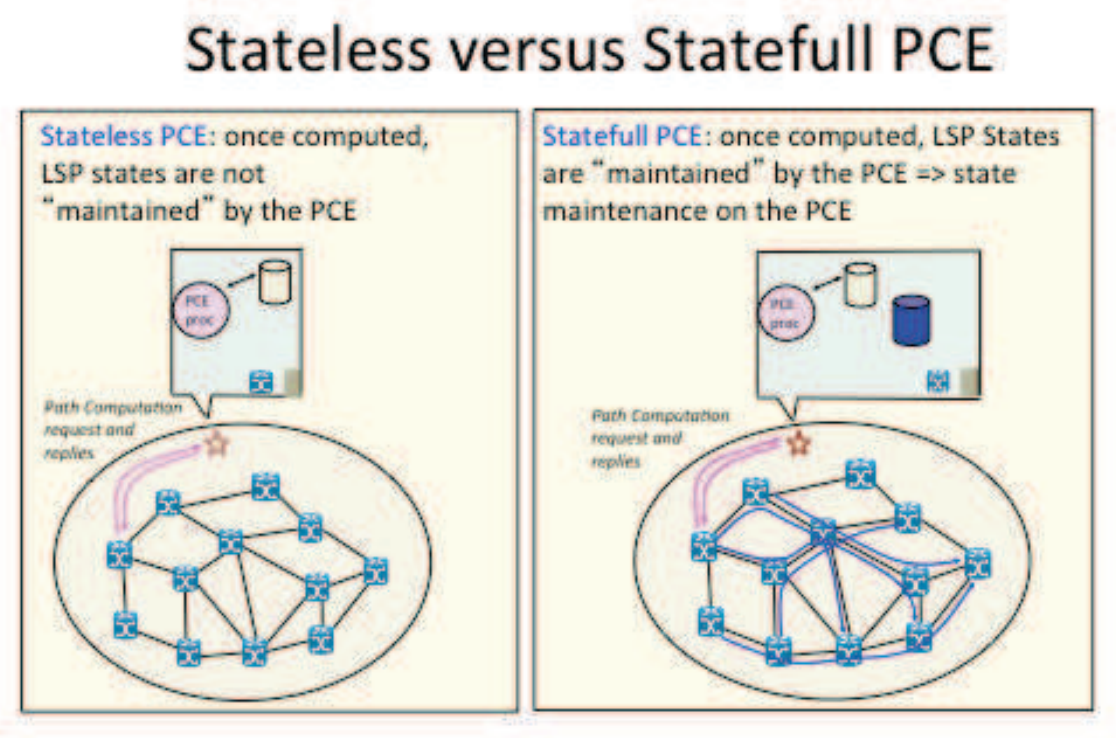


Figure 25 – *Statefull versus Stateless PCEs*

For the number of issues listed above, we decided to focus our research on protocols and algorithms for stateless PCEs, which, as of today, correspond to the ***vast*** majority of the deployments.

Back to the reference architecture, it is worth pointing out that the building blocks designed for stateless and statefull PCES are identical: the PCE discovery mechanisms, synchronization of the TED, signaling protocol (PCEP) (with minor addition for stateful PCE) are common to both modes of operation. Stateless and statefull PCEs raise different technical challenges though: most of the research and technical work of this thesis also apply to statefull PCEs.

2.1.2.6 Policy

The PCE may support a set of policy rules so as to potentially reject a request that should not be allowed. Such policy rules may make use of the TE LSP attributes

(destination of the TE LSP, nature of the TE LSP), the identity of the PCC ("back-listing") or the current state of the PCE (states of the queues, congested PCE), as discussed later in this thesis.

2.2 PCEP: A new signaling protocol for PCC-PCE and PCE-PCE communication

2.2.1 Introduction and problem scoping

Signaling is a core component of the PCE architecture and designing a protocol is a complex task that has major implications on the architecture, its performance and extensibility in order to address future requirements.

There are always a number of ways to design a protocol that require a deep understanding of the dynamics of deployed networks and requirements for the protocol to be designed in anticipation of future needs, which makes the task of designing the protocol even harder.

The first step consisted in studying the requirements of the PCC-PCE signaling exchanges in order to determine whether or not an existing signaling protocol could be extended with additional messages, objects and protocol TLV (Type-Length-Value).

The first approach in [36]* was to extend the RSVP protocol specified in [17] used in conjunction with [18] for fast retransmission in case of message loss. In this approach RSVP was used as a client-server signaling protocol without allocating any resource on intermediate hops according to the RSVP mode of operation; RSVP was extended with a new message type (called the "Path Computation Message") with a flag field identifying a request from a reply, and several new objects were used to identify the request and its attributes, the metric in use for path computation, the path cost, errors and/or reasons why a path satisfying the constraints could not to found, the requested bandwidth, number of requested TE LSPs, ability to exclude some network elements to mention a few.

Once the IETF Path Computation Element (see [37]*) Working Group was formed in 2004, the main task of the Working Group was first to select or specify a signaling protocol for the PCE architecture. A number of avenues were explored after having spelled out the list of requirements for that protocol in order to address the number of uses cases for the PCE. Several options were listed including [36]* but also the use of other protocols that could have been extended to that end such as [34] or [35].

The process of determining whether to extend an existing protocol or specifying a new protocol in order to address new requirement is a recurring process and took place a number of times in the history of the Internet, each approach having its own pros and cons. In the present case, we elected to design a new protocol, which turned out to also be the consensus at the IETF. The new protocol developed during the course of this thesis was inspired from [36]* in terms of specification of new

objects, which led to the PCEP protocol (a core component of this thesis) and described hereafter. PCEP has now been adopted as a full standard by the Internet Engineering Task Force and specified in [38]* (developed according to the requirements spelled out in [39]).

PCEP is a signaling protocol operating over the transport protocol TCP (Transmission Control Protocol, a transport protocol designed in 1981, which has then been augmented with a number of additions, and provides reliable transport between two hosts in addition to flow control mechanism).

PCEP is used between a Path Computation Client (PCC), the requestor of the path computation request (typically an LSR) and the Path Computation Element, the path computation engine, which (as discussed in the previous section) could be a process running on an LSR or an off-line server reachable via the IP/MPLS network. Note that a node may both act as a PCE when answering path computation request and a PCC when generating such request. In some examples such as the BRPC algorithm specified in chapter 3, several PCEs may be involved in a path computation chain consequently both acting as PCC and PCE during the process of computing a TE LSP.

The PCEP protocol has been designed to be highly extensible, allowing for the addition of new messages and objects in order to address future needs and requirements. Since its inception, a number of new messages and objects have been defined thus further extending the PCE applicability scope.

The requirements for the PCE signaling protocol that led to the specification of PCEP can be found in [39] (generic requirements), [40], [41], and [42].

The aim of this section is to summarize the characteristics and capabilities of the PCEP signaling protocol designed during the course of our research work, without providing all details of the protocol. The PCEP protocol details can be found in [34], which specifies the packet formats, protocol messages and variables. In the following section, we rather describe the design choices, protocol capabilities and the Finite State Machine of the protocol.

2.2.2 PCEP protocol messages

PCEP specifies several messages in [38]* (additional messages have been specified to extend PCEP in other specification such as [43]*) and a high-level message flow is shown in figure 12 as part of the protocol model.

The core PCEP messages are as follows:

- The Open and Keepalives messages allows for initiating and maintaining PCEP sessions between a PCC and a PCE,
- The PCReq message is sent by a PCC to a PCE to request the computation of a TE LSP (or a set of TE LSPs), along with attributes characterizing the request,
- Upon receiving a PCReq message, a PCE replies with a PCRep message that can either be positive in which case the set of computed path(s) along with

their characteristics are provided, or negative (no path computation could be performed and the PCE may provide more indication of the reasons why the path computation failed),

- PCEP also specifies a notification message called a PCNtf message that is sent either by a PCC to the PCE or a PCE to a PCC to notify of a specific event,
- The PCErr message sent upon the occurrence of a protocol error,
- The Close message used to close a PCEP session between two PCEP peers.

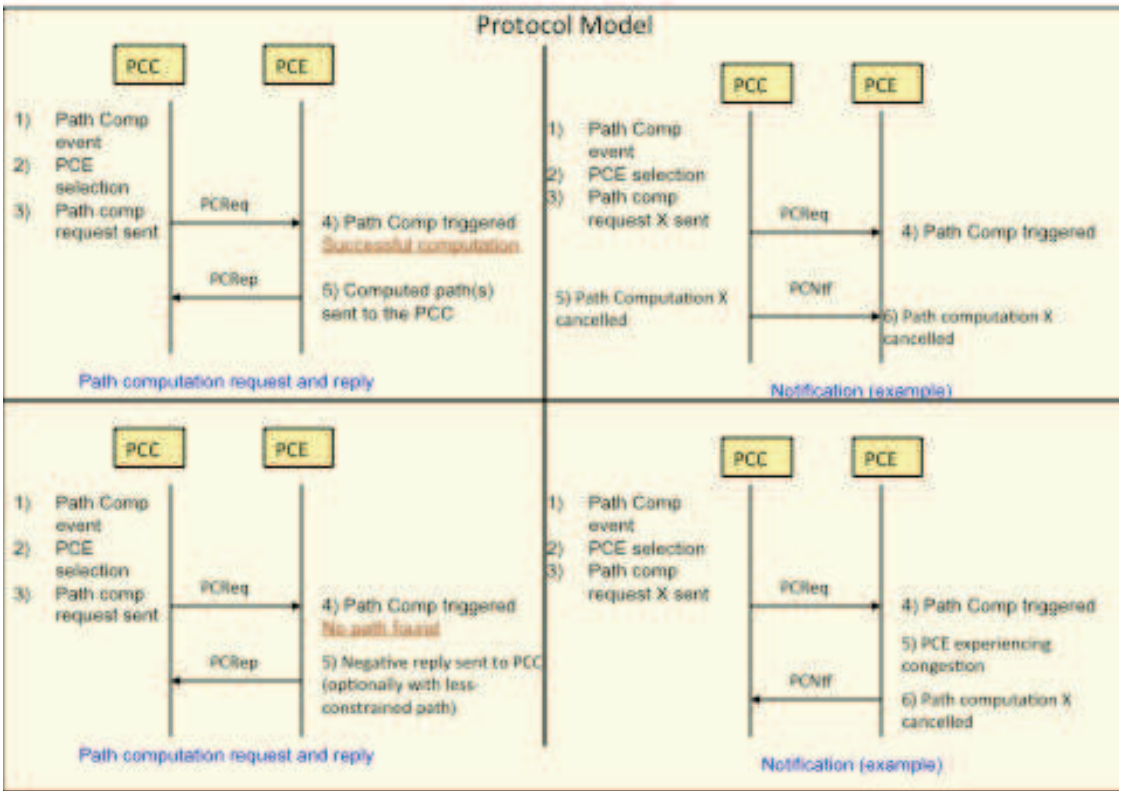


Figure 26 – *PCEP messages flows*

2.2.3 The PCEP Open and Keepalive messages and the Initialization Phase

In order for two PCEP peers to establish a PCEP session, they must first establish a TCP session using the well-known TCP 3-way handshake procedure followed by the establishment of the PCEP session itself. During the PCEP session establishment, several parameters are dynamically exchanged and negotiated thanks to the Open message, along with other parameters used to maintain the PCEP session using the Keepalive messages.

Note that the PCEP session establishment may fail (for example because the PCEP peers may not agree on the session parameters or simply because one of the peers does not answer); the requesting PCEP peer is allowed to retry after the expiration of a timer preferably using an exponential back-off session establishment procedure timer.

The Open message is used to establish the PCEP session and comprises PCEP session attributes such as session ID (Identity) the Keepalive and DeadTimer.

- The Keepalive is defined as the maximum period of time that can elapse between two consecutive PCEP messages sent by the sender of the corresponding Open message and can be set to 0, in which case no Keepalive are exchanged between the PCEP peers. Note that TCP provides its own Keepalive mechanism but was considered as not sufficient since a PCEP process failure may not be detected,

- The DeadTimer is the amount of time in seconds after the expiration of which the PCEP peer declares that the PCEP session is down if no PCEP message has been received. Note that timers are reset upon the reception of any PCEP message (not just Keepalive messages).

PCEP allows for dynamic negotiation of PCEP session attributes whereby a PCEP peer not agreeing with some PCEP session attributes has the ability to "suggest" alternative PCEP session attributes included in another Open message.

There must only be one PCEP session between two PCEP peers and the attributes may be unbalanced in terms of Keepalive traffic and respective timers.

PCEP sessions may either be permanent or intermittent (in which case the PCC establishes a PCEP session each time a path computation is required); the mode of operation is policy-based according to the circumstances in which the PCE is used such as the frequency of path computation requests, number of active sessions required for a given PCE in light of its own capability along with other considerations.

2.2.4 The PCEP Path Computation Request message (PCReq)

The PCReq message is the PCEP path computation message sent by a PCC (or a PCE acting as a PCC in a specific context such as the Backward Recursive Path Computation (BRPC) algorithm specified in chapter 3) to a PCE to request a path computation.

Figure 12 shows two message flows leading to a positive and negative reply respectively, whereby:

- An event at the PCC triggers a path computation (e.g. configuration of a new TE LSP on an LSR acting as a head-end LSR, reoptimization of the path for an existing TE LSP, …),
- PCCs and PCEs exchange PCEP notification messages,

- The PCReq message that provides a number of characteristics for the requested path and the request itself (described hereafter) is sent to the (dynamically or statically) selected PCE.
- A (positive or negative) path computation reply in the form of a Path Computation Reply (PCRep) message is returned by the PCE to the PCC.

PCEP messages comprise a common header used to specify several parameters in the forms of flags, followed by a variable length body made of a set of (mandatory or optional) objects specified in [38]*. Similarly to many other protocols, we use the Reduced Backus-Naur Form called the (RBNF) specified in [44].

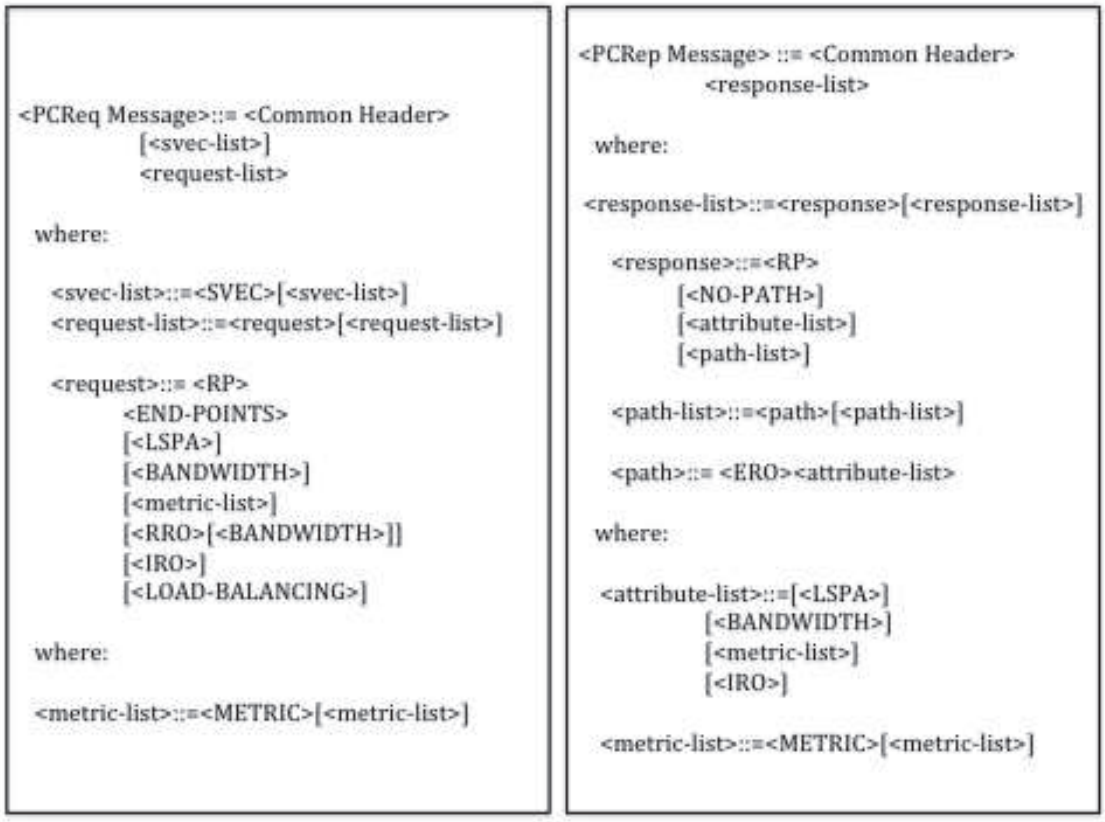


Figure 27 – *Format of the PCEP PCReq and PCRep messages in Reduced Backus Form*

Figure 13 shows the generic format of the PCReq and PCRep message; without describing in details the set of objects carried within these messages in addition to their flags, field, and processing rules, it is worth providing the functionalities supporting by these messages from a functional standpoint. This information can be used as algorithmic inputs. We also describe the overall PCE-based path computation handling and network behavior in such an architecture.

All of the PCEP objects may include TLVs that may either be mandatory for object processing or optional. A common header object is used to uniquely identify the

object type and class; a flag is used to indicate whether the object processing is mandatory or optional (typically used by the PCE to meet a nice-to-have requirement). A second flag is used by a PCE to indicate when an optional object was ignored (or not) during the path computation.

As shown in figure 13, a single path computation request may be made of a list of correlated or independent path computation requests where each request is itself characterized by a number of parameters such as:

- Request parameters: this mandatory object is used to indicate to the PCE the request ID (identification) number, the request priority (there are up to 7 priority levels) used by the PCC to indicate the degree of priority/urgency of the request and the PCE scheduler to assign the request to the proper priority queue according to the scheduling algorithm (preemptive queue, strict priority queue, round robin) or to indicate to the PCC that the PCE does not support multi-priority scheduling (which can be used by the PCC by the PCE selection algorithm), whether the request is a new request of a request for reoptimizing an existing TE LSP in which case the existing path must be provided if the PCE is stateless, whether the TE LSP is unidirectional or bidirectional and whether a loose path can be return or the PCC requests strict paths only (this is further discussed hereafter),
- The source and destination of the TE LSP,
- The LSP Attributes (LSPA) that provides the holding and setting priorities of the TE LSP but also a series of flag such as the L flag that indicates whether or not Local Protection with FRR is required: in this case, the PCE must find a path comprising links that are all protected by local protection technique such as FRR, which can be dynamically determined thanks to IGP extensions such as [45]*,
- Bandwidth of the requested TE LSP,
- A list of metrics: the PCC has the ability to provide the metric that must be optimized by the path computation engine (IGP versus Traffic Engineering metrics or even the hop-count metric) but also some bounded value for the said metric used by the PCE to determine whether or not the most optimum path can be considered as acceptable by the PCC, and finally whether or not the path cost must be provided in the returned computed path included in the PCRep message. Note the PCEP allows for inserting more than one metric object. If both metrics are used as optimization metrics, the PCE algorithm must be able to handle multi-metric optimization problem (known as NP-Complete). It is also possible to combine metrics used for optimization of the computed path and upper bound specifying the maximum path cost for a path to be considered as acceptable by the PCC,
- The RRO (Route Record Object) is used to specify the existing path followed by a TE LSP for which a reoptimization request is made,
- The IRO (Include Route Object) is used to enforce another constraint: when present, the IRO specifies that the TE LSP must traverse a set of network elements that can be an IP prefix, an interface ID or an Autonomous System,

- SVEC object: the research on PCE-based path computation models showed that the ability to synchronize path computation requests was indeed a must have in several scenarios.

  We thus decided to specify the notions of Dependent and Synchronized path requests (that have been implemented in the PCEP protocol thanks to the SVEC object).

  A set of path computation requests are said to be **independent** when they are not related to each other by contrast with **dependent** requests (e.g. computation of a set of diversely routed TE LSPs).

  The second notion relates to the **synchronization** of path computation requests in which case the path computation request cannot be serialized.

  For example the PCC may send a set of *N* path computation requests with M<N requests that should be considered as *M* synchronized path computation requests (as opposed to being simply serialized). Synchronizing a set of *M* path computation requests could help finding a more optimal TE LSP placement or even increasing the probability of finding a solution and consequently a positive reply for a subset of K<M<N path computation requests. This is the case when the network gets congested in terms of bandwidth booking and displacing a request *i* to follow a longer path still satisfying the constraints in terms of bandwidth, bounded delay, … may allow for finding a placement for a synchronized request *j* that could not have been satisfied if both requests *i* and *j* had been processed in a independent and serialized fashion (lthough both requests are not strictly speaking "correlated" as in the case of a set of diversely routed paths).

  Another example is when a stateless PCE may synchronize path computation requests in order to avoid double resource allocations to a pair of TE LSPs, in which case, the PCE becomes statefull for a bounded period of time equal to the period of time required to process the set *M* of synchronized path computation requests.

  Note that if a set of *N* dependent path computation requests are always synchronized the opposite is not true: a set of independent path computation requests may or may not be synchronized.

  In summary, thanks to the SVEC object, the PCEP signaling protocol allows for the support for three models:
    - The bundle of a set of independent non-synchronized path computation requests (which reduces protocol overhead by avoiding to send a set of PCReq messages),
    - The bundle of a set of independent and synchronized path computation requests so as to improve the placement of a set

of TE LSP, reduce the probability to provide a positive reply or even to reduce the probability of call set up failures,

- The bundle of a set of dependent and therefore synchronized path computation requests (e.g. computation of a set of diversely routed TE LSPs).

- Load balancing: there are circumstances where there is no path available in the network for a single TE LSP of bandwidth *B*; in some cases, it might be possible to defragment the bandwidth in the network thanks to some of the algorithms described in Chapter 5 resulting in displacing some TE LSPs either gracefully or thanks to (soft) preemption but there are circumstances where the only available option consists in making use of a set T of TE LSPs so that the sum of their bandwidth is equal to *B*.

  PCEP supports the ability for the PCC to signal the total amount of requested bandwidth B, in addition to the maximum number *T_Max* of TE LSPs such that the sum of their bandwidth is equal to *B*, and the minimum bandwidth for each TE LSP of the set *T*. Note that this last parameter is of the utmost importance. Indeed, should the set of TE LSPs be used to carry primary traffic, the head-end LSR will have to perform some hashing function in order to dynamically determine which TE LSP to use on a per-packet/flow basis; a hashing algorithm is used to ensure that packets belonging to the same traffic flow do not get load balanced across a set of TE LSPs that may provide significantly differing delays in which case packets of the said flow may be delivered in an out-of-order fashion, a very undesirable effect especially when (but not limited to) traffic flow make use of transport protocols not capable of packet re-ordering. A second example arises when TE LSPs are used for bandwidth protection: as extensively discussed in chapter 5, if a TE LSP used as a backup tunnel has a bandwidth less than the minimal bandwidth of any primary bandwidth protected primary TE LSP on the head-end LSR, it would become non usable.

The motivation for describing the set of objects carried within a PCRep message was to highlight the degree of flexibility provided by the PCEP protocol that was determined as mandatory during the course of this research.

2.2.5 The PCEP Path Computation Reply message (PCRep)

The PCEP PCRep message is sent by a PCE to a PCC in response to previously received path computation requests, which may have been synchronized and received as bundled in a single PCReq message or not. In other words, a PCE may also decide to group the computation of unsynchronized path computation requests received by means of multiple PCReq messages, at the extra cost of incurring additional delays for some requests of the set. Conversely, the PCE may send path computation replies using multiple PCRep messages for path computation requests received within a single PCReq message. Note that a single PCRep message may comprise a set of positive and negative replies.

Two situations may take place:

- The (bundle) path computation request(s) can be satisfied: in this case, the PCE returns to the requesting PCC one or more PCRep messages containing the set of computed paths. Note that in the case of multiple TE LSPs satisfying a single request such as when load balancing is required because a single path satisfying the constraints cannot be found, an additional object indicating the bandwidth is added to the computed path itself (which is encoded using an Explicit Route Object (ERO)).

- The path computation request cannot be satisfied: there are several circumstances in which a path computation cannot be satisfied: the PCE may not be operational or simply no path satisfying the constraints could be found. This later case is notified thanks to the presence of a NO-PATH object in the PCReq message that provides the reasons for a negative reply but also optionally the list of constraints that could not be satisfied (bandwidth, affinities, …). An interesting option consists for the PCE of sending a list of suggested values for which a path could have been found, if the PCE supports this capability (such an option unavoidably requires additional computation time). Thus, a negative reply may comprise a set of objects listing all of the constraints that could not be satisfied along with a list of suggested values (closest match) for which a solution could be found.
  Upon receiving a negative reply, the PCC may then decide to re-issue a path computation request after having adjusted the set of required constraints or select a new PCE chain.
  Note that the PCEP protocol does not support the ability to prioritize the set of constraints so as to provide a mean for the PCE to perform on-the-fly hierarchical constraint relaxation; this would allow the PCE to fast compute a TE LSP as close as possible to the original request without requiring the PCC to re-issue a second “adjusted” path computation request.

2.2.6 The PCEP Notification and Error messages

PCEP supports two other types of messages referred to as to the PCNtf (Notification) and PCErr (Error) messages used to report a notification and error respectively.

The PCNtf message can be sent either by a PCC to the PCE or by a PCE to a PCC, and may or may not be related to a specific pending path computation request.

For example, a PCC may notify a PCE that it wants to cancel a specific pending request (because it has received a positive reply from another PCE, should the PCC have sent the same request to multiple PCEs). Conversely, the PCE may decide to cancel a pending request because it experiences long processing delays or the PCE is simply congested. PCNtf messages may also be used by a PCE to notify a PCC of an overloaded state and may even indicate the estimated duration for which the PCE may be overloaded at the expiration of which further requests may be sent by the PCC. In absence of any overload state duration indication, a PCE having sent an

"overload" state notification must indicate when exiting from the congestion state. In all cases, a PCE must make use of a hysteresis approach using for example a dual threshold mechanism triggering notifications of overloaded states to avoid too frequent oscillations between PCEs serving the various path computation requests in the network. Should a PCE experience high instabilities, it is also expected to implement a linear or even an exponential dampening algorithm for overload notifications frequency pacing.

Furthermore, PCC receiving "overload" state notification should make use of back-off techniques to avoid a major shift of computation load on the newly selected PCE, should other PCCs have received the same overload notification triggering PCE re-selections.

The PCErr message is used to report a protocol error such as the receipt of an invalid message, a request requiring a capability non supported by the PCE, a request triggering a policy violation or simply because a mandatory object is missing in the PCReq message.

2.2.7 The PCEP Finite State Machine

In this section, we describe the Finite State Machine of the PCEP signaling protocol.

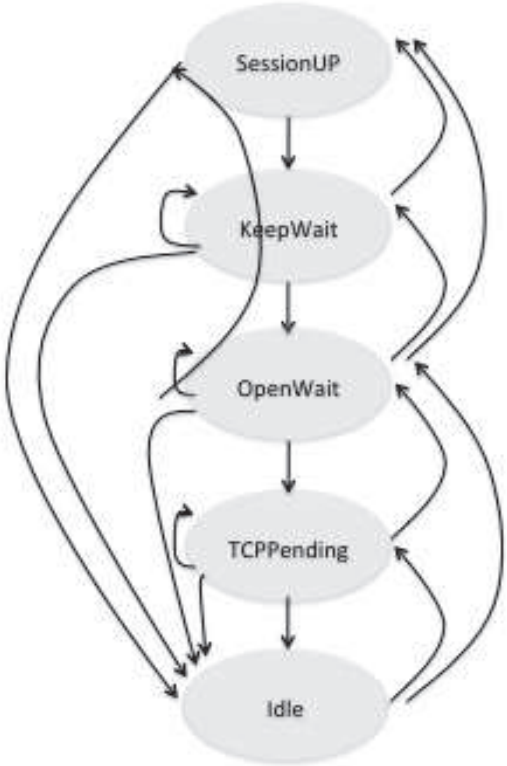


Figure 28 – ***PCEP Finite State Machine***

2.2.7.1 Set of Variables

We specify the following set of PCEP variables:

- *Connect*: Timers (in seconds) armed after the initialization of the TCP connection. The connect timer=60 seconds,
- *ConnectRetry*: Number of times the system has tried to establish a TCP connection with the PCEP peer without success,
- *ConnectMaxRetry*: maximum number of times a system tries to establish a TCP connection using the PCEP port before going back to Idle state (*ConnectMaxRetry*=5),
- *OpenWait*: amount of time a PCEP peer waits to receive an Open message. When the *OpenWait* timer expires the system goes back to Idle state (*OpenWait*=60s),
- *KeepWait*: amount of time a PCEP peer wait to receives a KeeAlive message (or any other PCEP message). When the *KeepWait* timer expires the system goes back to Idle state (*KeepWait*=60s),
- *OpenRetry*: number of the times the system has received an Opne message with unacceptable PCEP session characteristics,
- *RemoteOK* is a Boolean state set to 1 when a system has received an Open message with acceptable characteristics,
- *LocalOK* is a Boolean state set to 1 when a system has received a Keepalive message acknowledging that the PCEP peer has received an Open message with acceptable characteristics.

2.2.7.2 State Description and transitions

The PCEP FSM states and the state transitions showed in Figure 14 are also detailed hereafter:

- **Idle State**: in that state the system has allocated local resources but no PCEP session is currently active (the system is only listening to the PCEP well-known port).
    - *TCPRetry*==0
    - *LocalOK*==0
    - *RemoteOK*==0
    - *OpenRetry*==0

  Upon detecting a local event triggering the initialization of the PCEP session with a (dynamically discovered or statically configured) PCEP peer:
    - A TCP connection is initialized
    - The Connect Timer is started
    - The System moves to the *TCPPending* state

  Upon receiving a TCP connection (on the well-known PCEP port), if the TCP connection establishment succeeds:

    - Sends an Open message with local PCEP session attributes
    - Stars the *OpenWait* timer

- The systems move to the *OpenWait* state

- ***TCPPending* State**: in that state the systems tries to establish the TCP session
  - If the TCP connection established succeeds:
    - Sends an Open message with local PCEP session attributes
    - Stars the *OpenWait* timer
    - The systems move to the *OpenWait* state
  - If the TCP connection establishment fails or the Connect Timer expires:
    - If *ConnectRetry*=*ConnectMaxRetry*, the system moves to Idle state
    - If *ConnectRetry*<*ConnectMaxRetry*, the system initiates a TCP connection with the PCEP peer, increments the *ConnectRetry* variable, restart the *Connect* Timer.

- ***OpenWait* State**: In the *OpenWait* state, the system waits for an Open message from its PCEP peers (which whom the TCP connection has been successfully established).
  - If the system receives an Open message before the expiration of the *OpenWait* timer:
    - The system examines all of its sessions that are either in the *OpenWait* or *KeepWait* state. If another session with the same PCEP peer already exits and with the same IP address, then the system triggers a "collision" resolution procedure (if the system has initiated the current session and it has a lower IP address or if the session was initiated by the PCEP peer and the system has a higher address, PCEP resources are released and the system moves to the Idle state); otherwise the system checks the PCEP session attributes of the Open message:
      - If an error is detected, an error message is sent, the system release the PCEP resources and move to Idle state
      - If no errors are detected, *OpenRetry*=1 and the PCEP session characteristics are unacceptable, an error message is sent, the system releases the PCEP resources and moves to Idle state
      - If no errors are detected and the PCEP session characteristics are acceptable, then the system:
        - Sends a Keepalive
        - Starts the *Keepalive* timer
        - *RemoteOK*==1
          - If *LocalOK*=1, the system clears the *OpenWait* timer and moves to UP state

            - If *LocalOK*=0, the systems clears the *OpenWait* timer, starts the *KeepWait* timer and moves to the *KeepWait* state.
        - If no errors are detected and the PCEP session characteristics are unacceptable and non negotiable, an error message is sent, the system releases the PCEP resources and moves to the Idle state.
        - If no errors are detected and *OpenRetry*=0 and the PCEP session characteristics are unacceptable but negotiable, an error message is sent, then the system performs the following set of actions:
            - *OpenRetry*++
            - An error message is sent that proposes new acceptable values
            - If *LocalOK*=1, the system restarts the *OpenWait* timer and stays in the *OpenWait* state
            - If *LocalOK*=0, the system clears the *OpenWait* timer, starts the *KeepWait* timer and moves to *Keepwait* state
    - If the system does not receive an Open message before the expiration of the *OpenWait* timer, the system, sends an error message, releases the PCEP resources, closes the TCP session and moves back to the *Idle* state.

- ***KeepWait* State**: in the *KeepWait* state the system waits for the reception of the Keepalive message from its PCEP peer acknowledging its Open message, or an error message if the PCEP cannot accept the proposed PCEP session characteristics:
    - If an error is detected such as a malformed KeepAlive message, the system sends an error message, releases the PCEP resources, closes the TCP connection and moves to the Idle state.
    - If a KeepAlive message is received before the expiration of the *KeepWait* timer, *LocalOK*==1 and:
        - If *RemoteOK*==1, the system clears the *KeepWait* timer and moves to the UP state,
        - If *RemoteOK*==0, the system clears the *KeepWait* timer, starts the *OpenWait* timer and moves to the *OpenWait* state.
    - If an error message is received before the expiration of the *KeepWait* timer:
        - If the proposes values are unacceptable, an error message is sent, the PCEP resources are releases, the TCP connection is closed and the system moves to the Idle state.
        - If the proposed values are acceptable, the system adjusts is PCEP sessions accordingly, restarts the *KeepWait* timer, and sends a new Open message.

        - If *RemoteOK*==1, the system restarts the *KeepWait* timer and stays in the *KeepWait* state
        - If *RemoteOK*==0, the systems clears the *KeepWait* timer, starts the *OpenWait* timer and moves to the *OpenWait* state.
    - If no Keepalive message or no error message are received before he expiration of the *KeepWait* timer, the system sends an error message, releases the PCEP resources, closes the TCP connection and moves to the Idle state.

- ***UP* State**: In this state, PCEP peers exchange PCEP messages and the PCEP session is fully operational.
    - If the *Keepalive* timer expires (in the case where Keepalive messages are exchanged) the systems restarts the Keepalive messages and sends a Keepalive message
    - If no PCEP message (Keepalive, PCReq, PCRep, PCNtf) is received before the expiration of the *DeadTimer* timer, the systems terminates the PCEP session, releases the PCEP resources, closes the TCP connection and moves to the Idle state.
    - If a malformed message is received, the system terminates the PCEP session, releases the PCEP resources, closes the TCP connection and moves to the *Idle* state.
    - If the systems detects that the PCEP peer tries to initializes the second TCP connection, that TCP connection attempt is terminated and an error message is sent to the PCEP peer.
    - In any case of TCP connection failure, the system releases the PCEP resources, closes the TCP connection and moves to the *Idle* state.

2.3 The PCE Monitoring extension

During the course of this research it quickly became apparent that monitoring tools were required, especially for troubleshooting and performance monitoring for example in the case of the PCE chain involving a set of collaborative PCEs (such as with the BRPC algorithm discussed in chapter 3). Indeed, when multiple PCEs involved in the path computation reside in different routing domains, the PCC cannot rely on the IGP to gather information about all PCEs and thus suffers from partial visibility of the PCE chain state.

To that end, as part of this research, we extended the PCEP protocol to support new messages types in order to gather PCE state metrics, which may be as simple as a Boolean (e.g. the PCE is alive, or overloaded) or a more complex expression to report performance metrics of various types.

We specified in [46]* PCEP protocol extensions: two new messages types called PCMonReq and PCMonRep, in addition to new objects and mechanisms by which PCE state metrics can be gathered. The protocol extensions being straightforward

we will briefly focus on the functional capabilities on these protocol extensions in this chapter (more details on the protocol extensions can be found in [46]*).

We propose two mode operations: *in-band* (the state metric is gathered in the context of a path computation request – for example, the PCC requires to also gather the processing time of that request) or *out-of-band* (the state metric is gathered as a standalone request such as the averaged processing time over the past *X* requests or *T* minutes). Furthermore a monitoring request may be either specific (relates to a specific path computation request) or general (gathering of PCE state metrics not coupled to any particular path computation requests).

A wide range of PCE state metrics specified in [46]* can be gathered on either a single PCE or along a PCE chain such as the current/average/min/max/variance computation times, the state of the PCE (alive, congested, …) either in the context of a specific path computation request or out-of-band in order to check the health and performance of a PCE path computation chain.

2.4 Conclusion

In this chapter, we have described the functional blocks of the PCE architecture that was designed during years of research and led to a radically different model for the computation of TE LSPs. In this architecture, PCC sends requests to (dynamically discovered) stateless or statefull PCEs, that in turn trigger path computation of a set of synchronized and/or correlated path computation requests in order to solve a specific problem according to well-defined objective functions before returning (when successful) a set of computed paths.

To that end, in addition to specifying the PCE architecture and its functional blocks, we have specified various protocol extensions and a new signaling protocol called PCEP, now recognized as an international standard and that is deployed in several networks. PCEP has the property to support sophisticated request and reply modes of operations while being widely extensible and flexible. This chapter provided an overview of the protocol in addition to the detailed Finite State Machine.

In the following chapters, we will explore several new algorithms based on this architecture that have been designed to solve several of the technical problems highlighted in chapter 1 that are now solvable thanks to this new architecture, and a set of new protocols and algorithms.

3 Using a backward recursive algorithm to compute optimum constrained shortest inter-domain TE LSP

One of the key objectives of this thesis was to specify a new path computation model for inter-domain TE LSPs (and other types of non-MPLS based tunnels). Our proposal called the BRPC (Backward Recursive Path Computation) has been patented and standardized in [47]*.

3.1 The BRPC algorithm

As discussed in chapter 1.4.1., the per-domain path computation approach was the only available path computation technique prior to the emergence of the PCE architecture. In this chapter we first describe the new BRPC algorithm and then provide a performance analysis via discrete event simulation comparing the two path computation techniques.

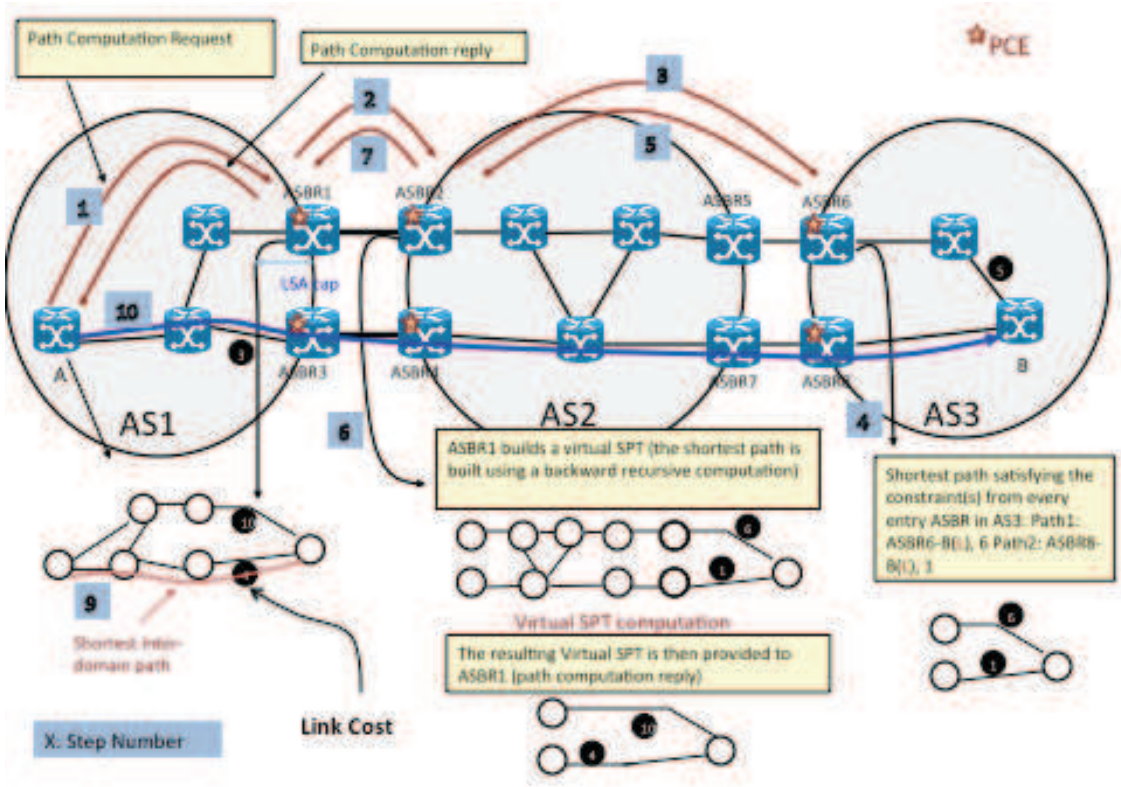


Figure 29 – *The BRPC algorithm for inter-AS TE LSP path computation using collaborative stateless PCEs*

The BRPC algorithm applies to both inter-area and inter-AS and will be described in the case of inter-AS TE LSP, a superset of inter-area TE LSPs.

BRPC is an algorithm involving a chain of collaborative PCEs that recursively compute the shortest constrained inter-domain TE LSP thanks to a per-domain computed Virtual Shortest Path Tree (VSPT) that is signaled in the backward direction according to the algorithm below. The VSPT is a tree rooted at the TE LSP destination when each branch of the tree is a link that virtually represents the shortest constrained path between the TE LSP destination and an entry ASBR of the domain the computing PCE is responsible for. The VSPT is made of the set of virtual links from every entry A(S)BR to the TE LSP destination.

Terminology:

AS(p): *p* is the index in the set of p traversed AS from source to destination.

Boundary Node (BN): either an ASBR (inter-domain) or ABR (inter-area).

BN-en(k): Set of Boundary Node Entry for domain k (e.g *ASBR2* for *AS2*).

BN-ex(k): Set of Boundary Node Exit for domain k (e.g. *ASBR1* for *AS1*).

BN-en(k,i): *k* is the index in the set of *X-en(i)* entry BNs for the domain *i*.

BN-ex(k,j): *k* is the index in the set of *X-ex(j)* exit BNs for the domain *j*.

PCE(i): PCE in charge of *ASi* (e.g. ASBR6 and ASBR8 for AS3).

VSPT(i): VSPT computed by *PCE(i)* and returned to *PCE(i-1),* a Point-to-Multipoint tree where each link is rooted at the TE LSP Destination and represents the shortest constraint paths from the root to each *BN-en(k,i)* where |*X-en(i)*|] is the number of entry BNs in domain *i* to the next hop domain and $k \leq |X\text{-}en(i)|$.

Description of the BRPC algorithm

Step-1 (determination of local computation or selection of a next hop PCE)

The head-end TE LSP (e.g *A*) first determines if the TE LSP destination *B* resides in the same domain thanks to a lookup in its routing table populated by its IGP (and not BGP).

If the destination IP address is found in the IGP Routing Table ***then*** the head-end performs a local path computation or makes use of an intra-domain PCE discovered thanks to IGP discovery

Else

- Find the next-hop domain by inspecting the AS-path computed by BGP
- Determine the set of *PCE(k)* advertising path computation capabilities to compute the TE LSP terminating on a node residing in *AS(n)*

If no *PCE(k)* is found ***then*** stop and log a local error

Else selects a *PCE(t)* in the set of candidates *PCE(k)* according to one of the PCE selection algorithms described in chapter 4.

End.

Upon receiving a Path computation request from a downstream PCE, the receiving *PCE(i)* triggers the following algorithm:

Determine if the TE LSP destination resides in *AS(i)* (*i*==*n*).

If the TE LSP destination resides in ASi (i==n) ***then*** call the *VSPT_calc* function and return *VSPT(n)* to either the requesting head-end LSR (e.g *A*) or *PCE(i-1).*

Else (relaying of PCEP path computation request)

- (optional) call the *Conc_path_calc()* function as defined later in this section and ***if*** the function returns a positive result (at least one path satisfying the constraints exists in *AS(i)*) ***then***
    - Find the next-hop domain *i+1* by inspecting the AS-path computed by BGP
    - Determine the set of *PCE(k)* advertising path computation capability to compute TE LSP destined for a node residing in *AS(n)*

    Else

    Return a negative reply

    End if

If no *PCE(k)* is found ***then*** stop and return a path computation error (PCEP error message)

Else

- Select a *PCE(t)* in the set of candidates *PCE(k)* according to one of the PCE selection algorithms described later in this section.
- (optional) Call the *Conc_path_calc* function
- Relay the path computation request (PCEP message) to *PCE(t)*

End

End

Upon receiving a PCE Path computation reply from a upstream PCE, the receiving *PCE(i)* triggers the following algorithm:

If i==1 (the first invoked PCE is also responsible for the domain where the TE LSP head-end resides) ***then***

- Concatenate the returned *VSPT(i+1)* to the local TED for *AS(1)*
- Compute the shortest path between the origin and destination using the concatenated TEDs of *AS(0)* and *AS(1)*
- Return shortest path(s) to the PCC (head-end LSR)

Else

- Extract the returned *VPST(i+1)*
- Concatenate the returned *VSPT(i+1)* to the local *TED(i)* (or the set of virtual links computed locally by the Conc-path-calc function) for *AS(i)*
- Compute *VSPT(i)* on the resulting topology (note that the links retrieved from *TED(i)* are actual link whereas links from *VSPT(i+1)* are shortest constrained path from downstream domains), calling the *VSPT_calc*() function.
- Return *VPST(i)* to *PCE(i-1)*

End

The *VSPT_calc()* function:

The *VSPT_calc()* function called by a *PCE(i)* is responsible for computing *VSPT(i)*, a Point to Multipoint tree where each branch is the shortest constrained path from the root (TE LSP destination) to each *BN-en(k,i)*. The function is agnostic to the path computation algorithm and a number of shortest path computation algorithms can be used such as CSPF.

The *VSPT_calc()* function has been further optimized to prune each inter-ASBR link that does not satisfy the constraints of the TE LSP (e.g. bandwidth, TE LSP attributes, …). To that end, the IGP Traffic Engineering advertisement has been enhanced for inter-ASBR links even though there is strictly speaking no routing adjacency between ASBRs. This optimization does not apply to the inter-area path computation case where there is no such inter-domain links (areas are directly connected via ABR).

For the sake of illustration, in the example depicted in Figure 6, *ASBR6* (the last PCE of the PCE path computation chain) computed *VSPT(3)*, a tree with two links: link_1 represents the shortest path between *ASBR6* and *B (BN(1,3)-B)* with a cost of 1+5, and *ASBR8* and *B* with a cost of "1". The optimization related to pruning inter-AS link that would not satisfy TE LSP constraint would for example consists in pruning the link *ASBR5-ASBR6* by *PCE(2)* (*ASBR2*) if it determines that this link cannot satisfy the constraints.

3.2 Path Optimality

One of the key characteristics and fundamental properties of the BRPC algorithm lies in that it guarantees to always find the shortest inter-domain constrained paths thanks to a collaborative PCE-based approaches without requiring for a single node in the network to have the full visibility of all traversed domains, which is known as

not being desired in most cases in existing networks for a number of reasons including confidentiality preservation.

Should a more optimal path segment become available in a remote domain, the PCC simply needs to rerun the BRPC algorithm, which can be triggered upon receiving a notification that such a better path exist or after the expiration of a local head-end based reoptimization timer.

3.3 Reoptimization of an Inter-Domain TE LSP

The ability to reoptimize an inter-domain TE LSP that was computed using the BRPC algorithm is fully supported thanks to the PCEP signaling protocol that allows a PCC requesting to re-compute an inter-domain TE LSP for providing the existing path and TE LSP attributes so as to avoid any bandwidth double counting.

3.4 Computation of inter-domain TE LSP across AS made of multiple areas

As shown in Figure 16, AS may themselves comprise multiple IGP areas. Still, the BRPC algorithm remains unchanged and would recursively compute the shortest path across IGP areas and AS with no change; the optimization related to encrypted paths and described in Section 3.7. only applies to inter-AS path computation.

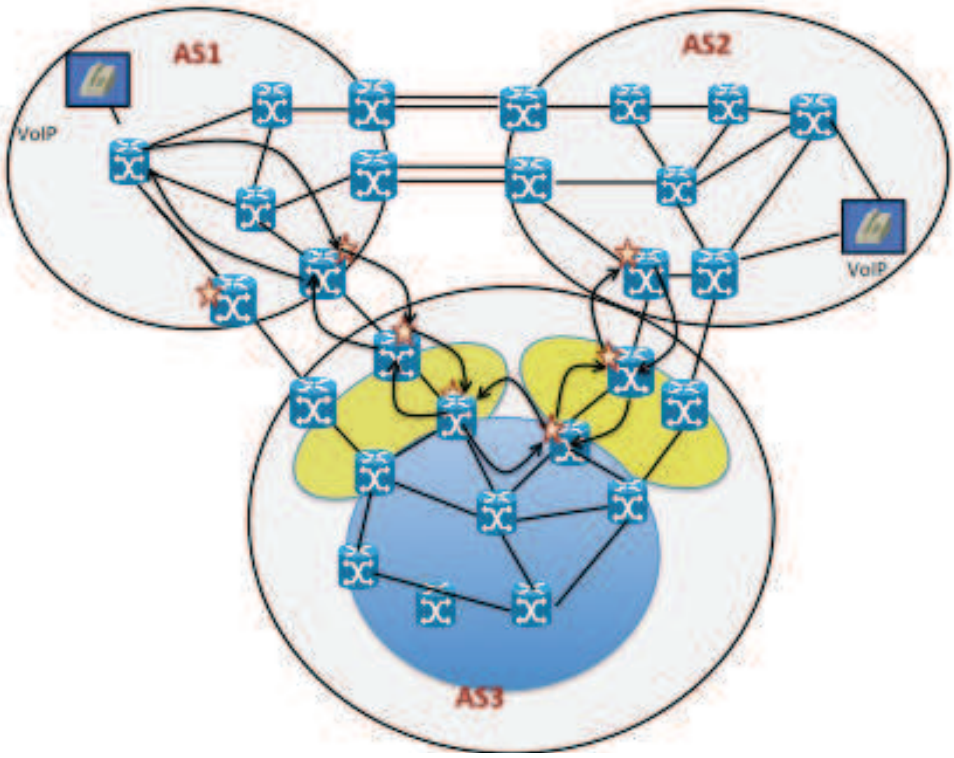


Figure 30 – *BRPC path computation across IGP areas and Autonomous Systems*

3.5 Extension to an arbitrary set of Autonomous Systems (AS)

Although IGP areas/levels are organized following a tree structure (for example with OSPF, all areas are connected to the back-bone area), the situation is different in the case of inter-AS TE LSP computation where there be an arbitrary number of meshed AS. In such a case, a set of AS paths may exist between two arbitrary LSRs, thus leading to more than one paths between two end-points, each path transiting through a different set of (potentially overlapping) AS.

We extended the BRPC algorithm to support the case of an arbitrary mesh of AS that may be traversed by a given TE LSP computed using the BRPC algorithm.

To that end, we extend the BGP protocol (inter-domain routing protocol): indeed, in its current mode of operation, BGP speakers select their preferred path towards a specific IP prefix according to local BGP policy; thus the head-end LSR would not be able to retrieve all sets of AS paths (where in this context a path is defined as a set of traversed AS by contrast with the TE LSP itself), but the preferred path according to the set of BGP speakers that make use of potentially different policy rules, and would thus not be able to discover the shortest constrained inter-AS TE LSP computed by the BRPC algorithm. Consequently, we extended the BGP protocol behavior so as to advertise all paths without making any selection according to BGP prefix selection policy rules in order for the head-end LSR to discover all possible set of traversed AS (BGP paths) from itself to the targeted TE LSP destination.

We call *BGP_Path* the set of BGP Paths (traversed AS); each element that is a superset of another element is then pruned from the *BGP_Path* set.

The following algorithm is then used to compute the shortest inter-AS TE LSP:

Step-1:

For each path P_i from H to T (Head-end H to Tail-end T) of the set BGP_Path ***then***

- Arm a timer T_{resp}
- Determine the next-hop PCE_j reachable by H (PCC) and capable of computing TE LSP destined to T
- Send a PCReq message to PCE_j, comprising a new object encoding the list of traversed AS P_i

End

Step-2:

For each received request comprising a BGP path P_i ***then*** follow the BRPC algorithm and compute a TE LSP T_n where $n \leq |BGP\_Path|$ if such a constrained path exists.

Step 3:

- Store each computed TE LSP received prior to the expiration of the timers T_{resp} along with the corresponding cost
- Select a number of *K* TE LSPs in the set of computed paths. *K*=1 means that one TE LSP will be used. If *K*>1, *H* may decide to signal a set of *K* TE LSP where $k \leq |BGP\_Path|$ that meet the constraints and such that the sum of their bandwidth is equal to the requested bandwidth (this may allow for load balancing among a set of (potentially diverse) TE LSPs).

Step-4:

- For each active TE LSP, send a path computation reoptimization request to the PCE used to compute the said TE LSP
- If a shortest path TE LSP is found, then the head-end LSR (PCC *H*) may decide to re-signal the TE LSP along the newly computed path if the path cost decrease is above some pre-defined threshold.

3.6 Concurrent path computations

Inter-domain path computation involves the collaboration of a set of PCEs, whereby each *PCE(i)* is responsible for the computation of a *VSPT(i)* computing the set of shortest constrained paths from each *BN-en(k,i)* to the TE LSP destination (root of the tree). As the PCE path computation request is relayed along the chain of PCEs involved in the path computation, each *PCE(k)* along the chain has to wait until it receives *VSPT(k+1)* before computing its *VSPT(k)*.

We introduce a further optimization with the aim of reducing even further the path computation time.

The core of this (optional) component consists in specifying the *Conc_path_calc()* function according which a PCE(i), after (or before) relaying the Path computation request to a downstream *PCE(i+1)* computes path segments in its domain prior to receiving *VSPT(i+1)*.

Conc_path_calc() function

> Compute the set of shortest constrained paths between each pair of *BN-en(k1,i)* for *k1*=1 to *BN-en(i)* and *BN-ex(k2,i)* for *k2*=1 to *BN-ex(i)*. The output is a set of virtual links where each link represents the shortest constrained path between each pair of entry-exit boundary node in the domain i.
>
> ***If*** no path can be found, ***then*** regardless of the received *VSPT(i+1)*, *PCE(i)* sends a PCEP error message to the upstream PCE that is relayed up to the requesting head-end to signal that no constrained path exist along the selected domain chain, and once *VSPT(i+1)* is received, it is silently discarded.

By concurrently computing shortest paths in each visited PCE along a PCE chain, the computation time of *VSPT(i)* at each hop triggered upon receiving VPST from downstream PCE is further reduced. Note that the gain is increased when using

CPU-intensive constrained shortest path computation (which is not the case of CSPF) that would for example require the computation of a set of diversely routed path between two disjoint pair of boundary nodes.

Note that alternatively the *Conc_path_calc()* function can be called by PCE(i) prior to relaying the path computation request to its selected downstream PCE in order to avoid the triggering of processing in downstream domains if no path can be found in domain AS(i). It must be noted that this would add delays in computing the overall path at the benefit of reducing signaling and downstream processing. The decision of calling the *Conc_path_calc()* function before or after relaying the path computation request can be driven by the observed signaling overhead and notification of congested downstream PCE.

Optionally the *Conc_path_calc()* function is called on a regular basis as a background task to compute the maximum shortest path between each pair of entry and exit boundary nodes in its domains. Upon receiving a path computation request, the TE LSP attributes are checked against the set of pre-computed maximum bandwidth links. If no path is found, the Path computation request is not further relayed and an PCEP error message is sent back to the upstream requesting PCE or the PCC without further processing. The process can be further enhanced by computing maximum bandwidth paths on a per set of TE LSP attributes basis.

3.7 Encryption of Path-key to protect confidentiality

Inter-domain TE LSPs are critical to provide bandwidth guaranteed paths across ASs, both IGP areas and Autonomous Systems, in addition to reliable paths protected with local protection mechanisms such as MPLS TE FRR. In the case of Inter-AS, if the Autonomous Systems belong to different Service Providers it is also required to protect confidentiality and thus the BRPC technique has been extended to preserve topology confidentiality, so as to "hide" the set of hops traversed within an AS thanks to the use of path-keys, another element of this thesis.

Instead of returning a set of path segments along with their respective costs between a set of Boundary routers according to the BRPC algorithm, we have designed a new mechanism consisting in returning a loose hop[1]. Using loose hops enables to hide the set of traversed LSRs of the computed path segment and to retrieve the actual path upon signaling the TE LSP as the RSVP Path message is processed by the entry node (e.g. ASBR). This innovative mechanism has been standardized in [48]*. According to this technique, a PCE does not return the list of hops of a path segment but only the first and last hop, along with a path-key that is used upon signaling the TE LSP in order for the boundary node to retrieve the actual path within its own domain (called a path segment). Thus, one can preserve AS-internal topology information from being disclosed.

[1] Note that the concept of "loose hop" is introduced in [8]

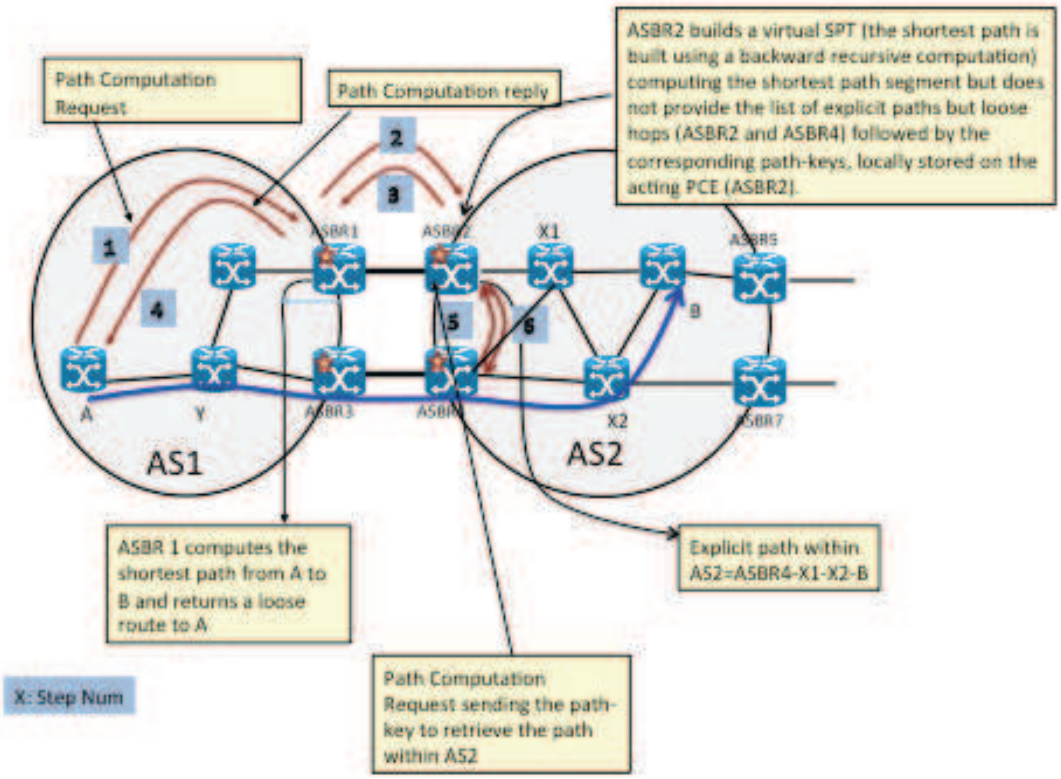


Figure 31 – *Use of Path-key to preserve confidentiality using the BRPC algorithm to compute shortest constrained inter-domain paths*

Algorithm

Step-1:

- The PCC (head-end of the inter-AS TE LSP) sends a path computation request (PCEP PCReq message) according to the BRPC procedure with a special flag, called the "O bit" of the RP Object carried in the PCReq message set (indicating that the PCC does not request a strict path so as to preserve confidentiality).

Step-2:

- The BRPC algorithm is triggered with the following modification: when building the VSPT from each <*BN-en*,*BN-ex*> routers pair, each computing PCE provides the respective path segment cost, the BN-en and BN-ex LSR address flagged as loose hops, the identity of the computing PCE and a path-key (a scalar): the actual set of traversed hops for the corresponding path segment within the domain remains undisclosed.
- Each computing PCE caches the path-key along with the explicit path segment (set of traversed hops) for a specific time (a recommended value is 60 seconds).

Step-3:

- After having received the shortest constrained inter-AS TE LSP (with loose hops), the head-end LSR initiates the signaling of the inter-domain TE LSP, but modifies the ERO (Explicit Route Object) of the RSVP-TE message (according to [16]) by listing the boundary nodes and a set of newly defined objects called the PKS and standardized in [44]* that comprise the computing PCEs and the corresponding path keys for each "hidden" path segment,
- When processing the ERO, a boundary node extracts the PKS object, and sends a path computation request to the PCE listed in the PKS object along with the corresponding path-key (the ASBR acts as a PCC),
- Upon receiving the path computation reply with the list of traversed LSRs in the said domain, the signaling procedure continues after adding the list of traversed hops in the ERO within the domain.

Example (referring to the network depicted in Figure 17)

- The head-end LSR *A* sends a path computation request to *ASBR-1* in his domain *AS1* (automatically discovered by the IGP),
- *ASBR1* relays the request to the next-hop PCE of the dynamically discovered PCE path computation chain according to the BRPC algorithm (*ASBR2*),
- *ASBR2* computes the two shortest constrained path segments from each entry *BN-en* LSR in its domain, namely *ASBR2* and *ASBR4* and the TE LSP destination B and returns the following VSPT:
    - Path Segment 1 == <*ASBR2-PKS1-B*>, path_cost (path_segment_1)
    - Path Segment 2 == <*ASBR4-PKS2-B*>, path_cost (path_segment_2)

  We make the assumption that there is a path segment satisfying the constraints from *ASBR2* and *B*, and from *ASBR4* and B.

  Both PKS are locally cached by *ASBR2* for 60s.

- *ASBR1* concatenates its local topology information related to *AS1* with both path segments augmented with the inter-AS link information and computes the shortest path from *A* to *B* according to the BRPC algorithm.
- If we make the assumption that the shortest constrained end-to-end path from *A* to *B* transits via *ASBR3* and *ASBR4*, the TE LSP is signaled with loose hops and the returned PKS2. ERO == <*A-Y-ASBR3-ASBR4*(loose)-PKS2-*B*>
- Upon processing the signaling control message (RSVP Path message), *ASBR4* extracts the path-key PKS2 and the ID of the computing PCE (*ASBR2*),
- *ASBR4* acting as a PCC sends a path computation request including the path-key to *ASBR2* in order to retrieve the path segment in *AS2*,
- *ASBR2* retrieves the related cached computed path segment (*ASBR4-X2-B*) and returns it to the request PCC (*ASBR4*).
- *ASBR4* recalculates the ERO == <*ASBR4-X2-B*> and the signaling of the TE LSP continues.

3.8 Performance analysis via simulation of the BRPC multi-PCE collaborative approach for inter-domain TE LSP computation

In this section, we provide simulations results (published in [49]*) comparing several performance metrics for the per-domain path computation and the multi-PCE collaborative approach (BRPC) so as to evaluate the performance of the PCE-based algorithm specified in this thesis.

To that end, two networks have been used to perform these simulations in order to avoid any results that would be topology-dependent. Both networks are representative of real-life networks and their topology is shown on figure 18 and 19.

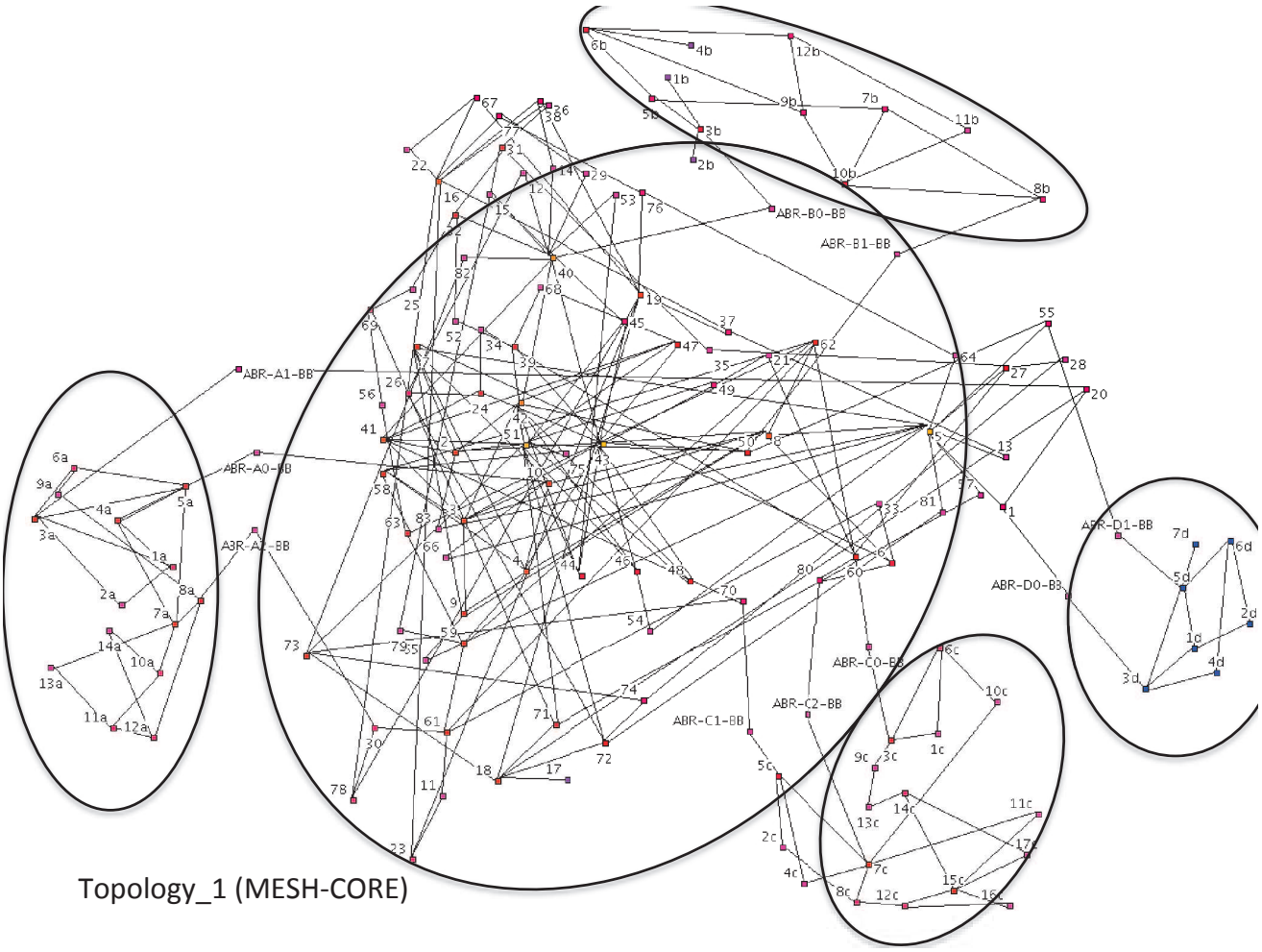


Figure 32 – ***Network topology 1 (MESH-CORE) used for performance metric analysis using an event discrete simulator***

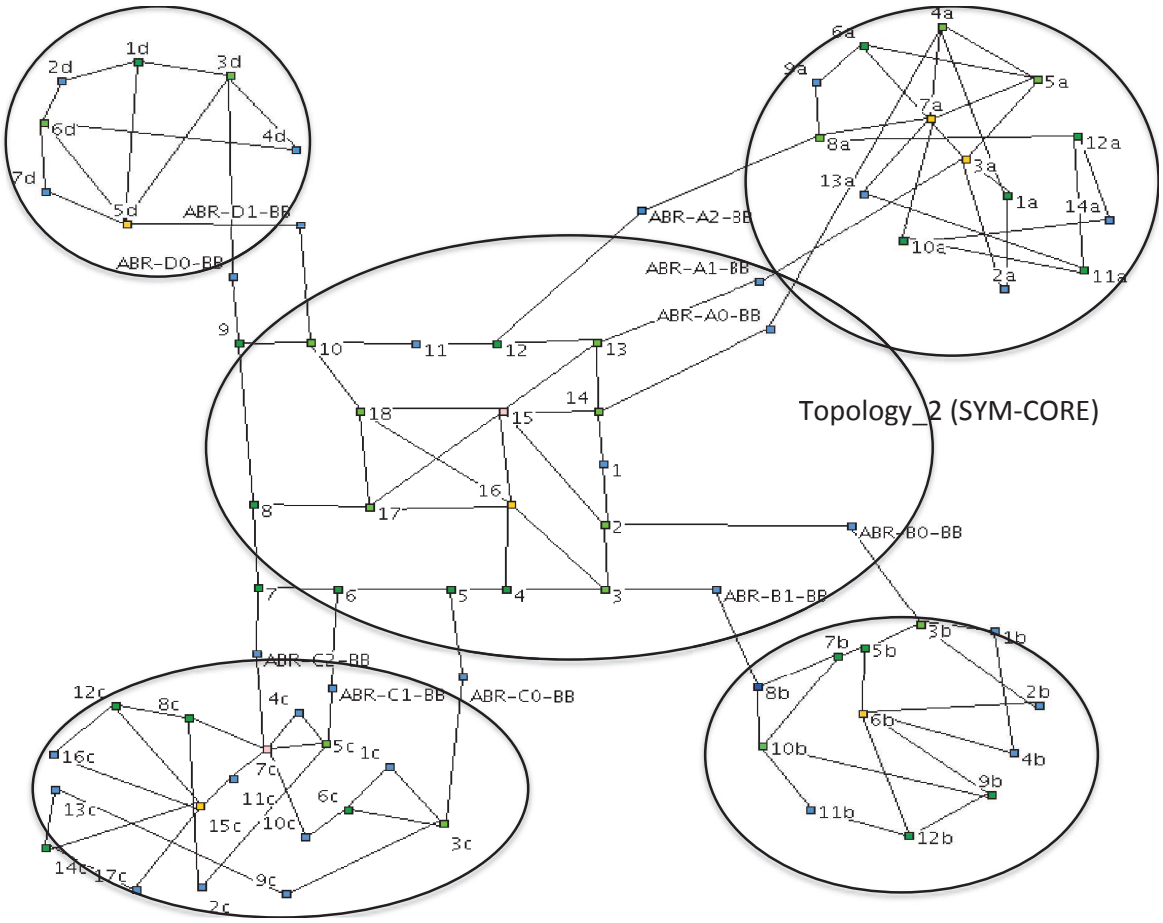


Figure 33 – *Network topology 1 (SYM-CORE) used for performance metric analysis using an event discrete simulator*

The table below shows the topology and TE LSP characteristics of each network used for simulations.

Network and TE LSPs characteristics in each network in terms of number of links, number of nodes and distribution of TE LSP size per-domain:

Domain name	Number of Nodes	Number of Links	OC48 Links 2.5GBits/s	OC192 links 10 GBits/s	[0,20Mbit/s] TE LSPs	[20,100Mbit/s] TE LSPs
D1	17	24	18	6	125	368
D2	14	17	12	5	76	186
D3	19	26	20	6	14	20
D4	9	12	9	3	7	18
MESH network (area 0)	83	167	132	35	0	0
SYM network	29	37	26	11	0	0

(area 0)						

Details on the simulations:

- Figure 18 and 19 depict the two networks called MESH-CORE and SYM-CORE respectively. Each of them is made of a core network (area 0) surrounded by 4 areas at the periphery (the table above shows the number of nodes in the core area),
- The table above show the link bandwidths and TE LSP size distributions in each network,
- All TE LSP are inter-domain (none of them originates and terminates within the same domain, or originate/terminate in the area 0). The TE LSPs are generated iteratively. Approximately 30% of the TE LSPs have a size between 0 and 20Mbits/s, and the other 70% of the TE LSPs have a size comprised between 20MBits/s and 100MBits/s.
- Each TE-LSP was placed from a source in one domain to every other node in a remote domain chosen for the destination. Note that paths may not exist for some TE LSPs. That being said, we made sure of the TE LSP distribution size above, which is similar to what is observed in real-life deployed networks.
- The data rate of each TE-LSP is chosen according to a uniform distribution in the interval [Dmin, Dmax],
- All TE LSPs are operational (until being affected by a network failure) during the course of the simulation.

3.8.1 Set of assumptions

Per-domain path computation: an improved version of the per-domain path computation techniques described earlier in this chapter is used that implements crankback mechanisms upon CAC (Call Admission Control) set up failure. Indeed, when the head-end of an inter-domain TE LSP signals an inter-domain TE LSP, it first selects a *BN-en* LSR of the adjacent domain that computes the shortest path segment between itself and the selected BN-ex LSR and the process continues until completion.

The signaling of the TE LSP may fail because of outdated TED (similarly to the BRPC procedure) but also because the selected *BN-ex* LSR may not have any path segment meeting the constraints of the TE LSP to reach any of the *BN-ex* LSR in the next-hop domain. In such a situation that cannot occur with BRPC, the first strategy consists of signaling an error back to the head-end LSR, which would in turn select another *BN-ex* LSR. Such an approach has been enhanced by allowing for cranckback (note that this techniques have been used for several years by other protocols such as PNNI). In this case, should a *BN-en* LSR in domain *i* fail to find a path to reach the next *BN-en* LSR in the next hop domain or the TE LSP destination, an error message is intercepted by the *BN-en* LSR in domain *i-1* that would in turn select another BN-ex LSR among the list of candidates. When all candidates have been exhausted, the

BN-ex LSR will itself crankback to the upstream *BN-en* of domain *i-2* that would follow the same procedure.

For sake of illustration let's take the simple example of inter-area TE LSP. Let's consider an inter-area TE LSP from a node *A* in domain 1 to a node *B* in domain 3 that transit through a backbone area (called domain 0) where domain 1 and 0 are interconnected via *ABR1* and *ABR2*, domain 0 and 2 are interconnected via *ABR3* and *ABR4* and all ABR acts as a PCEs (not using BRPC). Upon receiving the path computation request from node *A*, *ABR1* would select a next-hop ABR, say *ABR3*. If *ABR3* cannot find a path satisfying the constraints to node *B* and/or should the signaling fail because of an outdated TED, instead of relaying the error message to node *A*, *ABR1* would then select *ABR4* as its next hop ABR. Should that second attempt also fail, *ABR1* would crankback to node *A* that would in turn select *ABR2* and the procedure would continue until exploring all possible paths.

Failure generation: during the course the simulations, links independently fail with a mean failure time of 24 hours and a mean restore time of 15mn, the inter-failure and inter-restore times being uniformly distributed

The simulation lasts for one week with the smallest unit of simulation being equal to one minute. The inter-arrival between failures is uniformly distributed with the same failure probability for all links in the network.

3.8.2 Performance metrics

Several performances metrics are considered to compare the per-domain approach with crankback to the BRPC approach designed in this thesis.

- **Path cost** is a critical metric for each TE LSP. Shortest paths are computed using the advertised link metric that may reflect the link bandwidth, delay, or any polynomial combination of link attributes. We show the maximum and average path cost, in addition to their distribution.
- **CAC Failures:** CAC failures may take place for two reasons:
    - The first reason is that all nodes in a domain (OSPF area or IS-IS level) maintain an identical copy of the TED that is propagated by the IGP. In order to avoid permanent flooding of LSA to refresh the TED, available resources are updated when crossing a threshold. Although this allows for reducing the number of flooded LSAs, this unavoidably leads to a risk of outdated TED. Consequently, a node X in the network may compute a path using an outdated TED, thus leading to a signaling failure (also referred to as CAC failure).
    - In the case of the per-domain path computation approach the signaling of the TE LSP may also fail because the selected next-hop BN-ex does not allow for finding a path segment satisfying the constraints in the remote domain.

  Comparing the CAC failure rates (both during initial set up and under failure conditions) is a critical performance metric to compare the two approaches.

- **Signaling delays:** as discussed in the previous section, significant delays may occur during crankback, which may dramatically increase the path set up time, a non-negligible issue in situations where TE LSPs carrying time sensitive traffic must be rerouted after a failure in the network.

3.8.3 Performance results for the second topology (SYM-CORE)

In this section, we provide the performance metric comparison for the "SYM-CORE" network; the results are similar in nature with the other network and more details can be found in [49]*.

3.8.3.1 Path Cost

In this section we analyze the distribution of the average and maximum path costs (the average path cost is the average of the TE LSP path cost calculated for all TE LSPs where the maximum corresponds to highest path cost) with the per-domain (PD) and BRPC path computation (PCE) techniques respectively. This distribution is expressed in terms of Cumulative Distribution Function (CDF) of the average and of the maximum path cost.

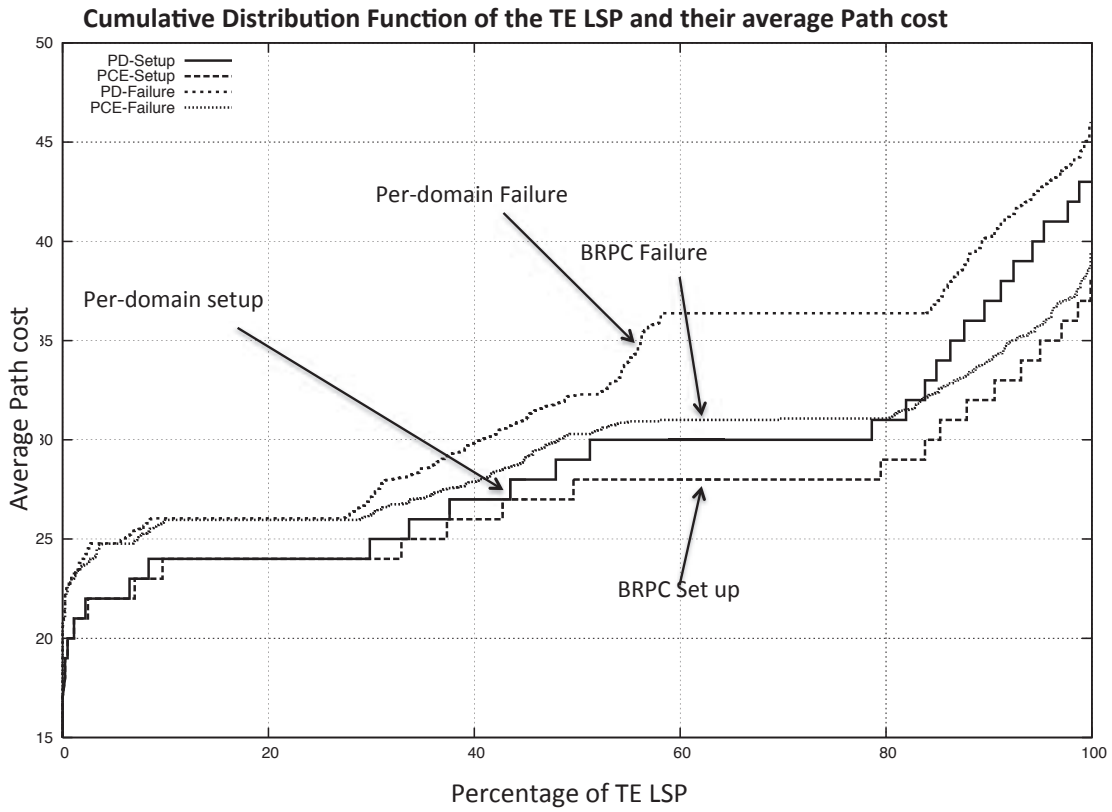


Figure 34 – *Distribution of the average path cost*

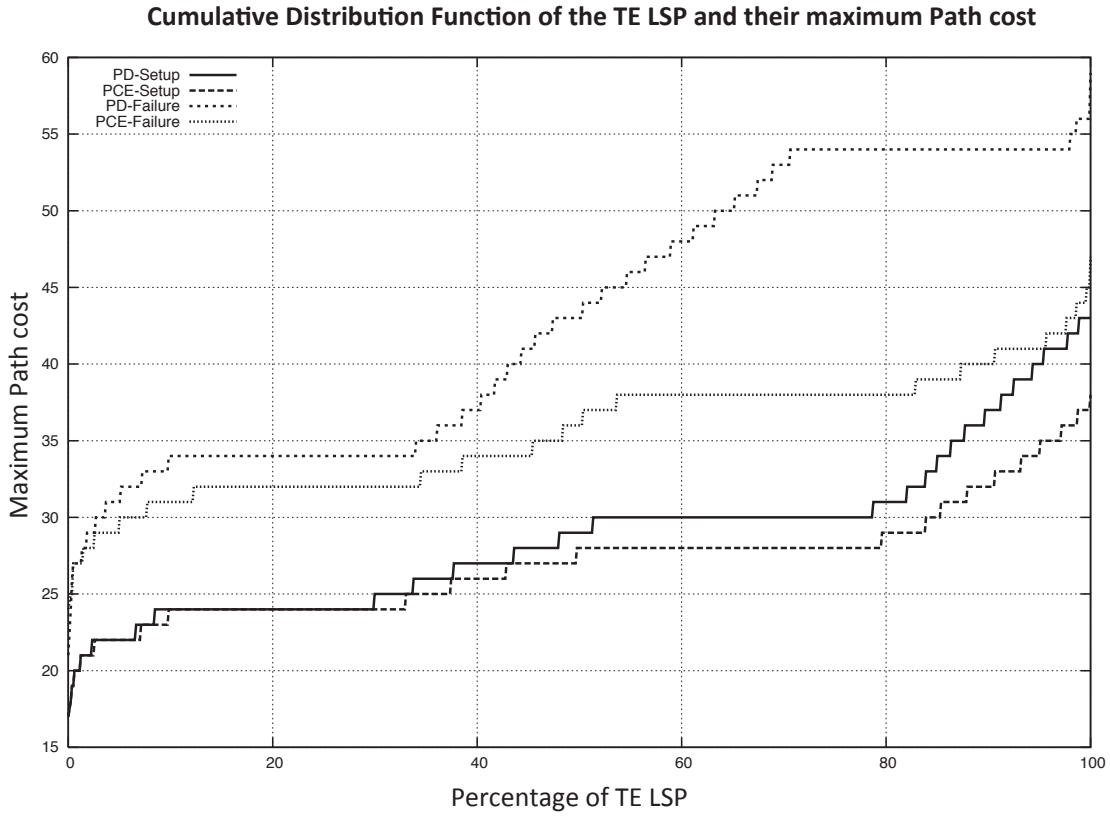


Figure 35 – *Distribution of the maximum path cost respectively*

Figures 20 and 21 outline the benefits of BRPC over the PD approach. For that purpose, two cost parameters are considered: the average path cost and the maximum path cost of the established TE-LSPs. During the same computer simulation, a TE-LSP can be subject to CAC failures and then subject to a reattempt.

Figure 20 outlines the superiority of BRPC against PD in terms of average path cost. Indeed, up to a given average path cost, BRPC finds a higher number of inter-domain TE LSPs than PD. The results are confirmed when link failures occur in the network during the course of the simulation.

Thus, for an average path cost less or equal to 23, we have 40% of the TE-LSPs that are setup under the PD approach whereas for the same cost threshold, 70% of the TE-LSPs are setup under the BPRC. We can conclude that the BPRC enables to setup TE LSPs in wide area networks at lower costs than PD. One observes that the superiority of BPRC over PD is a bit less significant for average costs greater or equal to 24.

The Figure 21 refers to the CDF of the maximum path cost for both PD setup and PCE setup. Again, BRPC enables to set up a majority of TE-LSPs at lower maximum costs.

3.8.3.2 Signaling delays

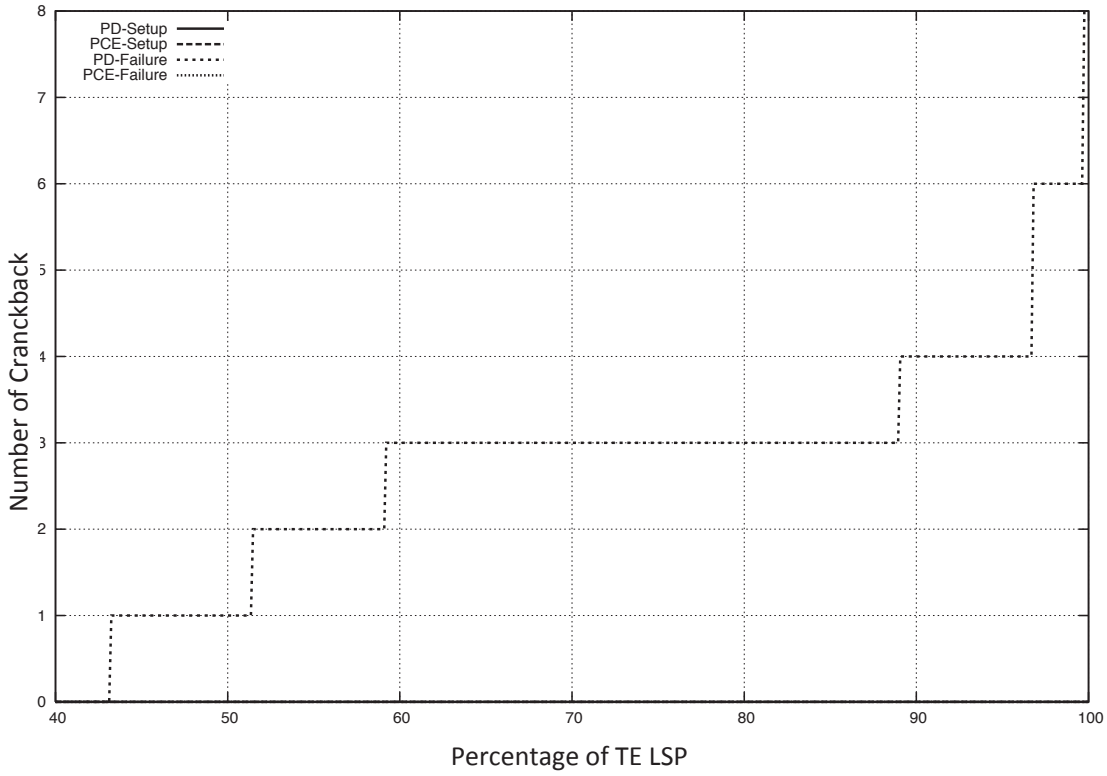


Figure 22 – *Distribution of crankback operations.*

The vertical axis in Figure 22 represents the number of times crankback took place during the course of the simulation during TE LSP establishments ; the metrics is expressed in number of crankback operations.

Figure 22 illustrates the CDF of the number of crankback operations (that themselves have a direct implication on the signaling delays) induced by CAC failures. Such delays contribute in majority to CAC duration. We can observe that only 43% of the TE-LSPs can be established in at most a single crankback. A great majority of the TE-LSPs need at least two crankback operations for being successfully setup. This increase is mainly due to the crankback "trial-and-error" approach with detrimental performance on the overall QoS in the network. Such a phenomena is exemplified in networks where the booking rate gets higher and/or a large number of TE LSP are simultaneously rerouted.

3.8.3.3 CAC Failures

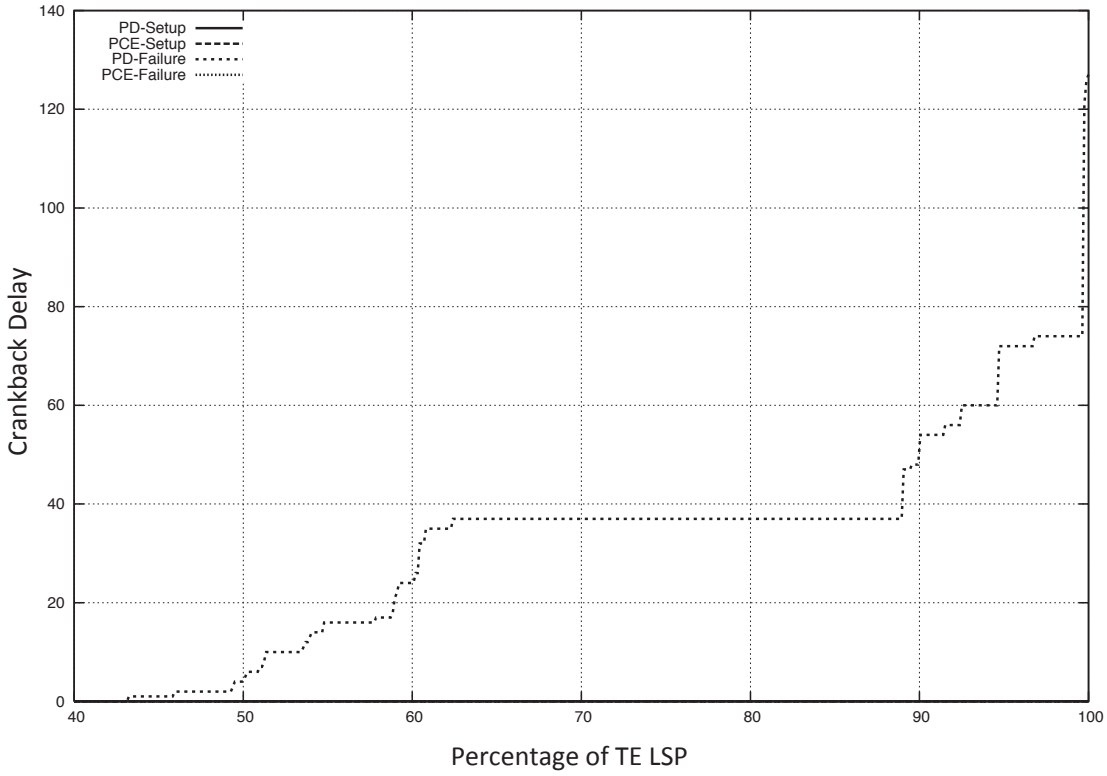


Figure 23 – *Distribution of set up delays due to crankback CAC failures*

As shown in Figure 22, the number of CAC failures occurring in the network during TE LSP set up differs very significantly between the two techniques: where up to 55% of the TE LSP experience a signaling failure when failures occur in the network, that number drops to 3% with the BRPC algorithm.

The distribution of setup delays due to CAC failures in the case of PCE setup does not appear in Figure 23 due its very low values. In the case of BRPC, CAC failures are only due to outdated TED, by contrast with the per-domain approach that "blindly" selects *BN-en* LSRs. It has to be noticed that Figure 23 have similar shapes as Figure 22, because of the direct link that exists between the number of crankback and the path setup delay.

3.8.3.4 Conclusion

All simulations results can be found in [49]* but, as expected, the multi-PCE collaborative approach used by BRPC outperforms compared to existing techniques such as the per-domain path computation. It can be shown that in extreme conditions, the PCE-based path computation algorithm outperforms even more the performances of the per-domain path computation approach. By "extreme conditions", we refer to conditions where the network experiences high reservation

booking rates inducing churn of requests after a massive failures that trigger a large number of TE LSPs establishment.

3.9 Modeling of the PCE request processing an inter-domain multi-PCE collaborative approach

Mathematical modeling of PCE-based architecture and global system is undoubtedly very challenging compared to the modeling of the scheduling activity of a single PCE. A single PCE may be described by means of a network of queues with tractable complexity if the network is made of a very limited number of nodes. Meanwhile, to the best of our knowledge, no analytical model of PCE performance evaluation in complex systems has been proposed in the literature. Modeling analytically a large set of PCEs in WAN environment seems not realistic, unless severe approximations and simplifications are considered. We justify in the remaining of this section why, according to our analysis, only discrete event simulations seem today susceptible to drive to meaningful conclusions in this matter.

First, the analytical modeling of a stream data flow or of an elastic data flow is in itself quite complex. Concerning stream traffic, Markov Modulated Poisson Process (MMPP) [99] must be used to characterize the behavior of adaptive voice or video coders today deployed by the service providers. The rate of the coder/decoder is assumed to be adapted according to the variability of the analog voice or video original signal. The simplest version of MMPP consists in a two-state process describing the fact that the coder can be in two distinct states A and B. For instance, one estimates that state A corresponds to the situation where the variability of the native analog signal is low enough to be coded at low bit rate. At the opposite, state B refers to time periods where the variance of this signal is high and then necessitates a high bit rate coding. One assumes that the packet generation rate generated by the coder depends on its current state. The sojourn times in each of the two states follow a continuous time Markov chain of parameter r_A and r_B respectively. For instance, the sojourn time in state A (state B respectively) is exponentially distributed with an average duration $1/r_A$ ($1/r_B$ respectively). As long as the coder is in state A (in state B respectively), it generates packets with a rate λ_A (λ_B respectively). More generally, voice and video coders can operate with a general number N of states. Analytical formulations describing the behavior of typical queuing systems with MMPP sources have been proposed in the literature these last twenty years [100], [101], [102]. For instance, for single server queuing models like MMPP/M/1/K queues [103], it is possible to determine analytically the blocking probability of such multiplexers based on a single Markovian server under a finite queue size K. Concerning elastic traffic analytical modeling, long range dependence and self similarity have to be considered [104]. It has been shown experimentally and mathematically that elastic traffic (TCP-IP flows) for Web browsing could not be assimilated to a simple Poisson process [105]. It has been observed experimentally that the time correlation between TCP-IP packet inter-arrivals could be highlighted at different time scales. Such a phenomenon is known as self similarity. Self similarity has been subject to numerous mathematical formulations that have

proven their good coincidence with reality. Self similar traffic is characterized by long range dependence in the statistical distribution of packet inter-arrival delays. In comparison, the statistical distribution of packet inter-arrival delays is much shorter under Poisson traffic. Self similarity is quantified by means of the Hurst parameter *H* that is between 0.5 and 1. The higher the value of parameter *H*, the higher the self similarity. For a given average input bit rate, self similar traffic inserted into a queuing system requires much higher buffer sizes than Poisson input Poisson traffic. This characteristic directly impacts the SLA of the IP flows and thus the computation of MPLS path managed by PCE.

A second difficulty of an analytical approach to investigate the performance of Traffic Engineering in PCE environment consists in the modeling of the flow resulting from the multiplexing of several MMPP sources or several self-similar sources. We know today that the superimposition of self similar flows each with its own Hurst parameter is also self similar with a self similarity corresponding to the highest value of the *H* parameters among the self similar tributary flows. Furthermore, new traffic types keep appearing on IP/MPLS network that are even harder to model mathematically, resulting in even more complex path computation requests patterns.

A third difficulty needs to be considered when MMPP and self similar traffic flows have to be multiplexed together in a same multiplexer. This happens typically at an output port of a packet switch or IP router. Indeed, the time structure of the packet flows at the output of the switch or router strongly depends on the service discipline applied to manage simultaneously the two types of data flows while respecting their respective QoS requirements.

A fourth level of complexity has to be considered due to the fact the stream and elastic flows do not transit in a real IP network through a single node but through a succession of nodes within an Autonomous Domain, or (leading to even more complex situations) a set of domains that may be configured using different parameter settings.

Last but not least, the overall dynamics of path computation requests involving a set of PCEs (potential residing in different domains) keeps evolving with very complex (and hardly predictable) patterns due to the collection of traffic demands with highly varying characteristics.

In summary, we can conclude that aiming to model analytically the behavior of PCE traffic engineering in the perspective of its performance evaluation is far too complex to be achievable. This is the reason why this thesis only focusses on discrete event simulations in this matter for example for the performance evaluation of the BRPC algorithm. Note also that many of the algorithms specified in this thesis have been implemented and deployed in various real-life networks with extensive testing of their performance.

Several key research papers have been published in the area of PCE performance modeling such as [50] that quantifies the performance of an inter-layer traffic engineering approach using a PCE using the path set up delay as a quantitative metric. Other research papers [51] have evaluated by means of computer simulations (and not analytically) the respective performance of per-domain path computation, BRPC and H-PCE (Hierarchical PCE) so as to quantify the blocking probabilities due to the lack of accuracy of the TED.

In [62]* we developed a simplified mathematical model for BRPC in the context of inter-area TE LSP path computation using a 3-area IGP deployment scenario where the PCEs collaborate in the distributed computation of inter-area shortest constrained paths according to the BRPC algorithm (without the optimizations listed in Section 3.6 for concurrent path segment computation and path segment encryption techniques as discussed in Section 3.7).

Still it turns out that mathematical modeling of the **overall** system performance remains an open issue and is still considered as very challenging, and open for further research.

3.10 Conclusion

In this chapter, we have specified a new PCE-based algorithm referred to as the Backward Recursive Path Computation algorithm that involves a set of PCEs (called a PCE chain) to perform the distributed computation of the shortest constrained paths across a set of routing domains such as IGP areas and Autonomous Systems.

The BRPC algorithm is then further optimized to allow for parallel computing occurring simultaneously on several PCEs involved in the collaborative PCE chain to further improve the path computation time.

In addition, several mechanisms making use of path keys are introduced to preserve confidentiality across domains thus hiding the set of traversed hops within an Autonomous System, a strong requirement for Inter-AS MPLS Traffic Engineering.

Prior to the emergence of the PCE architecture, as pointed out in chapter 1, a well-known partial approach called the per-domain path computation consisted in computing path segment related to AS at each AS entry node (ASBR) after having "blindly" selected the next hop exit ASBR. Such an approach could not provide the shortest constrained path end-to-end and was suffering from several caveats such as call set up failures, should an ASBR select a next-hop ASBR that could not provide a successful path segment to the next ASBR or destination.

Thus, we conclude this chapter by providing a series of simulation results that compares the per-domain path computation approach and BRPC, showing that BRPC outperforms in a number of ways: optimality of the computed paths but also the call set up failure rates and set up times due to call set up failures. We have also

underlined five main reasons explaining why a performance evaluation of PCE Traffic Engineering is not realistic in terms of complexity.

4 PCE selection techniques and Stateless PCE

4.1 Introduction

During our research, beyond the IGP capability for the dynamic discovery of PCEs, it quickly became apparent that additional information was required in order to adequately select a PCE, should a set of PCE be available to serve a path computation request. The modeling of the arrival rates at the PCE is still challenging and it is worth reminding that with the exception of reoptimization requests, which could be handled by jittering the triggering times to avoid any global synchronization, it is not rare in most networks to observe bursts of requests. Indeed, such batches of potentially large path computation requests may be caused by a link or node failures that may potentially impact a large number of TE LSPs for which a new path computation is required. Modeling such arrival rates is extremely challenging mathematically. Consequently in this chapter we improved our architecture by augmenting the already specified set of algorithms with additional mechanisms and technologies specified.

To that end, we first propose a PCE selection algorithm based on explicit PCE-load indication provided by the PCEs.

Furthermore, in MPLS TE enabled networks making use of head-end based CSPF computation or stateless PCE for intra/inter-domain path computation, TE LSPs are computed without any attempt of synchronization. This undoubtedly provides a great deal of flexibility but with the known caveat of leading to bandwidth fragmentation, an issue similar to the bin-packing problem that is NP-complete. Bandwidth fragmentation has the undesirable effect of increasing the probability of not being able of placing a TE LSP even if a solution exists (at the cost of displacing other TE LSPs). Such global reoptimization can be handled by statefull PCE thanks to the awareness of the network topology, bandwidth reservation states and the set of existing TE LSPs along with their characteristics such as their preemption level (as discussed in chapter 1) by displacing TE LSPs in order to accommodate a new request (that being said, this task may be complex and expensive in terms of signaling, traffic reroute in the network to mention a few issues).

Bandwidth fragmentation is even more prevalent in networks with a wide range of link and TE LSP sizes. In current large-scale MPLS-TE enabled networks it is not rare to see a wide range of link and TE LSP size distribution, which further increases the probability of bandwidth fragmentation. A partial solution to circumvent the aforementioned issues of bandwidth fragmentation consists in assigning preemption levels (priority) according to the TE LSP bandwidth size so as to favor the more challenging placement of large TE LSPs, considering that a TE LSP with priority K can preempt upon signaling any TE LSP of priority L if $K<L$ as specified in [11]. Preemption has first been implicitly specified as “hard” preemption: when preempted, a TE LSP would simply fail leading to traffic loss until the signal (RSVP

TE Path Error) is received by the head-end LSR for the (hard) preempted TE LSP before getting (potentially) rerouted along another path. The solution has been enhanced by the introduction of the notion of "Soft Preemption", specified in [52]*. Instead of tearing down a soft preempted TE LSP, its corresponding bandwidth is locally zeroed, a signaling indication is sent to the corresponding Head-end LSR that gets time to reroute the TE LSP using the well-know non disruptive "make-before-break" procedure (before the expiration of a local timer armed at the time of the preemption by the preempting node). After the expiration of this timer, the TE LSP is hard-preempted by the preempting node if the TE LSP has not been rerouted by its head-end LSR. By combining a policy consisting of assigning preemption according to the TE LSP size and graceful soft preemption, this helps reduce the probability of blocking states in these networks. Although bandwidth fragmentation within a preemption level is unavoidable and may still occur considering the limited number of preemption priorities, such an approach allows for reducing the blocking probability although it is worth mentioning the undesirable effects of signaling churn and traffic jitter experienced by the user traffic during TE LSP reroute.

Thus the last section of this chapter is devoted to specifying an algorithm used to defragment the network when the PCE determines an increased rate of path computation failure, thus leading to increasing the path computation success rate.

4.2 Definition and computation algorithm of the PCR (PCE Path Computation Resource) variable

To that end, we define a new variable called the *PCR* (PCE Computation Resource) that reflects the path computation resources available at a given PCE. The *PCR* has the form of a numerical value, normalized across all PCE thanks to user configurable policy. The *PCR* provides an indication of the path computation resources available, such as the average service time (locally measured by the PCE including the waiting and processing times and averaged out over a set of *N* requests using a low-pass filter), the number of path computation requests that a PCE can handle according to its waiting queue states and the number of requests waiting in the queue or any other metric.

Since the *PCR* must be dynamically advertised using a routing protocol such as OSPF or IS-IS, there is a delicate trade-off between providing an accurate value of the *PCR* for each PCE and preserving the scalability of the IGP. We studied other alternatives such as providing PCE state feedback using the PCEP signaling protocol, but this would have prevented a PCC not having (yet) an active PCEP session with a congested PCE to get such feedback.

Note that although an opaque LSA can be used in order to advertise the *PCR* without triggering any SPF computation, each LSA is flooded across the IGP domain.

Algorithm

Step-1:

The PCE first initializes the PCE value, *PCR* == 0 for an initial period of *T1* seconds

Step-2 (Local computation of the *PCR* value)

- For each path computation request processed by the PCE, a local *Path_computation_time* value including the waiting and processing time for the request is calculated by the PCE
- Every *T2* milliseconds (ms):
    - The PCE computes *PCR(t)*, which represents the instantaneous path computation resource of the PCE.
    - $PCR(t) == (1- \alpha) * PCR(t) + \alpha * PCR(t-T2)$

Step-3 (Trigger for PCR advertisement, occurring in parallel of Step_2):

- ***If*** $(PCR(t)<PCR_{Low})$ ***or*** $(PCR(t)>\beta*PCR(t-T2))$ ***or*** $(PCR(t)<\gamma*(PCR(t-T2))$ ***then*** trigger an LSA generation to update the PCCs in the domain of the new *PCR* value for the PCE.
  End

Discussion about the variables of *T2*, α, β and γ: the values of α and *T2* are adjusted in order to reflect the degree of accuracy of the *PCR* value advertised by each PCE, where smaller values for *T2* allows for more accurate computation of the *PCR* and smaller values for α provides a more instantaneous value of the *PCR*, potentially reflecting a sporadic queuing state with the well-known caveats. The values for β and γ are used in order to trigger a *PCR* update in case a major change of the PCR value, thus triggering the redirection of path requests to alternate PCEs and potential system instabilities.

Since path computation requests can be sent with a signaled priority (setting the "Prio" flag field of the RP object carried in the PCEP PCReq message as specified in [38]), the algorithm described above is further extended so as to advertise the *PCR* on a per priority basis.

4.3 PCE selection algorithm

Algorithm

- Let's *N* be the set of pending path computation requests
- Determine the set *S* of PCEs dynamically discovered by means of IGP advertisement capable of serving the request (e.g. PCE serving requests for TE LSPs having a destination in a specific AS, or any PCE in the case of inter-area TE LSP).
- Determine the set $S' \subseteq S$, of the PCEs supporting the required capability for each element of the set *N* (e.g. ability to compute a set of diversely routed TE LSPs).
- Exclude all element of *S'* for which *PCR*=0 or $PCR<PCR_{low}$ and compute $S'' \subseteq S'$

- ***If*** *N*==1 ***then*** select the PCE with the lowest advertised *PCR* ***else*** load balance the set of *N* requests among the *S''* list of PCEs proportionally to the advertised *PCR(i)* for *i*=1 to |*S''*|.

It is worth pointing out that the environments in which such PCE-based architectures are deployed may greatly vary in terms of dynamics (number of path computation requests, burst sizes, path computation request arrival rates in steady state, number of boundary nodes or off-line servers acting as PCE) and the values of the set of variables listed above may significantly differ according to the networking environment. The uses of learning machines so as to dynamically adjust such values are for further study.

4.4 A bandwidth defragmentation algorithm reducing the blocking probability in a stateless PCE-based path computation architecture

In the last section of this chapter, we specify a new technique in order to mitigate the risk of blocking probability in PCE-based networks due to bandwidth fragmentation.

The first aspect consists of monitoring the Path Computation Failure Rate (*PCFR*), a new variable maintained by each PCE involved in a collaborative multi-PCE path computation chain, such as in the case of the BRPC algorithm specified in chapter 3.

Algorithm: Computation of the *PCFR* variable on a PCE, and bandwidth defragmentation triggers

Step-1:

For all received path computation requests

> ***If*** no path can be found (by the first PCE in the case of the BRPC algorithm) or no path segment can be found that satisfies the constraints listed in the path computation request ***then***
>
> > Let's consider a slotted time approach where time slot are equal
> > Let's *t* be the time at which the PCE determine that no path (segment) satisfies the request
> > Let's *C(t)* be the number of failed path computation for the current time slot
> > Let's $PCFR_{Max}$ be the maximum path computation failure rate before determining that bandwidth must be defragmented
> > Let's $PCFR_{Max\_accelaration}$ be the maximum value of the second derivative of PCFR before determining that bandwidth must be defragmented
> > Once the time slot expire at time *t': PCFR(t')*== (1- α) **PCFR(t*) + α**C(t)*

If $PCFR > PCFR_{Max}$ or $d^2(PCFR(t))/dt^2 > PCFR_{MAX\_accelaration}$ ***then***
Trigger an IGP LSA advertisement signaling a bandwidth defragmentation request for all TE LSPs in the domain the PCE is responsible for.

If the LSA is flooded in the entire AS for inter-area TE LSP path computation or another notification mechanisms is used ensuring that all head-end LSR having at least a TE LSP traversing the bandwidth fragmented domain ***then*** Stop
Else
Each boundary router of the said domain triggers an RSVP notification message for each TE LSP traversing the domain that is sent to the respective head-end, notifying of the request for reoptimization
End if

End if

End if

Until PCE is declared as non-operational or disabled

End for

Step-2

Each head-end LSR notified of a downstream bandwidth fragmented domain according to the previous algorithm triggers a path reoptimization request.

Step-3

Upon receiving path reoptimization request, the PCE responsible for the bandwidth fragmented domain performs the following algorithm:

Arm a local timer *T1*

Repeat

Buffer all received path reoptimization requests

Until expiration of the timer *T1*

Re-order all path computation reoptimization requests according to the bandwidth size.

Start serving the requests by decreasing order in terms of TE LSP bandwidth requirement, inserting a fixed delay between each reply (or signaling setup).

4.5 Mathematical modeling

The algorithms specified in this chapter have not been mathematically modeled. Similarly to the modeling challenges that arose with mathematical BRPC modeling, as discussed in Section 3.9, a mathematical or simulation approach is for further study.

4.6 Conclusion and future work

In this chapter, we specified a set of mechanisms allowing a PCE to inform a PCC (or a set of PCEs in routing domains) of its potential (intermittent or permanent) congested state and available resources thanks to the use of a variable called the *PCR*. The *PCR* is a normalized scalar that provides an indication of the PCE's path computation resources available (e.g. average service time (locally measured by the PCE including the waiting and processing times and averaged out over a set of *N* requests using a low-pass filter), number of path computation requests that a PCE can handle according to its waiting queue states, …).

An algorithm is proposed that consists in using a low-pass filter to compute the *PCR* with a dual-threshold approach to trigger an update of the *PCR* in the routing domain, thus influencing a second PCE selection algorithm specified in this chapter and used by the PCCs.

The algorithms and signaling protocols designed in this chapter allows for efficient load balancing of the path computation requests in the network among a set of PCE according to the path computation request service time requirements and the dynamically computed PCE path computation resources.

We conclude this chapter by specifying new techniques to handle the situation of bandwidth fragmentation that may occur when using stateless PCE, which unavoidably leads to increased path computation failures. In order to mitigate this issue, we propose an algorithm capable of detecting a bandwidth fragmentation issue in a domain thanks to the computation of the Path Computation Failure Rate (*PCFR*) within the domain by a PCE, the use of various triggers such as the second derivative of the *PCFR* to signal the necessity for all head-end LSR to trigger a path reoptimization procedure.

Upon receiving such a bandwidth defragmentation request, path reoptimization requests are then buffered by the PCEs in order to ensure a placement of the TE LSP by order of priority starting with the TE LSP of larger sizes while controlling the rate at which TE LSPs get rerouted. This algorithm allows for detecting and then defragmenting the bandwidth in a domain, thus increasing the success rate of satisfying path computation requests in PCE-enabled networks and globally improve the use of the networks resources.

5 A Distributed-PCE based technique to compute back-tunnel for Fast Reroute Node protection with bandwidth sharing

5.1 Introduction and Problem Statement

The use of a distributed PCE-Based approach to compute backup tunnel used for MPLS Traffic Engineering Fast Reroute (FRR) while providing bandwidth protection is undoubtedly one of the core contributions of this thesis. The proposed approach detailed in this chapter solves a key issue in network recovery, specifying a new model referred to as the "Facility Based Computation Model" (FBCM) ([53]).

Reliability is a fundamental performance aspect of IP/MPLS packet networks and networks architecture have fundamentally evolved over the past decades leading to replacing technologies such as SONET/SDH by IP over DWMDM (fibers) mostly thanks to the emergence of fast recovery techniques such as (link/node) FRR and predictive quality of service. Such IP/MPLS protection mechanisms provide fast recovery of the traffic with QoS guaranties upon a link or node failure within a few dozens of milliseconds thanks to local protection techniques.

A brief overview of FRR was provided in chapter 1: one of the key challenges lies in the provisioning of backup-tunnels used to reroute TE LSPs (thanks to encapsulation, pushing an extra outer label) affected by a link/node/SRLG failure.

Thanks to signaling, the head-end LSR of a TE LSP has the ability to require the use of MPLS TE FRR (link or node protection) in case of failure in the network. We enhanced the signaling of RSVP-TE so as to be able to indicate whether the TE LSP should also be "bandwidth protected"; if bandwidth protection is required, this means that the TE LSP must be rerouted over a backup tunnel offering an equivalent bandwidth for the duration of the local reroute.

Figure 24 illustrates the notion of bandwidth protection whereby when a node failure is detected, all TE LSPs requesting FRR are locally rerouted onto a backup tunnel between the PLR and the MP; the PLR then signals the local reroute to the head-end, which triggers a reoptimization (reroute) of the TE LSP avoiding the failed element. Between the local reroute and global reroute, the bandwidth allocated to the backup tunnel used to reroute all TE LSPs affected by the failure directly impacts the QoS provided to the rerouted traffic.

Although QoS degradation for a short period of time may be tolerable for some traffic, there are traffic types requiring to not only guaranty fast recovery (within a few (tens of) milliseconds) but also the absence of QoS degradation during reroutes (prior to the reoptimization of a locally rerouted TE LSP) so that Service Level

Agreement (SLA) could be preserved. Let us recall that the complexity of node's failure recovery increases exponentially with the failing node's degree.

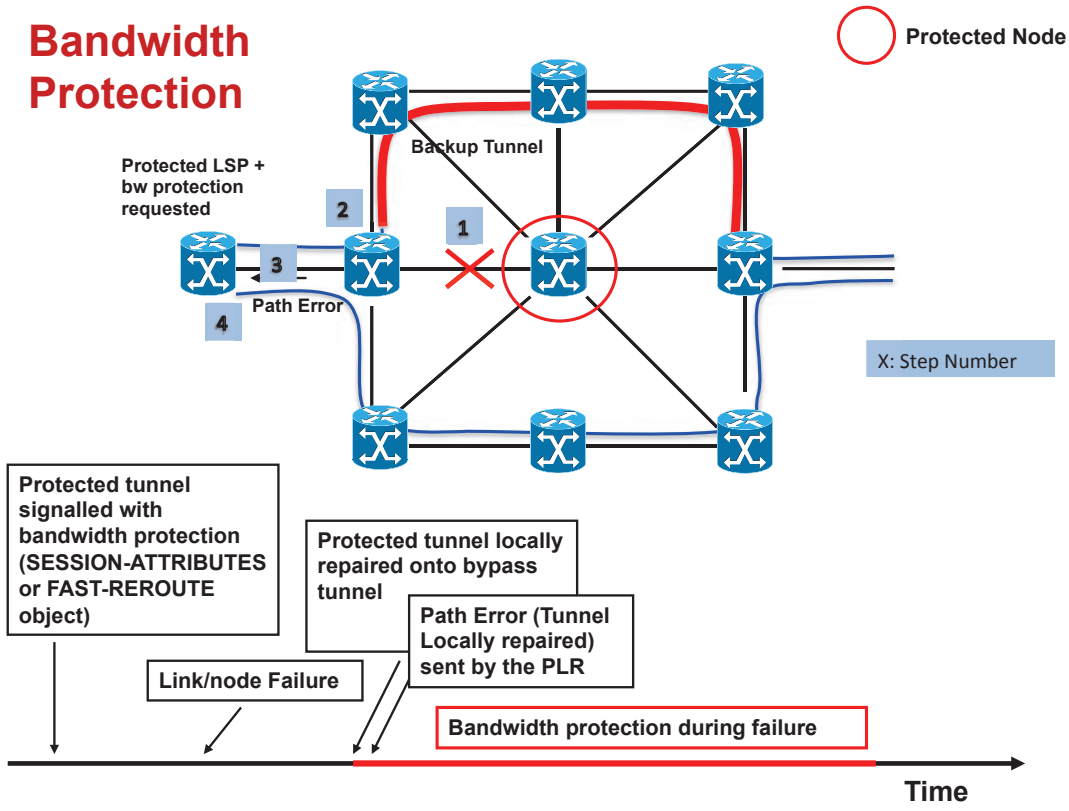


Figure 36 – *The notion of "bandwidth Protection" during local reroute with FRR*

Although the new PCE-based backup tunnel path computation technology specified in this chapter applies to node FRR Link and Node protection, we will focus on FRR Node protection with bandwidth protection, which is a superset of the problem space. Indeed, at a first sight, considering the higher cost of node protection in terms of backup bandwidth provisioning and the overall complexity, it was first envisioned to exclusively focus on link protection. This was even more compelling since link failures occur dramatically more often than node protection, especially considering the number of functionalities supported by modern routers such as Non-Stop forwarding, graceful restart. That being said, several major outages that took place in North America ([54]) showed that nodes could fail; furthermore a node failure may have a dramatic impact on the overall network availability compared to a link failure. Consequently, this led us to concentrate our work on the superset of the problem space (node failures) in additional to SRLG failure.

The approach designed in this chapter applies to link, SRLG, overlapping SRLG and node protection.

We call "back-up capacity" the bandwidth dedicated to backup tunnels in the network that cannot be used for primary TE LSPs.

5.2 The Naïve CSPF-based Path Computation (NCPC) approach

The simplest approach to compute backup tunnels supporting FRR node protection with bandwidth protection consists in using the CSPF-based algorithm on each node in the network, in a totally non-collaborative manner (independently):

Algorithm:

For each next-next hop LSR ***then***

> Determine the set S of primary TE LSPs traversing the protected node and the next-next hop LSR,
>
> Compute the sum of the primary TE LSP bandwidth $Prot_{BW}$ for all TE LSP requiring node and bandwidth protection,
>
> Trigger a CSPF computation for a backup tunnel with $bandwidth == Prot_{BW} + \Delta$ where Δ is a user configurable margin factor should new primary TE LSPs using the backup tunnel be signaled in the future in order to avoid a re-computation of the backup tunnel,
>
> Signal the backup tunnels with $bandwidth == Prot_{BW} + \Delta$

End for

The NCPC approach is undoubtedly extremely simple but exhibits a number of undesirable properties and limitations:

- First, it may not always be possible to find a placement for all back-up tunnels, the reason being that backup tunnels are computed independently (the issue is no different than in the case of CSPF-based primary TE LSP computation). As shown later in this chapter, the new approach relying on the computation of a set of backup tunnels headed at different PLRs coordinated by a PCE allows for greatly maximizing the chance to find a placement of such as set of backup tunnels compared to the "greedy" NCPC algorithm.
- Second, such an approach is quite inefficient in terms of required backup capacity. Indeed, one of key properties of packet-based networks lies is in traffic multiplexing, which also applies to the required capacity in the network dedicated to handle network element failures, a prime objective being to reduce the required backup capacity, which is only used by a subset of traffic and for a short period of time. The NCPC approach could be optimized thanks to the use of predictive algorithms allowing to predict the actual amount of traffic routed onto the primary TE LSP in order to compute the required back-up capacity (instead of over-provisioning the required

back capacity by summing up the signaled bandwidth) – see [55], [56] and [57]. That being said, the recomputation of the required backup capacity is not always possible considering the pace at which the traffic router over primary TE LSPs varies; furthermore, this requires signaling extension so as to communicate the actual bandwidth used for each TE LSP (which may differ from the signaled bandwidth) to each traversed LSR, a very costly operation in terms of signaling. Another alternative to make the naïve approach slightly more optimal consists in using auto-bandwidth (see [12]), a technique whereby TE LSPs are dynamically signaled with a bandwidth value reflecting the actual traffic.

Although these techniques allow for mitigating the sub-optimality of the NCPC approach, the technique specified below clearly outperforms the NCPC approach in many ways, as discussed in the next section.

The NCPC approach does not allow for bandwidth sharing, as discussed in the next section.

5.3 Notion of bandwidth sharing between backup tunnels

A number of discussions took place in the scientific community on the topic of minimizing the required backup capacity and a reasonable assumption arose according which bandwidth could be shared between back-up tunnels protecting *independent* resources, thus resulting in large saving of backup capacity. Such an approach is known as backup multiplexing. Note that the general issue of backup capacity is discussed in great details in [20].

Protecting *independent* resources means that if two backup tunnels T1 and T2 protect primary TE LSPs against the failure of say two resources R1 and R2, they can share their "reserved" backup capacity if and only if the probability for R1 and R2 to simultaneously fail is extremely low. A counter example would be the case of two IP links using DWDM and sharing the same fiber (which is known as the notion of Shared Risk Link Group (SRLG)): in this case, without taking care of the existence of an SRLG, the fiber failure would lead to the simultaneous failures of both links (although they are perceived by the IP layer as separate links without fate sharing).

In the case of two nodes (LSR), it is generally admitted that the probability of simultaneous failures of both nodes is extremely low, except in specific cases controlled by the network administrator where both nodes are co-located and could simultaneously be impacted by a network power outage for example. Such situations are exceptional and correspond either to natural disaster or malicious actions on the infrastructure. Unfortunately, the lessons of the recent past in Japan has outlined that even if the probability of natural disasters is extremely low, such events deserve a specific attention.

The probability for two nodes to fail at the same time or for a node Y to fail right after the failure of the node X and before the reoptimization of the TE LSPs affected

by the failure of node X is extremely low (the red line on Figure 24 shows the period of time during which FRR is in operation, the TE LSPs are locally rerouted and not yet reoptimized by their respective head-end LSR). We also make the assumption that the network has been provisioned in terms of bandwidth capacity so as to provide enough capacity to find alternate paths for all TE LSPs (with the same bandwidth) in case of a failure of any network element; otherwise in the case of a single element failure some TE LSPs would stay on their locally rerouted path.

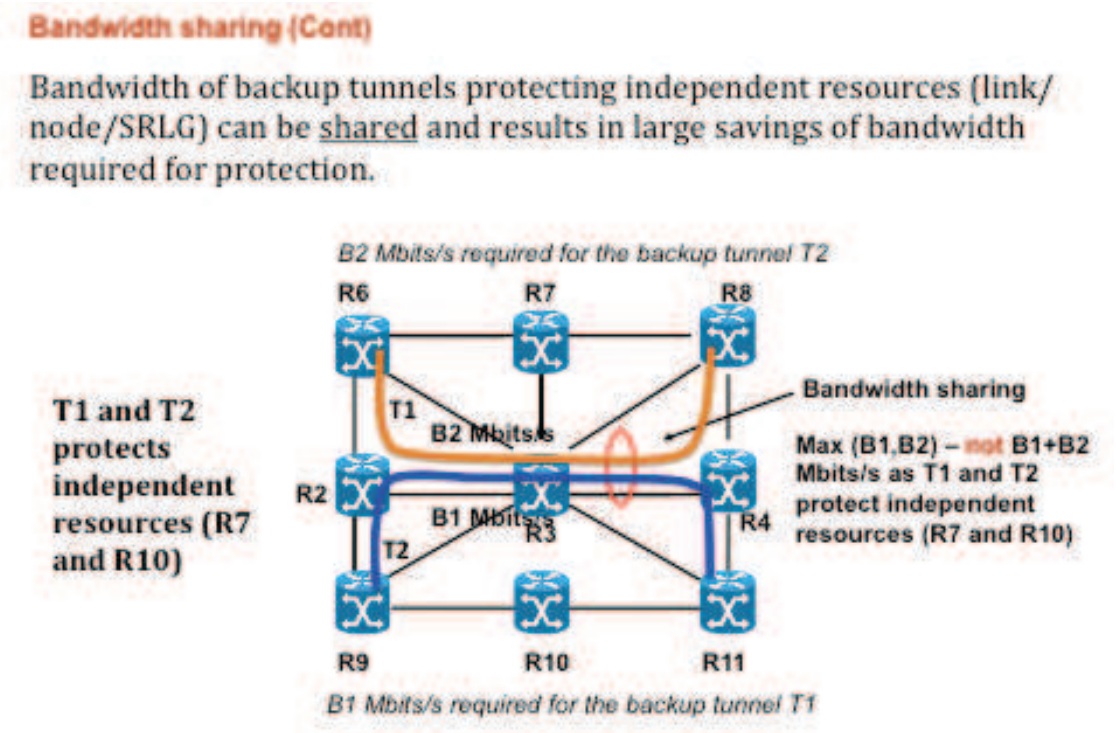


Figure 37 – *Illustration for the notion of bandwidth sharing between backup tunnels protecting independent resources*

Figure 25 shows that if the assumption is made that R7 and R10 will not fail at the same time, then it becomes possible for the backup tunnels T1 and T2 protecting the nodes R7 and R10 respectively to share bandwidth. Consequently the bandwidth reserved for T1 and T2 on the links R2-R3 and R3-R4 is equal to the maximum of their bandwidth value as opposed to the sum of their bandwidth thus leading to reducing the required backup capacity in the network.

Backup multiplexing occurs if two or more backup shares at least one hop. In fact, four cases have to be considered to determine the bandwidth of the backup path depending on the time-space correlation of the primary paths. If two primaries TE LSPs P1 and P2 are space and time disjoint (we mean by time disjoint that P1 and P2 are not active simultaneously), the common links of the two associated backup paths must have a capacity of Max(B1,B2). If P1 and P2 are space disjoint but not time disjoint, the common links of the two associated backup paths must also have a capacity of Max(B1,B2). If P1 and P2 are not space disjoint but time disjoint, the

common links of the two associated backup paths must have again a capacity of Max(B1,B2). At last, if P1 and P2 are not space disjoint and not time disjoint, the common links of the two associated backup paths must have a capacity B1 + B2. These properties can be generalized in the case of three or more primaries with their respective backups. Note that unfortunately, in operational network the lack of predictability of the traffic forces us to make use of the pessimistic worst-case assumption of primary TE LSPs are neither time nor space disjoint.

5.4 The Facility Based Computation Model (FBCM)

FBCM is a new approach designed in this thesis that consists in having the protected node acting as a PCE to compute the next-next hop backup tunnels **for** its neighbors, still allowing for bandwidth protection and sharing, in case of its own failure.

Several other approaches have been proposed in [58] and [59]. In the first case, it requires heavy routing extensions to signal the backup tunnels and their characteristics, in order to determine which backup tunnels is protecting which resources and achieve sharing of bandwidths between tunnels protecting independent resources. [59] proposes a lighter approach in terms of extra-signaling overhead at the cost of less-efficient degree of bandwidth sharing. Furthermore, in both cases, the approach requires a modification of the call admission control in order to avoid double booking accounting of back tunnel bandwidth protecting independent resources.

What makes FBCM a novel and a very efficient approach is that it does not require any routing or signaling extensions, nor does it require modifying the call admission control procedure, while allowing for maximum degree of bandwidth sharing.

Although FBCM supports both a centralized and distributed PCE-based backup tunnel path computation models, we will concentrate on the distributed version of FBCM. Although the distributed version of FBCM provides a greater degree of flexibility, it is technically more challenging and has been preferred over the centralized approach.

The first fundamental component of FBCM lies in the computation of the backup tunnel protecting the primary TE LSPs traversing a node X by the node X itself, which acts as a PCE for its neighbors, although the backup tunnels are headed at each neighbor of the PCE. Back to the example shown in figure 27, the node R2 would act as a PCE to compute all next-next-hop (NNHOP) backup tunnels for its neighbors R1, R3, R5, and R7. In the case of the node R1, there are three NNHOP backup tunnels: R1-R2-R7, R1-R2-R3 and R1-R2-R5 (there may be more backup tunnel if more than one NNHOP backup tunnel is required between a pair of LSRs, should a single backup tunnel with enough backup capacity not be found)..

5.4.1 Signaling backup tunnel with zero bandwidth

At a first sight, it seems necessary to explicitly signal the bandwidth of a backup tunnel. Unfortunately, such an approach requires to extend the RSVP-TE signaling

so as to indicate the identity of the node protected by the backup tunnel in order to share bandwidth on the traversed nodes along the backup tunnel path between backup tunnels protecting independent resources; this would require to modify the call admission procedure.

In contrast, the FCBM approach relies on the signaling of the backup tunnels with 0 bandwidth. The idea consists of splitting the bandwidth pool of each link into a primary bandwidth pool used to withdraw bandwidth for primary TE LSPs and the backup pool strictly used for backup tunnels. Figure 26 shows an exemplary network comprising a variety of links of different capacities (the link capacity is illustrated by the thickness of the links): in this network, each link is made of two bandwidth pools: the primary pool used by primary TE LSPs and the backup pool that is exclusively used to route backup tunnels; in other words the backup pool cannot be used by CSPF or PCE to find a path for a primary TE LSP. Such a split is required to avoid a depletion of bandwidth reserved for backup tunnel should all primary TE LSP exhaust the reservable link bandwidth on each link.

Because backup tunnels are signaled with 0-bandwidth, there is no need for changing the call admission control procedure since two backup tunnels would implicitly share bandwidth (their bandwidth is equal to 0). The reason why signaling backup tunnel with 0-bandwidth still allows for bandwidth protection while allowing for sharing is explained hereafter.

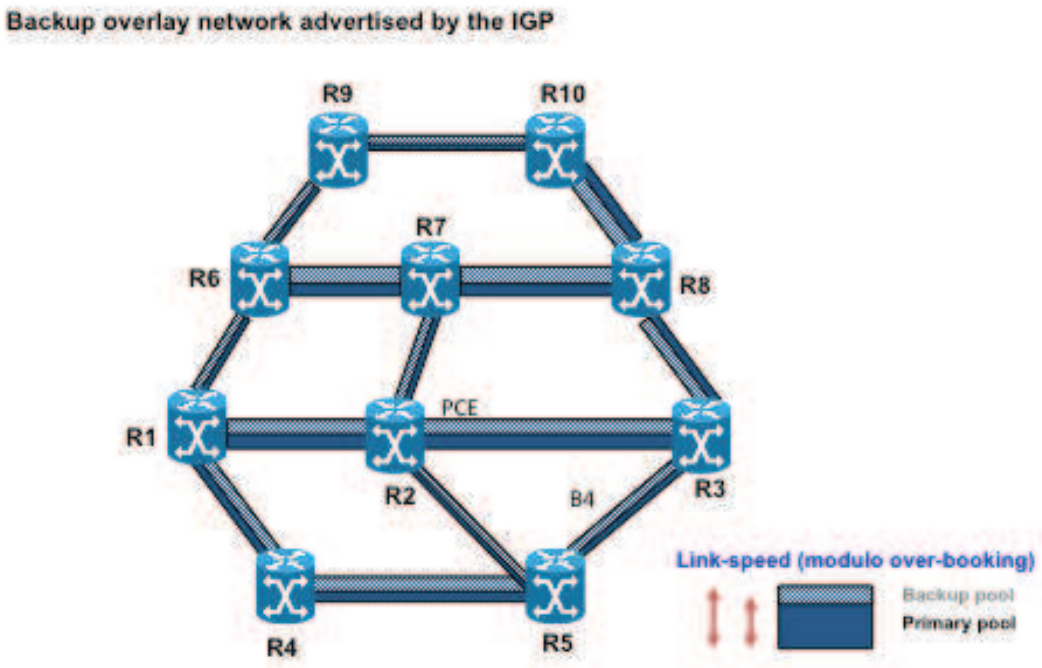


Figure 38 – *Illustration for the notion of primary and backup pools*

Consequently, the IGP routing extension is slightly modified to now advertise for each link not only the primary pool (that may itself be divided in multiple (nested)

pools) but also a static backup bandwidth pool. The fact that the backup pool is static since no bandwidth is withdrawn from the backup pool (backup tunnels are signaled with 0-bandwidth) helps preserve the IGP scalability since there is no need in updating the backup bandwidth capacity as new backup tunnels are signaled (a fundamental scaling property of FBCM).

The network topology made of all links with their respected backup bandwidth pool is also referred to as the backup network overlay (see Figure 25).

The following components are fundamental to the FBCM approach:

- Each link bandwidth is divided into a primary pool from which bandwidth is withdrawn by the primary TE LSPs and a static backup bandwidth pool, both advertised by the IGP,
- Use of the "single failure" assumption according which independent nodes do not fail at the same time, and a second network element would not fail before the TE LSPs affected (rerouted) by the first failure get globally rerouted (reoptimized)
- Backup tunnels are signaled with 0-bandwidth

It is worth mentioning that a node failure results in the simultaneous number of network elements. Although we describe the FBCM approach in the context of a node failure, it similarly applies to SRLG failures (since a SRLG failure also leads to the simultaneous failure of several IP links).

5.4.2 Bandwidth protection and bandwidth sharing with FBCM

The fundamental property of the FBCM approach lies in the fact that all backup tunnels protecting primary TE LSPS from the failure of a node *X* are computed by that node *X* for its neighbor. Furthermore, the node *X* acts as a PCE and is allowed to use the whole capacity of the backup overlay network. Because node *X* computes all NNHOP backup tunnels, this guarantees bandwidth protection: indeed, such NNHOP backup tunnels are computed without sharing bandwidth since they all protect primary TE LSPs against the failure of the **same** node: consequently these backup tunnels are simultaneously active upon the failure of the same node. **This allows the FBCM approach to provide bandwidth guarantee**.

Since all nodes acting as PCEs compute the NNHOP backup tunnels independently of each other, this implicitly allows for bandwidth sharing between backup tunnels protecting independent resources, yet another desired property of the FBCM model.

Thus two nodes *X* and *Y* computing NNHOP backup tunnels for their respective neighbors to protect primary TE LSPs against their own failure would make use of the full backup network capacity, which allows for bandwidth sharing (under the single failure assumption) and bandwidth protection, still without requiring any signaling or routing extensions (except the minor extension to advertise the backup pool), nor does it require any modification of the call admission control procedure.

The FBCM is illustrated in Figure 27 for a single node requesting NNHOP backup tunnel from it neighbor.

Algorithm (triggered by each node in the network acting as a PCC)

For each neighbor

Send a Path Computation Request (PCEP PCReq message) requesting to obtain all NNHOP backup tunnel paths

Wait until reception of a Path Computation Reply from the neighbor reporting the set B_n of each backup tunnel where n is the number of next-next-hops of the PCC traversing the said neighbor (acting as a PCE)

For n=1 to n

Signal the backup tunnel with 0-banwidth according to the received path

End for

End for

Algorithm (triggered by each node in the network acting as a PCE)

Compute the set n of next-next hops from the requesting PCC via the PCE

Compute a set of backup tunnels B_1 .... B_n using the signaled backup overlay network **without** bandwidth sharing taking into account all NNHOP backup tunnels for its neighbors

Return the set n of backup tunnels computed path

End

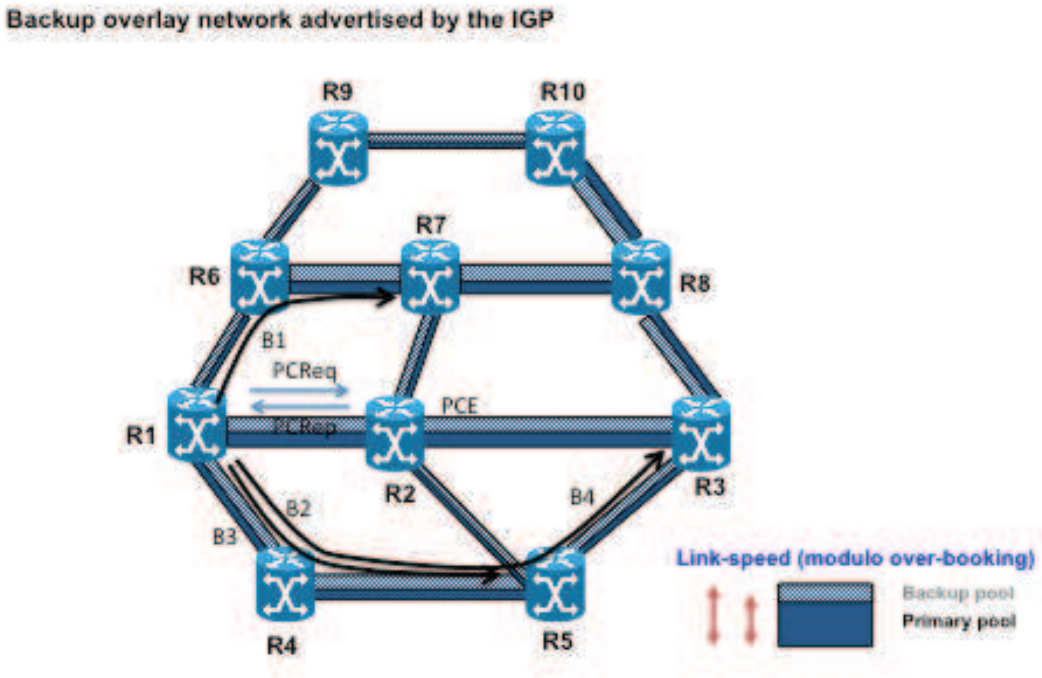


Figure 39 - *Set of NNHOP backup tunnels computed by the PCE Node R2*

Referring to the figure 27, a node *R1* sends a request to the node *R2* acting as a PCE. Upon receiving the NNHOP backup tunnels path computation request message, *R2* computes the set of NNHOPs==<*R7*, *R3*, *R5*>. For each NNHOP, a backup tunnel path is computed that takes advantage of the advertised backup overlay network capacity. In that example, *B2* and *B3* cannot share bandwidth (thus guaranteeing bandwidth protection) since they would both be simultaneously active upon a failure of the node R2 to protect the primary TE LSPs following the path segments [*R1*-*R4*-*R5*] and [*R1*-R4-*R5*-*R3*] respectively. A similar procedure would apply to all nodes surrounding the node *R2* for their respective NNHOP via the node *R2*.

Illustration of the FBCM algorithm

Figure 28 illustrates how FBCM effectively achieves bandwidth sharing. What is now shown is the set of NNHOP backup tunnels computed by the node *R7* (acting as a PCE) for each NNHOP of the requesting node *R6*, leading to the three backup tunnels *B5*, *B6* and *B7* (in this example, two backup tunnels *B5* and *B7* are used to protect the primary TE LSPs traversing the *R6*-*R7*-*R8* path against a failure of the node *R7*, using load balancing).

What can be observed is that *B2* and *B7*, which have been computed by two independent PCEs (namely *R2* and *R7*) have been signaled with 0-bandwidth and consequently share backup capacity along the links *R6*-*R9*, *R9*-*R10* and *R10*-*R8* since under the single failure assumption they are not simultaneously active (R2 and R7 do not fail at the same time). Conversely, the backup tunnels *B5* and *B6* that protect against the failure of the node *R7* have been computed by the PCE *R7* without

sharing bandwidth. This illustrates how FBCM allows for bandwidth protection with bandwidth sharing between backup tunnels protecting independent resources.

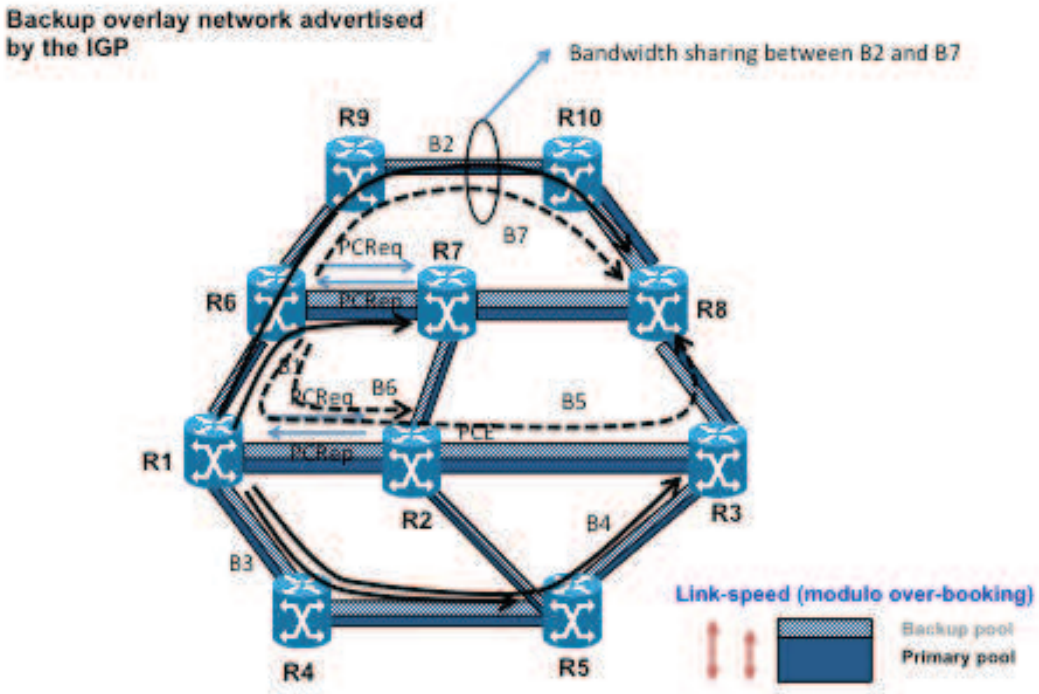


Figure 40 – *Illustration of bandwidth sharing between two NNHOP backup tunnels computed by the PCE Nodes R2 and R7 for two specific backup tunnels*

5.4.3 Increasing the probability of backup tunnel success placement of FCBM

As discussed earlier, the non-coordinated nature of the NNHOP backup tunnels path computation of NCPC is likely to lead to the inability of placing backup tunnels even though the backup capacity is actually available. In contrast, because NNHOP backup tunnels are computed by a single entity (the protected node acting as a PCE), the probability of success in placing the set of backup tunnels is significantly increased with the FCBM approach.

The Figure 29 illustrates a simple exemplary network where backup tunnels placement can be found with FCBM and would fail with NCPC (provided a specific order of CSPF path computations).

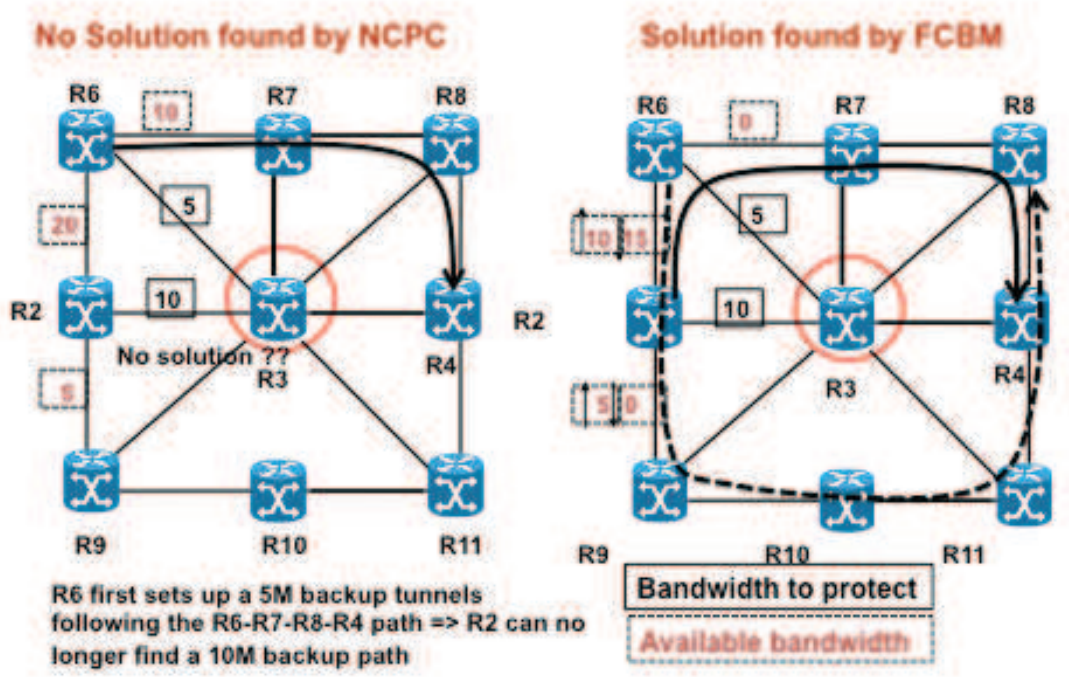


Figure 41 – *Increased probability of backup tunnel success placement with FCBM*

5.4.4 Required bandwidth to protect

The FCBM model does not make any assumption on the amount of bandwidth to protect, which could either be:

- The **total reservable bandwidth** (primary bandwidth pools): in this case, for each NNHOP back-up tunnel T between a node N_{i-1} and N_{i+1} required to protect the primary TE LSPs transiting via the node N_i acting as a PCE, the required bandwidth for the backup tunnel is equal to the Min (*primary_bandwith_pool* (N_{i-1}-N_i), *primary_bandwith_pool* (N_i-N_{i+1}).
  Let's consider a node K, computing backup tunnels for its n neighbors
  Let's n be the total number of neighbors of the node K
  n-1 is the number of backup tunnels computed by node K for a neighboring PCC called node X
  The total number of backup tunnels computed by K=$(n\text{-}1)^2$
  Consider two backup tunnels B_i and B_j where $i,j \in [1, (n\text{-}1)^2]$ and $i \neq j$
  The requested capacity for each link L_t traversed by B_i and B_j = Min (*Protected_bandwidth* of each link along the path segment protected by B_i and B_j).
  The major benefit of protecting the primary bandwidth pool regardless of the actual reserved bandwidth is that the computation of the backup tunnel is not impacted by the set of primary TE LSPs. Indeed, the entire primary pools

are protected whether or not primary TE LSPs are actually using the primary pool. Although the amount of backup capacity required is actually higher than required, this provides a great deal of stability since new backup tunnels placements are only triggered when a topology change takes place in the network. The overuse of backup capacity required by this scheme consisting in protecting the full primary pool may only be problematic if backup tunnels protecting these pools cannot be found considering the capacity of the overlay backup network. This would lead to partial protection (the bandwidth of the backup tunnel is less than the actual primary pool) or backup tunnel split as discussed hereafter.

- The second approach consists in protecting the actual amount of reserved bandwidth: by contrast, with this approach, a node *X* acting as a PCC sends a path computation request to its neighbor node *Y* in case of node protection, requesting backup tunnels path for each of its NNHOPs traversing the node *Y*. Considering a node *Z* being a neighbor of the node *Y*, the bandwidth required for the backup tunnel *T* protecting the path segment [*X-Y-Z*] is equal to the sum of bandwidths of all primary TE LSPs traversing the [*X-Y-Z*] path segment, potentially multiplied by a multiplying factor Δ leaving room for further admitted primary TE LSPs, which unavoidably leads to requiring less backup bandwidth than in the former case where the full primary bandwidth pool is protected.

  The major drawback of this approach is that backup computation needs to be triggered each time the sum of the primary TE LSP bandwidths (modulo the multiplying factor Δ) is exceeded. In highly dynamic environments where TE LSPs are often rerouted because of frequent link or node failures, or because TE LSPs' bandwidths are dynamically re-adjustment to accommodate for traffic fluctuation (e.g. use of auto-bandwidth), such frequent backup tunnel computation may lead to undesirable instability. Not only nodes need to send path computation requests to their neighbors on a frequent basis, but these neighbors acting as PCE need to re-compute a new set of backup tunnels that must be re-signaled. Furthermore, each time a new backup tunnel is signaled the PLR must trigger a local computation to select the backup tunnel for each primary TE LSP. Although this operation is fairly straightforward when only one NNHOP backup tunnel is in place for a given NNHOP, this may become cumbersome should a set *n* of backup tunnels be required to protect a single path segment (this bin-packing problem being known as an NP-Complete computation problem). Last but not least, this may require (should be PCC be completely stateless) retrieving the NNHOP backup tunnels currently in use by all neighbors.

An alternative consists in adopting a mixed approach switching between the two modes of operation described above. A PCC may start by requesting full primary bandwidth pool protection. For each protected path segment, if a single backup tunnel can be found by the adjacent PCE node or a very limited number of backup

tunnels are sufficient, then the PCC adopts the first approach, otherwise it switches back to either protecting a bandwidth pool as close as possible to the actually primary pool, or it signals the requested bandwidth to be protected (sum of bandwidths of the primary TE LSPs for the protected path segment).

5.4.5 Backup tunnel split

Once a backup tunnel *B* has been computed for a protected path segment *P*, the PLR needs to assign the set of primary TE LSPs that traverse the path segment *P* and that will make use of *B* is case of next-hop node failure.

Once again, in the case of link or SRLG protection, the mechanism is identical. An LSR must assign a backup tunnel for each primary TE LSP. These procedures and algorithms are not specific to the node failure protection case.

As described in the previous section, it may not always possible to find a single backup tunnel offering bandwidth protection for a path segment *P*, thus leading to a set of *n* backup tunnel *Bi* with $i \in [1,n]$. In such a case, the PLR must then perform a backup tunnel selection for each primary TE LSP sharing the same path segment.

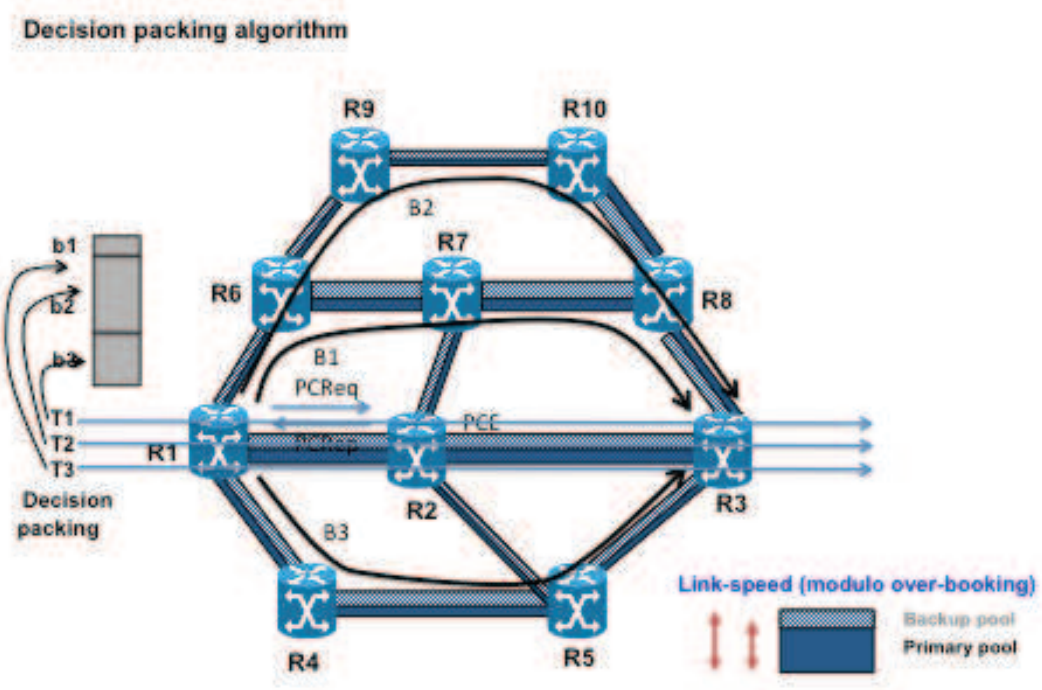


Figure 42 – *Bin-packing problem required in case of Bn with n>1 NNHOP backup tunnels protecting a path segment*

The decision-packing algorithm is illustrated in Figure 30. In this example, the PCE *R2* has returned three backup tunnels *B1*, *B2* and *B3* in order to protect the bandwidth of the path segment *R1-R2-R3* in case of failure of the node *R2*. As a reminder the protected bandwidth is either equal to the Min (*bandwidth_pool_link*

(*R1-R2*), *bandwidth_pool_link*(*R2-R3*)) or the Sum of all primary TE LSPs traversing the path segment *R1-R2-R3* modulo the multiplying factor Λ (in this example, the sum of bandwidth of the primary TE LSPs *T1*, *T2* and *T3*). The PCE (*R2*) then returns the path and bandwidth for each backup tunnel *B1*, *B2*, *B3*, which leaves the PLR (*R1*) with a bin-packing decision so as to determine which backup tunnel to use for each primary TE LSP. The backup selection algorithm is non trivial (NP-Complete problem) and out of the scope of this thesis, this area being covered in many research papers and thesis.

In order to avoid the situation whereby the PCE would return a large set of backup tunnels each with a small (potentially non usable) bandwidth size, we have extended in this research the PCEP protocol in order to allow the PCC to signal in its path computation request the maximum number of backup tunnels that must be computed for a given protected path segment along with the minimum amount of bandwidth for each backup tunnel. Such an extension mitigates the risk of unsolvable backup tunnel allocation at the PLR, and these variables can be used according to the nature of the primary TE LSPs (a large number of small primary TE LSP allows for more flexibility in terms of backup tunnels, thus a large number of small backup tunnels is tolerable; this is in contrast with the case where the PLR has to assign backup tunnels for a small number of large primary TE LSPs).

5.4.6 Extensions to multiple backup pools

As already pointed out, FCBM is not restricted to the support of MPLS TE Fast Reroute Node protection and for a single pool of bandwidth. Various modes of protection can be used in a network: link protection, node protection, Shared Risk Link Group (SRLG), and even SDLG (Shared SRLG Dependency Link Group).

The well-known notion of SRLG slightly increases the complexity of providing bandwidth protection compared to the simpler case of link protection since a set of links may simultaneously fail upon the failure of a link element (for example, a single fiber cut or DWDM component failure may lead to a number of link failures).

FBCM for SRLG protection makes use of a similar approach as for bi-directional links: if a bi-directional link *L* is protected with FBCM, the two backup tunnels *B1* and *B2* must be computed while ensuring that the sum of their bandwidth is taken into account in terms of backup bandwidth since the link failure would unavoidably lead to the simultaneous activation of both backup tunnels; in other words the bi-directional link should be treated as a set of two dependent unidirectional links, each requiring its own backup tunnel. This implies for both ends of the bidirectional link to be protected by FCMB to be coordinated when computing their respective backup tunnels. To that end, FCBM is extended so as to dynamically elect one end of the link as the PCE responsible for the computation of both backup tunnels so as to ensure that both backup tunnels are not treated as independent backup tunnels (a simple algorithm for the PCE election consists in using the smallest IP address). That approach is followed when combining link and node protection (when it is not

possible to tell whether the failure is a link or node failure), bidirectional links but also SRLG when a link belongs at most to one SRLG.

The case of a link belonging to multiple independent SRLG is slightly more complex:

Let's SRLG *S1* and *S2* be two SRLG made of links l_1, …, l_n and *S2* made of links l'_1, …, l'_m respectively

If $S1 \cap S2 \neq \varnothing$ (in other words there is at least one link l_k that belongs to both *S1* and *S2*) ***then***

- $<l_1 \ldots l_n> \cap <l'_1 \ldots l'_m> \neq \varnothing$ (SRLGs overlap)
- The computation of $<b_1 \ldots b_n>$ must the coordinated
- And the computation of $<b'_1 \ldots b'_m>$ must also be coordinated
- **Thus** the computation of $<b_1 \ldots b_n> \cup <b'_1 \ldots b'_m>$ must also be coordinated

End if

We say that the links l_1, …, l_n , l'_1, …, l'_m are SRLG dependent.

It thus results that if a link belongs to multiple SRLG (SRLG are not disjoint), two backup tunnels protecting independent links that are not SRLG diverse require their backup tunnel paths computation to be coordinated.

FBCM is then extended so as to regroup all links whose protection must be coordinated into Shared Dependency Link Group (SDLG) and consider SDLG as facility to protect by PCE, while considering the notion of aggregated bandwidth according which one SRLG fails at a time (single failure assumption) (not all backup tunnels protecting a given SDLG can be simultaneously active).

Furthermore, in some MPLS-TE enabled network, link bandwidth is even further divided in bandwidth pools in support of Diff-Serv aware MPLS Engineering (DS-TE), where bandwidth pools may either be strictly disjoint (a la TDM) or nested (also known as the Russian Doll Model (RDM)) and bandwidth is withdrawn from bandwidth pool according to their class type. For example, should bandwidth protection with FBCM be required on a per class-type basis, not only the bandwidth assigned to primary TE LSPs would be withdrawn on a per class type basis but the backup bandwidth pools would also follow the same bandwidth assignment strategy, which shows that the same bandwidth protection model can be used in such environments.

5.4.7 Set of already signaled bypass tunnels

We further extended the PCEP protocol to allow a PCC for providing a set of existing and signaled backup tunnels to a neighbor acting as a PCE with FBCM, should the node be a stateless PCE. Without knowing the information, a PCC may request the computation of a new set of backup tunnels (because the topology has changed, the amount of bandwidth to protect has been modified, …) that may lead to a completely different set of backup tunnels. By providing the set for existing backup tunnels that

has previously been computed, the stateless PCE can try to minimize the incremental changes to existing backup tunnels when computing a new set of backup tunnels.

5.4.8 Bandwidth protection and QoS scheduling

By contrast with IP IntServ ([60]) or ATM (Asynchronous Transport Mode), MPLS Traffic Engineering provides a bandwidth reservation model that is orthogonal to the actual QoS mechanisms in place (priority scheduling or use of congestion avoidance mechanisms such as Random Early Discard (RED) – see [61]).

Consequently, if FBCM is used to only protect a pool of bandwidth dedicated to a specific class type, this will allow for bounding the amount of traffic of the said class type on links where the scheduler has been configured to serve that class type to guaranty a required SLA. For example, if the class type Class Type (CT) *CT1* is used to protect primary TE LSPs carrying voice traffic, and bandwidth protection is only provided for these primary TE LSPs, in case of failure, the total amount of traffic of class type *CT1*, will never exceed the sum of bandwidth for *CT1* on the links traversed by the rerouted traffic plus the bandwidth of the rerouted TE LSP of class type *CT1*; if the scheduler along those links have been appropriately provisioned to provide the required SLA of traffic of class *CT1*, bandwidth protection is guaranteed.

Conversely, the rerouted traffic of other classes will be rerouted onto paths with no bandwidth guarantees, which means that the QoS experienced by these rerouted traffic flows but also the primary traffic of that class routed on those links may be degraded during failures.

5.5 Conclusion

With the emergence of critically time sensitive applications such as TDM (Time Division Multiplexing), video and voice over IP/MPLS, it became mandatory to design technologies providing extremely fast recovery upon link/SRLG and node failures such as MPLS Fast Reroute. Such protection technologies make use of local protection to reroute traffic within a few dozens of milliseconds. Additionally, rerouting along a path offering an equivalent Quality of Service (QoS) for some traffic became a must: this is also referred to as “bandwidth protection”.

In this thesis, a new technology that outperforms the naïve CSPF-based path computation approach, referred to as the Facility Backup Computation Model (FCBM) is specified in this chapter. FBCM allows not only for providing bandwidth protection for some traffic (when required) but also maximizing the degree of bandwidth sharing between backup tunnel protecting independent resources under what is known as the single failure assumption

The FCBM relies on:

- The configuration of a backup overlay network, a subset of the primary topology where each link capacity is equal to the amount of bandwidth allocated for backup tunnel,
- The signaling of backup tunnels with 0-bandwidth,
- The single failure assumption,
- The independent computation of backup tunnels for Node, SRLG, bi-directional links and SDG protection by each node acting as a PCE for the computation of backup tunnel protecting the given resource.

Referring to node protection, by using a distributed PCE-based approach where each node acts as a PCE to compute all NNHOP backup tunnels for its neighbors acting as PCC, the FBCM approach specified in this chapter allows for achieving the objective of enhancing MPLS TE that provides fast recovery with bandwidth protection for some types of traffic. Last but not least, the FBCM PCE-based model allows for minimizing the required amount of backup capacity in the network, while not requiring heavy signaling and non-scalable protocol extensions.

Part 3: Applicability of the PCE architecture to the Internet of things

6 Push-based packet and Event Inspecting (PPEI): A new PCE-based architecture for the Internet of Things

6.1 The Internet of Things (IoT) or Low Power and Lossy Networks (LLNs)

In this chapter, we explore the use of PCEs assisting in routing and traffic engineering decisions in what is sometimes referred to as Low power and Lossy Networks (LLNs) where a large number of highly constrained devices are interconnected with low power low data rates link layers. LLNs play a central role in the "Internet Of Things (IoT)".

We propose a new architectural model with several adaptations of the PCE architecture and the introduction of a new type of PCE, and study the overall applicability of PCE to the Internet of Things at large.

6.1.1 Introduction

The "Internet Of Things" (IoT) usually refers to networks interconnecting a new class of constrained devices such as sensors and actuators using the "TCP/IP" technologies that have been developed over the past three decades.

The term IoT might in many ways be misleading since the IoT does not just refer to "things" connected to the public Internet, but rather the use of IP technologies to connect these objects to either the public Internet of private IP networks. Furthermore, it is worth mentioning that the IoT is not limited to highly constrained objects but also refers to large number of cell phones connected to public networks using for example 3G/4G or LTE connections, high-end sensors in factories connected via industrial Ethernet links to controllers, …

What is a "Thing"? The term "thing" or "smart thing" usually refers to any device equipped with some processing power (e.g. a micro-controller or a micro-processor), memory ("Flash" and "RAM") and a communication module (wired and wireless). This class of device can range from a highly constrained device with a few Kbytes of memory, a AAA battery and a low-power wireless RF (Radio Frequency) module providing at best a few (dozens of) Kbits/s to a significantly more expensive device capable of computing Fourier Transforms, equipped with GBytes of memory and a high-speed Wifi connection. In general though, the IoT mostly refers to large-scale networks comprising a very large number of constrained devices, operating in harsh environments and requiring the network to provide tight Service Level Agreements (SLAs). Such networks are also referred to as "smart object networks" or Low power and Lossy Networks (LLN).

A detailed description of IP smart object networks covering the architecture, details on hardware, embedded software and lightweight IP protocols designed for these networks in addition to a number of detailed use cases can be found in [63]*.

The Internet of Things is undoubtedly one of the major next waves in networking and the Internet at large with a number of emerging applications such as energy (Smart Grids), water management, industrial automation, home and building automation, connected vehicles, intelligent transport systems (ITS), structural health management, healthcare to mention a few, and is already being deployed in several of these areas.

In this thesis we propose to make use of a PCE-based architecture specified in the previous chapters, with architectural and algorithmic modifications in support of the Internet of Things.

6.1.2 A short historical background

It is worth reminding that the first emergence of a few applications for the IoT a few years ago was mostly based on proprietary protocols and architectures. When it became obvious that such networks were to be interconnected to public and/or private IP networks, multiprotocol gateways were developed to interconnect these proprietary networks to IP networks. After years of "hard" technical discussions, it became obvious that multi-protocol gateways were ineluctably necessary so as to provide migration paths for already deployed non-IP or proprietary networks whereas true end-to-end IPv6 networks could be deployed for other verticals.

[63] covers in details the number of technical issues and limiting factors that arise when using multi-protocol translation gateways: 1) they are expensive and difficult to manage, 2) usually break QoS models and do not provide routing consistency and fast recovery consistency, 3) force down the path of the least common denominator, 4) they are clearly not an enabler for innovation, 5) expose networks to limited scaling without mentioning 6) the security issues of using such gateways.

6.1.3 IP Smart Object Network characteristics

In the context of this thesis, we focused on LLNs comprising a large number of (highly)-constrained devices interconnecting with low-speed links (on the order of a few (hundreds) of thousands of nodes) per routing domain. LLNs differ from "classic" IP networks in a number of ways that are briefly summarized as follows:

- *Links are low bandwidth* by contrast with links used in today's networks that provides several GBits/s of high-speed bandwidth. Links used in LLNs are usually low-bandwidth with at best a few hundreds of Kbits/s, sometimes providing higher bandwidth with specific technologies such as (low power) Wifi or Industrial Ethernet. In this chapter we focus on the applicability of PCEs to LLNs making use of low-speed links.
- *Link instability*: with current links (used in the Internet and private IP networks) such as Ethernet, SONET/SDH, Wifi to name a very few, Bit Error

Rates (BER) are typically extremely low (on the order of magnitude of 10^{-12}). Mostly because of the low-power nature of the links used in LLNs (such as IEEE 802.15.4([64]), the *BER* is very significantly higher leading to extremely low Packet Delivery Rate (PDR) or even offering intermittent connectivity. Figure 31 shows the PDR variation over time for a real-life deployed IEEE 802.15.4: it can be observed that the PDR varies between 60 and 100%, which makes those links particularly unreliable. It has been observed in real-life network that the packet delivery rate can be as low as 30%.

- *Limited processing power*: in contrast with typical routers used in "classic" IP networks, routers in LLNs are at best equipped with 32-bit micro-controller Unit (MCU), and often 8 or 16-bit, thus offering very limited processing power.
- *Limited memory*: in most cases, the memory (RAM) rarely exceeds a few dozens of Kbytes and in the very best case of a few hundreds of Kbytes.
- Last but not least, *"power"* is another limiting factor. Even when main-powered, limiting the power consumption is a must, which has a severe impact on protocols and operating system designs. This is why in most cases, MCUs provide the ability to enter in sleep mode to save energy (with various techniques to wake-up such devices), which has a number of consequences on protocol designs. Needless to say that "sleep" modes are even more critical when the device is battery-operated or powered thanks to energy scavengers.
- *Scale*: whereas large-scale core IP networks may have up to a few thousands of nodes in a routing area, LLN may comprise dozen of thousands of such nodes if not more.
- *Self-managed networks*: one of the main technical challenges in deploying LLNs relates to the absolute necessity to provide "autonomicity"; such network must be self-managed, they must allow for automatic provisioning without requiring any form of cumbersome manual configuration.

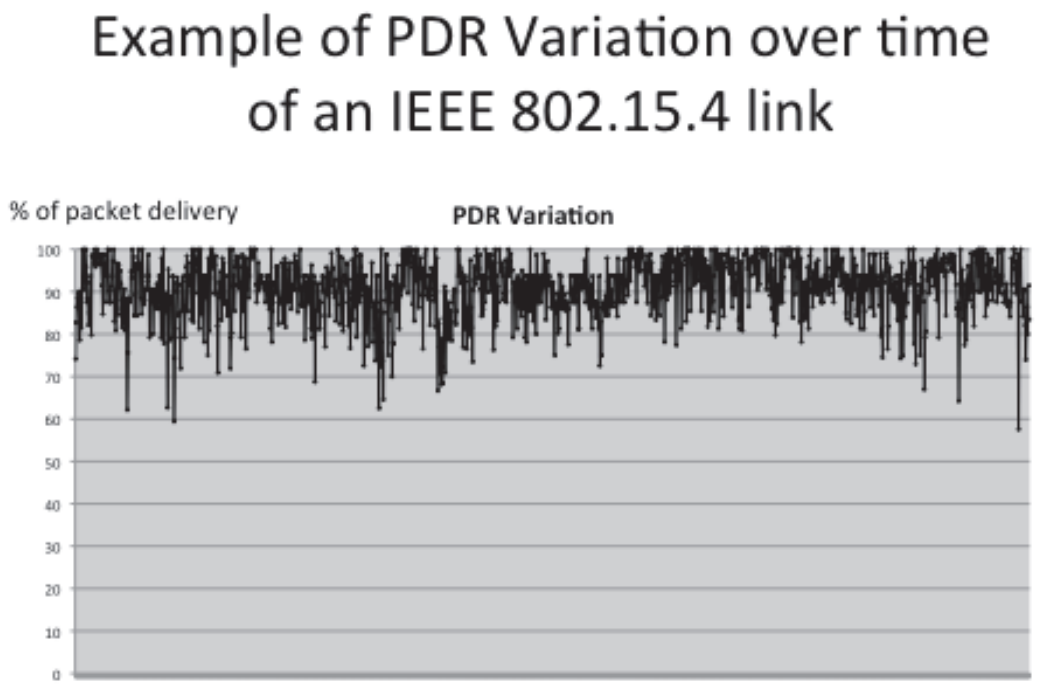


Figure 43 – *The Packet Delivery Rate of a low-speed IEEE 802.15.4 link*

6.2 A new PCE-based architecture for the IoT

6.2.1 From Smart object to Minimalist Connect Objects architectures

Over the past few years, tremendous progress have been made in terms of technology with new efficient low-power links such as IEEE 802.15.4 augmented with complex frequency hopping techniques so as to increase the overall bandwidth. Sophisticated techniques have been developed to minimize the probability for adjacent nodes to share frequencies in order to decrease collision probabilities.

Furthermore several new IPv6 protocols have been designed by the IETF for LLNs such the new lightweight resources management protocol called CoAP (Constrained Application Protocol) [65] and specified in the IETF Working Group named CORE (Constrained RESTful Environments, see [66]). Furthermore, a new routing protocol for LLNs and the Internet of Things called RPL (Routing for low Power and Lossy networks) has been designed by the IETF Working Group named ROLL (Routing Over Low power and Lossy networks [67]*).

Additionally hardware has been improved with the emergence of low-power consumption 32-bit micro-controllers offering dramatically increased processing power. Furthermore, this micro-controllers support a variety of power control management models and ways for device to operate in very low duty cycle mode of operation with periodic and/or event driven wake-up using preamble sampling, or

synchronized wake-up thanks to highly accurate network-wide clock synchronization techniques.

Supporting of all these technologies on a constrained device is now possible and various lightweight stacks have been developed, thus making these devices *smart objects*. That being said, it can be observed that the accumulation of complex technologies leads to a non-linear increase in complexity in terms of understanding, configuration and troubleshooting of these networks. Considering that these networks are operated in harsh and non-attended environments, a tight control of the overall system complexity is in order and great attention must be given to not reach an inflection point beyond which operating these networks would simply become too complex.

Complexity of such large-scale systems is a major research area, and it can be empirically observed that such a complexity is not linear and tends to grow almost linearly up to an inflection point where the level of understanding and consequently the ability for these systems to be operated, configured and understood collapses, as illustrated on Figure 32.

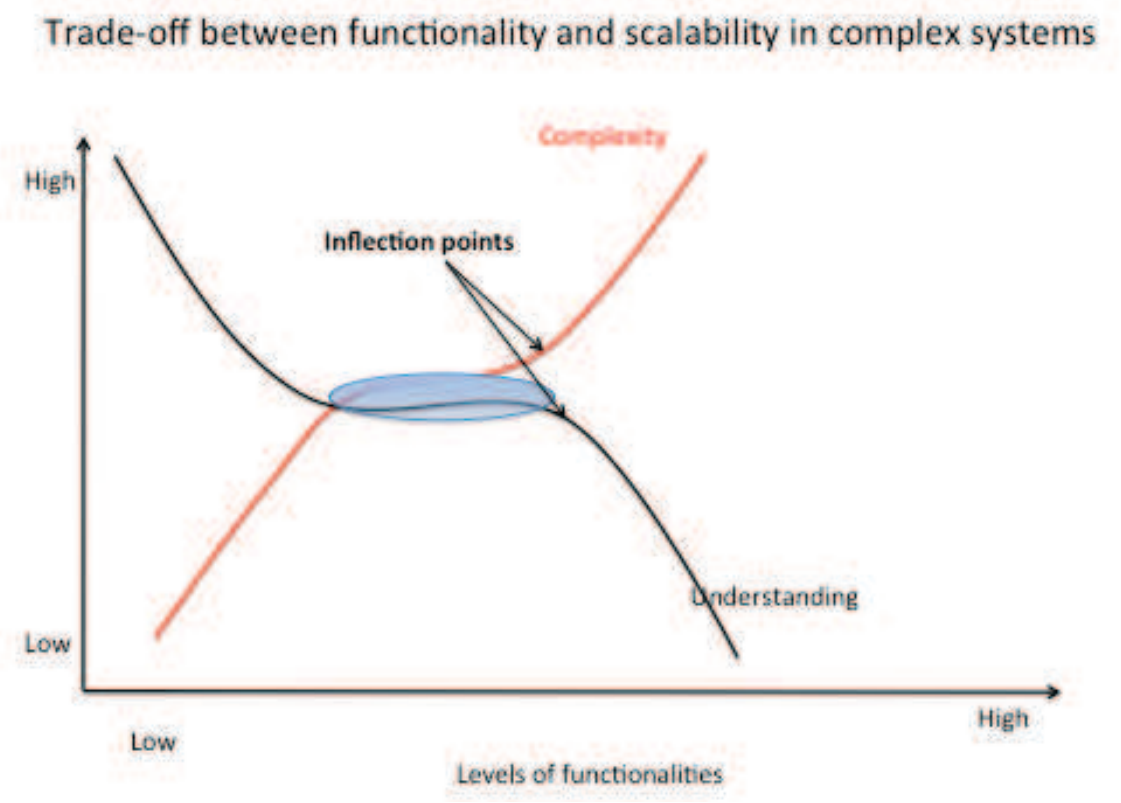


Figure 44 – *Large-scale system complexity increase as a function of the number of supported functionality (algorithms, protocols)*

In this thesis, we propose a radically different model moving from smart object to Minimalist Connected Object (MCO) that implies to rethink LLNs' network architectures where the PCE undoubtedly plays a central role. Figure 33 shows the current model where millions of Smart Objects are connected via a Low power and lossy network Border Router (LBR) located at the fringe of the IoT.

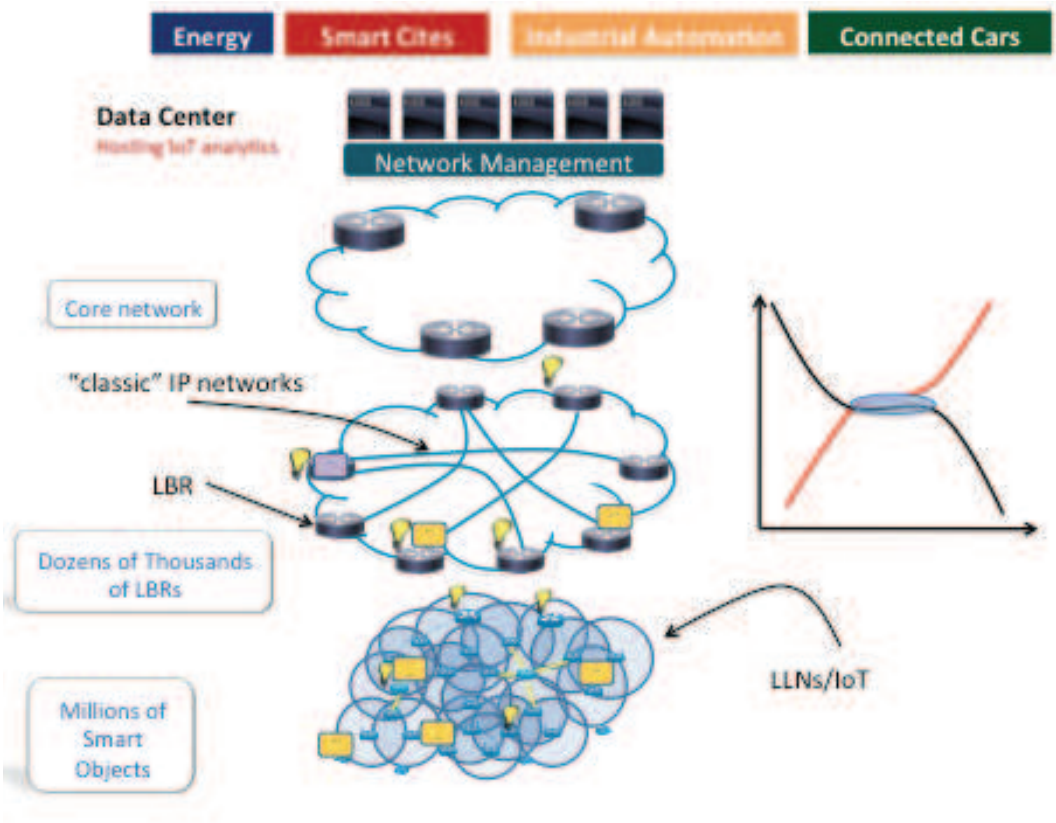


Figure 45 – *Classic IoT architecture interconnecting IP Smart object network to IP core networks via LBR*

6.2.2 Functional blocks and architecture

In our research, we define a set of functional building blocks thus defining a new PCE-based architecture, illustrated on Figure 34:

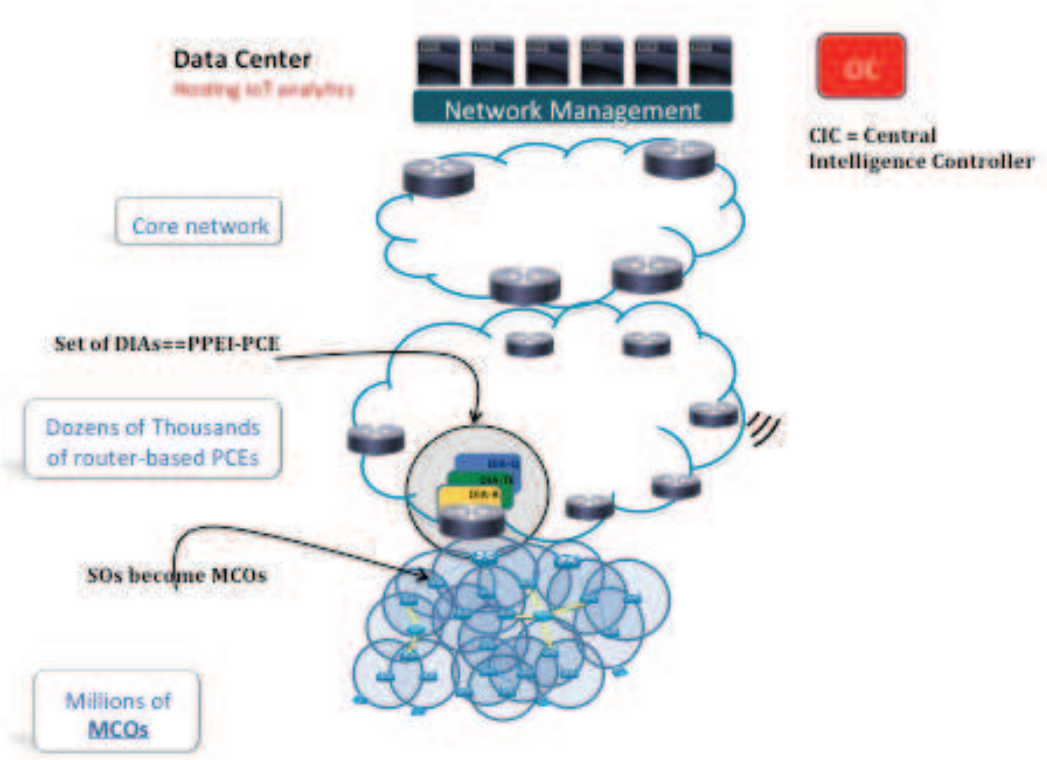


Figure 46 – ***Representation of a new PCE-based Architecture made of MCOs, DIAs (PPEI-PCEs) and CICs***

Our research led us to re-think the overall LLN network architecture (with changes to the "End-to-end principle" that has been governing several design principles of the Internet over the past three decades (see [68])). We introduce several components in the networks:

- **Minimalist Connected Object (MCO)**: by contrast with the current trend, a MCO is defined as a "minimalist" connected object supporting a minimum subset of functionalities such as security, a de-generated non-optimized form of routing providing connectivity (see next chapter), agents capable of providing state information to both the DIA and CIC described below, with no form of sophisticated QoS management (at least unless downloaded from an intelligence agent), heavy and complex processing protocols, traffic engineering, call admission control or even policy.

- **Distributed Intelligence Agents (DIA)**: the DIA is a software/hardware module hosted on the LBR responsible for a number of tasks, some of which being currently supported on Smart Objects. This leads to shifting a number of complex networking and application processing tasks from smart objects to DIAs. Although DIAs in this architecture support a number of networking

tasks as such routing and Traffic Engineering, other specialized DIAs could be dedicated to the support of other applications and networking tasks such as a dynamic push-based QoS model triggered by observed performance metrics, and a number of forms of network management and data aggregation using techniques such as data fusion/aggregation, alarm correlations, traffic pattern observations using learning machines to mention a few. In the context of this thesis, we limit our scope to the issue of routing and traffic engineering.

- **Central Intelligence Controller (CIC)**: The CIC is a central component of this new architecture that performs more complex tasks such as collecting a wide set of network performance metrics in the systems, collecting user-based SLAs and policy rules, aggregating NMS-related data for network behavior analytical tasks and performing complex networking and application based data processing thanks to heavy analytics.

Signaling: as illustrated in Figure 34, in this new PCE-based architecture, signaling takes place between many of these functional elements:

- PCC-PCE signaling: the use of PCEP for signaling between a MCO acting as PCC and a DIA (PCE) is unfortunately ill-suited to LLNs. PCEP has been designed for highly capable devices operating in high performance networks and would unavoidably be too bandwidth and processing consuming on constrained devices interconnected by low-bandwidth low-power links. Furthermore, PCEP is TCP-based, which is not suited to lossy networks: the back-off algorithm for packet loss handling with TCP does not perform well in highly lossy network where the PDR can be as low as 60% when not even lower. A lightweight signaling protocol must be defined that is outside of the scope of this thesis; in extreme forms, piggybacking of control plane information in user data packet could be used in order to avoid unnecessary overhead, thus making the signaling between PCC and PCE implicitly replaced by other techniques such as Deep Packet Inspection (DPI – see [69], [70]).

- PCE-PCE: signaling between DIAs hosted on various LSR/LBR in the network is used to synchronize, gather and exchange network views, performance metrics and network states with the objective to perform global network optimization and traffic engineering. Furthermore, DIAs will undoubtedly interact with the CIC in order to retrieve output of complex heavy analytics related to the network performance but also user-defined policy rules, in addition to computed and observed global SLAs. Since DIA and CIC will be hosted on more capable nodes interconnected by higher bandwidth links, the PCEP protocols augmented with extensions is suitable.

6.2.3 A Push-based Packet/Event Inspecting (PPEI) PCE-based architecture

In this thesis, we propose a new model taking advantage of the PCE-based architecture with a set of architectural modifications to the PCE protocols and algorithms to make them applicable to the Internet of Things.

First, ***if*** DIA_i is defined as a DIA in charge of performing a task *i* (e.g. routing, traffic engineering, QoS analysis, …), ***then*** for *i*=<1 … *n*> Sum of DIA_i= PCE

In other words, the collection of DIAs can be seen as a PCE handling a set of functions removed from smart objects (PCC) (that becomes MCOs) and shifted to DIA (PCE). In the context of this thesis, we limit our scope to two DIAs in charge of the routing and Traffic Engineering functions in the network that are described in the next chapter.

6.3 Conclusion

We have elaborated a sophisticated architecture called the PCE architecture and a set of protocols and algorithms to off-load the routing computation of Traffic Engineering LSP on PCEs in order to solve a number of problems that were so far unsolved.

In this last part dedicated to the use of the PCE architecture in highly constrained environments (LLNs) we have proposed several modifications of the current Internet Of Things architecture to apply a PCE-based path computation and traffic engineering approach thanks to the use of a new type of PCE referred to as PPEI (Push based Packet Event Inspecting) PCE. The mode of operation of a PPEI-PCE is described in the next chapter in the context of assisted routing in LLNs.

7 Routing and Traffic Engineering in PPEI PCE-based architecture

7.1.1 Introduction

During our research, we explored and designed a new path computation architecture, several protocols and algorithms for the computation of MPLS TE LSP referred to as the Path Computation Element (PCE).

In the previous chapter, we designed a new architecture (not limited to the routing aspects) that dramatically changes the Internet of Things architecture by shifting a number of networking and data processing functions to DIAs hosted on LBRs, where the collection of DIAs is a PCE. The DIA-R, in charge of the routing in LLN, is a PCE of a new type referred to as Push-based Packet/Event Inspecting PCE (PPEI PCE).

7.1.2 Use of PCE based computation in non MPLS networks

First, in this architecture, paths are computed for IP not MPLS; this does not change the architecture and only requires minor extensions to signaling protocols. Furthermore, it is envisioned that LLNs will at some point make use of label switching. Note that PCE-based architectures are also used in classic networks for the computation of IP non-MPLS based paths.

Even though PPEI-PCE could be used to compute IP paths (and not just TE LSP), it is worth pointing out that there are several motivations in using label switching in LLN (note that the terms LLN and IoT are used interchangeably):

- **Support of VPN and Traffic Engineering**: LLNs are similar to "classic" (less-constrained) IP networks peers where the support of Virtual Private Networks) VPNs for traffic isolation (a slightly different motivation as with IPv4 since LLNs are IPv6 only) and traffic engineering are highly desirable. Furthermore, the ability to engineer the LLN traffic is a must considering the degree of constraints and the need for a careful use of scarce bandwidth still while guaranteeing tight SLAs.
- **Routing size header**: packet header size is undoubtedly an issue in LLN where packet overhead is undesirable, especially because the payload is often extremely small, thus increasing the overhead especially with 40 bytes IPv6 headers (not mentioning IPv6 optional header or the need to perform IP tunneling in a number of circumstances). Compression techniques such as IPHC ([71]) have been developed that allow for the significant compression of IP headers. Still short headers such as labels could be used in order to further reduce the packet header overhead especially in presence of non-compressed and/or stacked headers (e.g. in the case of tunneling).
- **Reduction of routing tables stored in routers**: IP routing fundamentally relies on the use of routing tables in IP routers that are fed by routing protocols. The growth of such routing tables must be handled with great care

on constrained nodes such as smart objects and MCOs since the routing table may become a limiting factor on nodes where memory constraints are high. A number of techniques and algorithms can be used in order to bound the routing table size: 1) The IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) (discussed later in this chapter) proposes a non-storing mode where the traffic makes use of default routing in the upstream direction up to a more-capable node (usually the LBR) that redirects the traffic thanks to source routing and IPv6 tunneling. This allows for avoiding the requirement of having to store destination-based routes on intermediate nodes (only default routes are required) 2) Sophisticated IP aggregation techniques can be used to limit the routing table size by ensuring that nodes that belong to a sub-tree of a node *X* share the same subnet/prefix; this allows for storing a limited number of routing entries on each node at the cost of increasing the complexity of IP address assignment mechanisms on each node (especially in moving conditions where nodes move to other sub-trees because of mobility or preferred (less-expensive) paths forcing to potentially re-allocate a new address without traffic disruption).

The use of LSPs would help reduce the routing table size by only requiring the storage of label switching tables, the size of which could be even further reduced thanks to label stacking with hierarchical tunnels.

Consequently, it is envisioned that label switching techniques will eventually arise in LLNs, and new signaling mechanisms may be specified for label distribution such as the routing protocol itself by contrast with "classic" IP networks using protocols such as LDP [72] or RSVP-TE [11] (although undoubtedly too expensive in terms of control plane overhead and processing at least in their current form).

Still, although the PCE architecture has been designed for MPLS TE LSP, it is not tied to label switching and could be used for IP, non MPLS-enabled networks, which both apply to "classic" IP networks and LLNs.

7.1.3 Overview of RPL, the new routing protocol for the Internet of Things

After several years of work and careful analysis, the IETF has formed a working group called ROLL ([67]) in charge of designing a routing solution for LLNs in light of their unique routing requirements and technical characteristics.

It was found out that none of the existing routing protocols would meet the requirements (see [73], [74], [75], [76]) and specifying a new routing protocol for the IoT was required, which led to the specification of RPL ([77]*, [78]*, [79], [80]*, [81]*, [82]).

RPL has been specified with a number of constraints and network characteristics in mind:

- Node constraints in terms of memory and processing capabilities, but also power, especially in networks comprising battery-operated nodes,

- Support of a new set of routing metrics including dynamic link and node routing metrics (by contrast with other routing protocols that are usually limited to static link metrics), where each metric can be used both as a routing optimization metrics and/or a routing constraint. RPL routing metrics are specified in [78]*,
- Low-speed, low-power link layers, requiring to bound the control plane traffic required for routing,
- Large-scale networks potentially comprising dozen of thousands of nodes in a routing domain,
- Network instability in terms of link error rates, low packet delivery rates, presence of highly unstable links potentially offering intermittent connectivity, nodes "dying" in the network when running out of energy (if battery operated or powered with energy scavengers) or nodes in sleep modes.

Since this thesis is not related to RPL but the use of a new approach based on PPEI PCEs combined with a degenerated form of distance vector routing that could be based on the RPL protocol, we only provide a very high level overview of RPL hereafter. The detailed specification of RPL can be found in [77], with high level overview in [83]*. The RPL routing protocol has been evaluated in details in [84] and [85].

RPL is a **proactive** distributed distance vector routing protocol that computes Destination Oriented Directed Acyclic Graph (DODAG) using an Objective Function (OF) that is used by each node in the network to select its preferred next hop along the DODAG according to a set of metrics and constraints. The DODAG is a directed acyclic graph with no directed cycles (for any vertex *v*, there is no directed path that starts and ends at *v)*.

An OF may be as simple as "*optimize paths to minimize the number of hops*" or may take a significantly more complex form "*Find the shortest path according to a polynomial function of the ETX and BER metrics while avoiding links with a link affinity of Lx, a minimum link reliability level of Lv and comprising nodes of type Tn where Tn refers to battery operated node, with a hierarchical organization of a set of constraints (relax constraints in a specified order if no path can be found)*."

DODAG are built from the DODAG Root (e.g. an LBR) that multicasts ICMPv6 control plane packets (see [86] for the specification of the ICMP protocol). DIOs propagate in the network and ar used to build the routing graph and topology. RPL specifies four messages types using ICMPv6 called the DIO (DODAG Information Object), DAO (DODAG Destination Advertisement Object), and DIS (DODAG Information Solicitation) messages. DIO messages are disseminated according to the trickle routing algorithms ([87], [88] and [89]). DIO messages are used to build a DAG for upstream traffic, and DAO messages are sent by the nodes in the network for prefix advertisement.

RPL supports two modes of operation:

- *The storing mode* where nodes send DAO messages to their parent in the DAG to populate their routing tables for downstream routing,

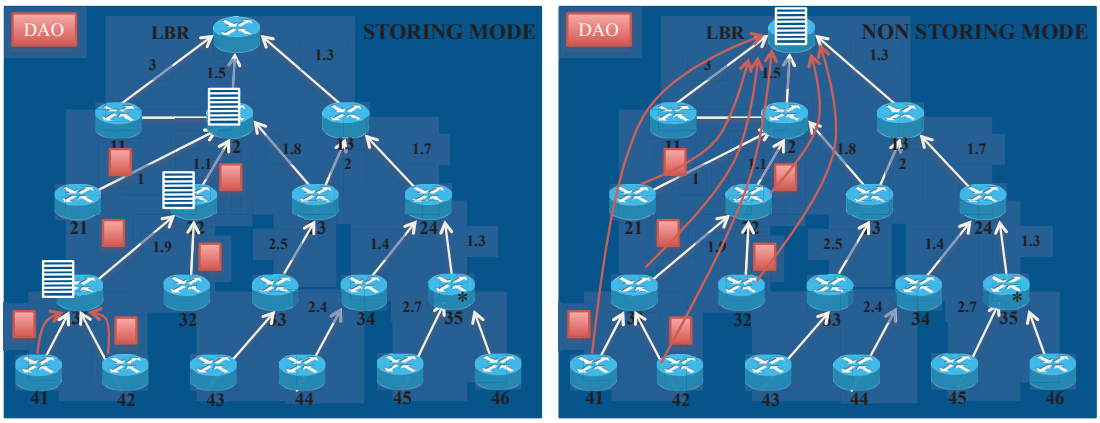
- *The non-storing mode* used in network with highly constrained nodes in terms of memory where DAO messages are sent to the DODAG root that stores downstream routes. Traffic is sent upstream following the default routes along the routing graph (DODAG) built by RPL up to the DODAG root, which in turn inspects its routing table populated by DAO messages to determine downstream routes. Once the downstream route has been computed, packets are source-routed (and tunneled) thanks to IPv6 routing header specified in [72] to their destination, thus without requiring the storage of routing tables for downstream routing on intermediate nodes. A similar mode of operation is adopted for packets originated outside of the LLN and destined to nodes within the LLN.

Figure 35 illustrates the two modes of operation. In the non-storing mode for example, DAO messages providing information on neighborhood and IP prefix reachability are sent to the DODAG root, which builds the routing table.

Since DIO messages are sent according to time variable schedules computed by the Trickle algorithm, a node attempting to join the network may have to wait for an unacceptable period of time. The DIS message allows for circumventing this issue by triggering the sending of DIO messages by nodes' neighbors (potentially filtering the set of nodes for which a DIO message is required in order to avoid packet message storms, which are highly undesirable in shared media such as wireless networks). Furthermore, replies are subject to jittering to limit the probability of collisions.

Populating the routing tables using RPL DAO message

- Two modes of operations: storing mode and non storing modes



Unicast to DAO parents

Unicast to DODAG Root (not processed by intermediate nodes)

Figure 47 – *Illustration of the RPL non-storing mode of operation*

Back to the figure 35 should be node *41* requires sending a packet destined to node *24*, the packet would be routed hop-by-hop along the DODAG (*41-31-22-12*) up to the LBR (DODAG root). As this point the LBR consults its routing tables populated by the DAO messages received by each node in the network, and computes the downstream route to the destination (*LBR-13-24*). The packet is then tunneled with a source route equal to *LBR-13-24*.

Distance vector routing protocols are known for their inability to effectively avoid routing loops. As a matter of fact, micro-loops even exist in link state routing protocols and may take place because of the propagation time of LSAs and events that may take place in specific order; that being said, these loops are called micro-loops (discussed in [90]) because of their short duration and recent research has shown that ordering of distributed events could be controlled so as to completely prevent such micro-loops to take place. This is not the case with Distance Vector. RPL does not guarantee the absence of loops but specifies several rules so as to either minimize their size and/or their probability of occurrence. Note that control plane mechanisms for loop prevention usually implies expensive control plane mechanisms so a careful trade-off had to be found for routing in LLN considering the constrained nature of these networks. Consequently, RPL has specified a loop detection mechanism based on data path validation that allows a node to detect a loop and break it.

Two techniques are defined in RPL for route repair: Global repair consists in rebuilding the entire DODAG (thus also allowing for reoptimization of the graph) and that is exclusively triggered by the DODAG root upon the expiration of a global timer or more sophisticated event-driven rules. The second repair techniques called "Local repair" consists in modifying the routing topology locally: when a node looses routing connectivity (adjacency to all parents), it locally repairs the DODAG after poisoning its sub-DAG (local repair may lead to less-optimal paths). The notion of "poisoning" a sub-DAG refers to the sending of a message by a node that has lost upstream connectivity to all of the nodes in its sub-DAG informing them to move to an alternate sub-DAG. Both techniques are orthogonal and complementary.

New IPv6 headers have been defined in [71]* and [72]* used in RPL-enabled LLNs for source routing, data path validation in addition to other routing functions in the network.

RPL supports Multi-Topology Routing (MTR) a concept according which a set of logical routing topologies are built on top on a specified physical network used to carry traffic according to their requirements in terms of SLAs (e.g. a Routing Topology RT_i is built to minimize delay to carry time-sensitive traffic where RT_j maximizes path capacity in terms of bandwidth while avoid battery-operated nodes to carry heavy non time sensitive traffic). Note that the concept of MTR had been introduced by other routing protocols such as OSPF or ISIS (see [91] and [92]).

7.1.4 Functional description of the DIA-R, a PPEI PCE used for routing in the Internet of Things

In our research, we propose a radically different routing model paradigm based on a mixed approach of distributed routing combined with PCE-based centralized routing.

The DIA-R (Distributed Intelligent Agent – Routing) is a PCE of a new type called Push based Packet Event Inspecting (PPEI). A PPEI PCE is a PCE in the sense that it performs path computation similarly to all PCEs, should the path be an IP or label-switched path. One of the core properties of a PPEI PCE lies in that it does not expect a signaling path computation request by contrast with a PCE receiving PCEP PCReq messages from PCCs. PPEI PCE may rather performs traffic (packet) inspection and event correlation (thus the term *Packet Event Inspecting*) in order to trigger the computation of a new path in the network before updating routing tables in the network's nodes. Note that PPEI PCEs may support lightweight signaling in the future.

Such an approach fundamentally differs from distributed and exclusively centralized routing since routing computation is based on incremental PCE-driven changes triggered by the PCE with combined distributed routing. Such triggers are

driven by observed user and control plane traffic in conjunctions with policy based or dynamically learned event rules used in conjunction with network performance metric gathered from the CIC such as SLA monitoring, output from a heavy analytic engine or by other means.

For the sake of illustration, each node in the network may use an over-simplified routing protocol and rely on the PPEI-PCE inspecting the network topology, packet flows, network performance to determine whether or not routing changes must take place on the routing topology to satisfy the SLA. The PPEI-PCE then becomes responsible for performing incremental routing adaptations when and where required.

We propose hereafter an algorithm but many variants could be implemented with such an architecture.

Algorithm

Let's G be the physical network topology G=(V,E) comprising a set of $|V|$ vertices (routers) and $|E|$ edges (links) (ordered 2-element subset of V).

A Routing Topology (RT) is a subset of G where each order pair $(n_i,n_j) \in E$ represents a link of RT between adjacent routers as determined by the routing protocol (note that two nodes may be neighbors without any routing adjacency: none of them is using the other one as a next hop router for any given path).

For a given network G, there is a finite set n of RT_i <RT_1 …. RT_n> defined by the set of oriented links $Li \in E$.

State collection engine: the aim of the state collection engine is to gather routing network topology information, thus a subset S of G=(V,E) (the reason why the gathered topology information is a subset of G is because partial information related to the graph edges may be available or even detected by protocols running in the network). To that aim, protocols such as IPv6 Neighbor Discovery can be used, augmented with link metric information collected by remote MCO. Link metric information is critical and can be locally computed by MCO using simple algorithm averaging out metrics such as the ETX using a low-pass filter, based on lower layer information such as the percentage of acknowledged packets or the link RSSI. MCO may filter out specific links not satisfying local condition (level of stability not exceeding predetermined threshold). Network topology information is then signaled by MCOs to the DIA-R using the degenerated routing protocol discussed hereafter or via unicast messaging. In others words, the state collection engine is responsible for gathering information related to the connectivity between each pair of nodes $(n_i,n_j) \in E$.

Signaling messaging in PPEI PCE architectures applied to LLNs is illustrated in Figure 35.

Step-1: Build *MinRT(G)*

Step-1 consists in using an over-simplified proactive Routing Protocol (RP) building a DODAG called $MinRT \in <RT_1 \ldots. RT_n>$

RP is an oversimplified proactive distance vector routing protocol specifically designed for MCOs. By contrast with RPL, RP does not support any form a complex regular expressions used by OF for parent selection and sophisticated parent selection algorithm that may take into account complex events such as historical of the link behavior, hysteresis or other link metric and or constraint. RP can be derived from RPL by keeping fundamental components such as the trickle algorithm for control packets dissemination used to build the *MinRT* graph but complex processing rules to optimize path computation are removed. Similarly although loop detection mechanisms based on packet header inspection (data path validation) can be kept, loop resolution is left to the DIA-R and MCO simply notifies the DIA-R without resolving (breaking) loops. Should an oversimplified version of RPL be used for RP, network topology information gathered by the state collection engine can be performed by using RPL DAO messages augmented by the required set of link metrics.

Step-2: Building OTR(G)

Let's Cost(*RTi*) be the total cost of *RTi* according to some objective function. The objective function used by the PPEI-PCE could be as simple as minimizing the cost of RTi where:

$$\text{Cost(RTi)} = \sum_{k=0}^{|E|} cost(Lk)$$

By contrast, the objective function may be significantly more complex consisting of computing the shortest paths satisfying a set of constraints $<C_1 \ldots C_n>$ while minimizing a set of path costs in the network and bounding the path cost difference between pair of paths (a fairly common requirements when duplicating traffic along diverse paths). In this algorithm we use the OF according which the cost of *RTi* is defined as the sum of the cost of its edges (as in the previous formula).

We define *OTR(G)* such that:

$OTR(G) = Min$ for j=1…| RTi)| ($\sum_{k=0}^{|E|} cost(Lk)$)

Properties

$OTR(G) \in <RT_1 \ldots. RT_n>$

Cost(*MinRT*) ≥ Cost (*OTR(G)*)

The architecture does not preclude the DIA-R acting as a PCE from using any form of complex NP-complete algorithm according to the OF or simpler algorithm such as the Dijkstra algorithm whose complexity is $O(|E| + |V| \log |V|)$ using Fibonacci heap (by contrast with the original implementation whose complexity was $O(|V|^2)$) . Even

more sophisticated versions of the Dijkstra algorithm such as incremental SPF can be used to further reduce the running time complexity.

Note that the fundamental architectural characteristic of the DIA-R is that it acts as a PCE capable of using the subset S of $G=(V,E)$ gathered by the state collection engine to compute $OTR(G)$ according to the OF.

At time $t=0$, $RT(t)$ is defined as the Routing Topology used in the network at time t where $RT(t) \in <RT_1 \ldots. RT_n>$, $RT(t0)$

Loop (infinite loop)

Let's $\Delta(t)$=Cost($RT(t)$)-Cost($OTR(G)$)

Let's $\Delta_{max}(t)$ be the maximum tolerable path cost difference between the current routing topology and the optimum graph $OTR(G)$ computed by the PCE (DIA-R).

Let's pc_i be the path cost of a path p_i between two vertices u and v where u and $v \in$ V, $|p_i| \leq |E|$[2]

> ***If*** $\Delta(t)$=Cost($RT(t)$)-Cost($OTR(G)$) > $\Delta_{max}(t)$ ***or*** ∃ an integer i such that pc_i does not satisfy the SLA provided by the CIC for p_i ***then***
>
> > ***Repeat until*** $\Delta(t)$=Cost($RT(t)$)-Cost($OTR(G)$) < $\Delta\_max(t)$ ***or*** ∃! an integer i such that pc_i does not satisfy the SLA provided by the CIC for p_i
> >
> > > Compute RT_j so that $\Delta(t) < \Delta\_max(t)$ and pc_i satisfies the SLA according to OF' called the improvement objective function
> > >
> > > > ***For*** each impacted node resulting from RT_j
> > > >
> > > > Send a unicast message (push mode of the PPEI PCE) indicating the change of preferred next hop selection
> > > >
> > > > ***End For***
> > >
> > > Arm a timer T_w
> > >
> > > ***Wait*** until expiration of timer T_w
> > >
> > > Compute $\Delta(t)$=Cost($RT(t+T_w)$)-Cost($OTR(G)$) and evaluate whether ∃ an integer i such that pc_i does not satisfy the SLA provided by the CIC for p_i
> >
> > ***End Repeat Until***
>
> ***End if***

End Loop

The algorithm specified above performs incremental centralized path computation by the DIA-R until a routing topology RT_i is found in the network, thanks to the collection of networking and routing states collected by the state collection engine, but also inputs from the CIC in the form of a series of events and network performances metrics.

7.2 Traffic Engineering in PPEI-based networks

In this section we explore the use of the PCE architecture to perform traffic engineering in LLNs thanks to the use of PPEI-PCE implemented on a DIA-TE module hosted on LBR considering the specific requirements of LLNs. Traffic Engineering is undoubtedly a critical components of the IoT where bandwidth is extremely scarce in fast growing traffic demand, while still careful care must be given to the control plane overhead of TE techniques.

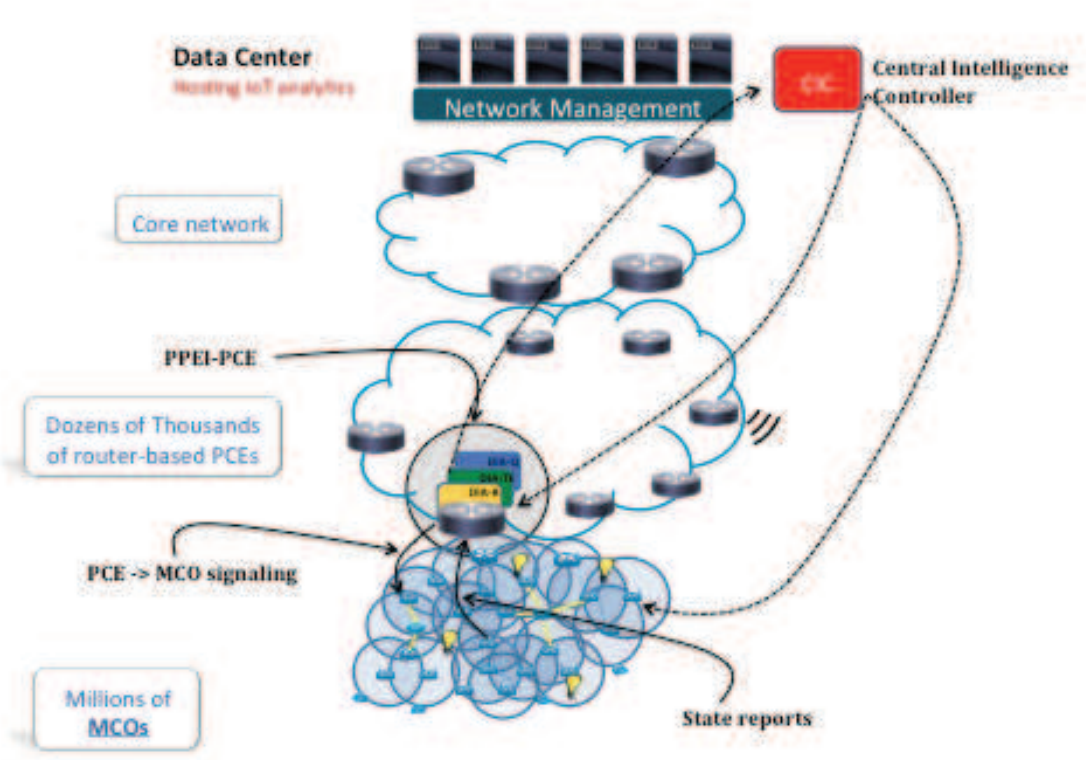


Figure 48 – ***Signaling messaging in PPEI PCE architecture in LLNs to perform routing and traffic engineering***

Algorithm used by the DIA-TE for traffic engineering in LLN using a PPEI PCE-based approach.

Step-1: **Traffic matrix computation**:

Statefull PCE is an example where the knowledge of the traffic matrix to globally optimize a set *S* of primary TE LSPs is required; by contrast, traffic profile from a node $k \in V$ to all nodes $n \in \{V\}\text{-}k$ is sufficient in the case of distributed CSPF path computation at the cost of a less optimal primary TE LSP placement.

Considering the current constrained of the nodes residing in LLNs, these networks are not expected to be TE-enabled.

Thus we specify a different approach where in routed LLNs, the DIA-TE engine hosted on the LBR collects the traffic matrix thanks to various technologies such as Deep Packet Inspection (DPI) since most of the traffic transit via the LBR. For the traffic not transiting through the LBR, also called the Point-To-Point traffic, when the traffic rate from a node *k* to a node *k'* exceeds a pre-configured threshold, a message may be generated by the node *k* that sends as a unicast message to dynamically update the LBR hosted PPE-PCE (discovered via IGP extensions).

Such a mechanisms allows the LBR for computing the traffic matrix M(t), which is represented as a matrix $M(t)$ such as:

$$M(t) = \begin{pmatrix} 0 & \dots & tr(1,|V|) \\ & tr(i,j) & \\ tr(|V|,1) & \dots & 0 \end{pmatrix}$$

Where *tr(i,j)* is the averaged traffic (alternatively peaks of traffic may be used) from node *i* to node *j*

At this point, the DIA-R applies a set *F* of filtering rules used by a function f() to filter out each entry *tr(i,j)* of the matrix *M(t)*.

For example:

- *tr(i,j)*==0 if *tr(i,j)*<T_{Min} (minimum amount of traffic to trigger traffic engineering)
- *t(i,j)*==0 if DSCP(inspected traffic) ∈ <*DCSP1*, ..., $DSCP_k$> (only the traffic marked with specific Diffserv Code Point ([93]) are considered for Traffic Engineering)
- …

Thus, the DIA-R computes: $M'(t)$=f$(M(t))$=f($\begin{pmatrix} 0 & \dots & tr(1,|V|) \\ & tr(i,j) & \\ tr(|V|,1) & \dots & 0 \end{pmatrix}$=

$\begin{pmatrix} 0 & \dots & tr'(1,|V|) \\ & tr'(i,j) & \\ tr'(|V|,1) & \dots & 0 \end{pmatrix}$

Where f() is the filtering function.

Note that the use of other techniques such as polling of Management Information Base (MIBs) via a protocol such as SNMP ([94], [95], [96]) is ill-suited for LLNs because of the amount of generated traffic and memory footprint required by the protocol. This justifies the use of DPI techniques that could either apply to the whole traffic or may alternatively be applied to selective samples using traffic sampling techniques and processing of CoAP packets.

Step-2: correlation of $M'(t)$ and $RT(t)$

The DIA-TE engine then correlates $M'(t)$ and $RT(t)$ that is made of link $L_{u,v}$ where $u,v \in V$.

Let's $|RT(t)|$ be the number of links in $RT(t)$. $|RT(t)| \leq |V|$

Let's call f'(.) the function allowing to compute the average link load on each link $L_{u,v}$ where $u,v \in V$.

The DIA-TE first gathers the routing topology $RT(t)$ from the DIA-R routing engine and then retrieves for each link $L_{u,v}$ where $u,v \in V$ the respective links bandwidths called $B(L_{u,v})$.

$B(L_{u,v})$ is typically dynamic is LLN (this is for example the case of IEEE 802.15.4 or IEEE P1901.2 low-power links that rely on the physical layer characteristics; the link bandwidth and thus the available throughput vary with the network conditions. In order to compute $B(L_{u,v})(t)$, network probes are randomly generated to all nodes in the network (marked as low priority traffic) and the values $B(L_{u,v})$ are recorded for each traversed link along the path. Gathering such data allows the DIA-TE to compute the dynamic list of all bandwidth links in the network:

B_{All}==<B_1, …., B_T> where T=$|RT(t)|$ and B_i=link_bandwidth($B_i(t)$) where $B_i(t) \in RT(t)$

Other link layers techniques may be available for the computation of B_i; this is the case for example when a central controller performs radio slot assignments.

We then define the function f'() that correlates the traffic matrix, the routing topology computed by DIA-R and the set of computed link bandwidths in the

network to determine links loads called LL_i for each link $B_i = B(L_{u,v})(t)$ where $B_i \in RT(t)$:

$$<LL_1, \ldots, LL_{|RT(t)|}> = f\left(\begin{pmatrix} 0 & \ldots & tr'(1,|V|) \\ & tr'(i,j) & \\ tr'(|V|,1) & \ldots & 0 \end{pmatrix}, \mathrm{RT(t)}, <B_1, \ldots, B_{|RT(t)|}> \right)$$

A list of congested links $E \subseteq <LL_1, \ldots, LL_{|RT(t)|}>$ is then computed that comprises the set of links that experience (long-lived) congestion and requires Traffic Engineering.

Note that complementary techniques can be used to flag links that belong to $<LL_1, \ldots, LL_{|RT(t)|}>$ to be added to the E list using ICMP messages or any other form of links congestion detection. For example, the routing protocol may carry additional information related to averaged link loads computed by the nodes in the network; a new TVL could be added to the DAO messages sent to the DODAG root in the example of RPL. In addition, the Explicit Congestion Notification (ECN) technique as defined in [97] and [98] could be used by the DIA-TE module to identify congested areas.

Step-3: Routing changes trigger

In this step, the DIA-TE module sends the list E of congested links to the DIA-R module that in turn modifies the routing table adjacencies where appropriate in the network according to the algorithms described in Section 7.3.

Note that should $RT(t)$ not be available, the DIA-TE engine may exclusively make use of congestion detection and notification techniques to compute a subset of the E list and still apply routing modifications thanks to the DIA-R module as a partial measure to alleviate the level of congestion where detected in the network.

Lastly, the DIA-TE module optionally may arm a timer used to pro-actively poll the set of links in the list E following the routing modification to make sure that the congestion has been released or a least been reduced. If not, an additional feedback is provided to the DIA-R engine for further actions.

7.3 Conclusion

Thanks to the new PPEI PCE architecture specified in Chapter 6, most of the complexity is removed from end devices called Smart Objects (SO) that become Minimalist Connected Objects (MCO).

In this chapter, we show how these smart objects are interconnected thanks to an oversimplified routing protocol that could itself be derived from existing routing protocols designed for LLNs such as RPL. RPL is the routing protocol specified for the Internet Of Things. The resulting non-optimized routing topology is then

incrementally optimized by the PPEI PCE that collects a variety of inputs such as the routing topology, events of various natures in addition to the expected SLA, in order to dynamically adapt the routing topology so as to meet the performance requirements. The PPEI PCE's input is thus no longer an explicit path computation request and the output is unchanged.

A second algorithm is specified in order to perform traffic engineering in these networks that computes the traffic matrix thanks to Deep Packet Inspection (DPI) and states reports, routing topology, and gather other network states elements such as dynamic link bandwidths among other attributes before computing a list of overloaded links that require traffic engineering. That list is then provided to the routing engine via an Application Programming Interface (API) within the PPEI-PCE for further action, thus optimizing the routing decision and performing traffic engineering.

8 Conclusion and Future work

8.1 Conclusion

The use of MPLS Traffic Engineering in new generation data networking carrying a variety of traffic types including data, voice, video and other real-time industrial data undoubtedly played a key role in optimizing network resources, satisfying a wide range of SLAs with fine granularity and increasing network availability thanks to local protection mechanisms.

MPLS Traffic Engineering relies on the forwarding of traffic along Label Switched Paths (TE LSPs) that are computed in order to optimize the network resources and satisfy tight SLAs for the traffic carried in the network.

The aim of this thesis was to specify a novel architecture allowing routers (called Path Computation Client – PCC) to off-load the computation of TE LSPs on Path computation Element (PCE) in a fully dynamic manner, thus proposing a radically new path computation model that solves a number of technical issues that were so far unsolved (and discussed in detail in Chapter 1).

After a short overview of the MPLS Traffic Engineering building blocks, we have described the PCE architecture that consists in the specifications of new networking functional entities (PCC and PCE) and the design of a new sophisticated signaling protocol called PCEP. PCEP allows PCCs to signal path computation requests of simple or complex forms specifying (un)correlated, (un)synchronized path requests. Path computation requests can be load balanced among a set of dynamically discovered PCEs according to various algorithms.

After having specified the PCE architecture, we have elaborated a new PCE-based algorithm called the Backward Recursive Path Computation (BRPC) algorithm, which involves a set of PCEs that collaboratively compute the shortest constrained path of a TE LSP across multiple routing domains (IGP areas and Autonomous Systems). It is shown that in contrast with existing per-domain path computation techniques, BRPC guarantees to compute the optimal path while minimizing the control plane overhead and path set up times. BRPC is then augmented with parallel path segment computation thus further improving the path computation time, and other mechanisms making use of path key to preserve confidentiality across multiple Autonomous systems. The performance of BRPC is then compared with the existing per-domain approach and simulations clearly show the BRPC algorithm outperforms non-PCE MPLS path computation techniques in many ways.

In Chapter 5, we have specified a novel PCE-based path computation model, a new set of algorithms and protocols in order to efficiently compute the set of backup tunnels used to fast reroute traffic along backup paths upon detecting a network failure in the network. We have shown that such a novel approach whereby each router acts as a PCE to compute the backup tunnels to protect the traffic transiting

through the router against its own failure allows not only for offering bandwidth protection (non degradation of QoS along backup path) but also optimizing the set of backup capacity in the network, a major objective to reduce cost and optimize network resources in data networks.

In chapters 6 and 7, we have extended the PCE architecture with a new type of PCE that performs traffic inspection, gathers data from various networking entities with regards to network performance and SLA to assist highly constrained devices in routing and traffic engineering in networks referred to as Low power and Lossy Networks (LLNs). Such networks that are at the heart of the Internet of Things may comprise hundreds of thousands of nodes, if not more. This concludes this thesis by showing that the PCE architecture not only solves a wide range of critical problems in IP/MPLS networks but can also be used in a variety of contexts, including the fast growing and extremely promising Internet Of Things that will ineluctably even further increase the role of the network in supporting a wide variety of new services.

8.2 Perspectives

As discussed in the previous section, during the course of this thesis, a number of new technologies (architectures, protocols and algorithms) have been specified that solve a wide range of problems. Still we found out several work items that deserve more research:

Automatic configuration of the backup overlay network

The chapter 5 proposes a very efficient PCE-based distributed model for the computation of backup tunnels protecting independent resources under the single failure assumption that provides a high degree of bandwidth sharing. The placement of these backup tunnels is based on the use of spare capacity also referred to as backup capacity dedicated on each link of the network. This backup capacity constitutes the backup overlay network, a subset of the network topology in terms of connectivity and resources since just a percentage of link bandwidth is allocated to the backup pool.

Off-line tools can be used to determine the appropriate percentage that should be allocated for backup on each link so as to maximize the probability of finding backup tunnels providing bandwidth protection for each protected element in the network, a solvable problem if and only if good estimates of the traffic matrix are available in the network. Unfortunately such information is not always available or the traffic matrix may change too often, leading to recomputing all backup pools, which would trigger the expensive recomputation of all backup tunnels in the network and re-assignment of backup tunnels to primary TE LSPs. Consequently, an interesting area of new research would consist in studying algorithms that would optimize the percentage of bandwidth to be allocated to backup bandwidth pools on each link, in a dynamic fashion in order to 1) maximize the degree of success in finding path for backup tunnels offering bandwidth protection 2) minimize the degree of splitting of

such backup tunnels to avoid bin-packing issues when affecting primary TE LSPs to backup tunnels and 3) minimize incremental changes in the network to avoid the recomputation of new backup tunnels as the traffic matrix changes.

Statefull PCEs

This thesis covers the whole spectrum of the PCE architecture, including the support of stateless and statefull PCE. That being said, a particular attention has been given to the use of stateless PCEs (e.g both the BRPC algorithm and computation of backup tunnels for FRR with bandwidth protection make use of stateless PCEs). This can be explained because states maintenance is a very costly operation in all data networks and the additional gain in terms of network performance efficiency is not always worth the extra cost in terms of complexity and operation.

For the sake of illustration, considering the use of a PCE for the global optimization of primary TE LSP placement within a single routing domain, two path computation models could be of use: 1) Rely on the distributed CSPF path computation by each LSR in the network for its own primary TE LSP augmented with the use of (dynamic) preemption, global rerouting to handle bandwidth fragmentation and the use of soft preemption to avoid traffic disruption for preempted TE LSPs or at the opposite side of the spectrum 2) Rely on a statefull PCE in charge of computing the whole set of primary TE LSPs with the objective of globally optimizing the use of bandwidth and overall network resources. With no doubt, the use of a statefull PCE allows for optimizing the use of network resources as opposed to non-synchronized TE LSP computation. That being said, this leads to a number of issues that deserve more research in terms of network behavior:

1) State maintenance implies additional signaling in the network (for the PCE to be in sync with all nodes and aware of the placement and states of all primary TE LSPs); such signaling overhead may become cumbersome as 1) the number of LSRs increases in the network (the number of primary TE LSPs in a full mesh network grows with the square of the number of primary TE LSPs multiplying by a factor K when load balancing or multi-constrained TE LSP is required (the number of primary TE LSPs is then $K*N*(N-1)$ in a N node-network), and 2) with the level of dynamicity of such TE LSPs in terms of bandwidth resizing.

2) Single point of failures are not acceptable, which means that a **set** of statefull PCEs are required for path computation request load balancing. Furthermore, the PCEs must synchronize their computations to avoid withdrawing bandwidth from the same pool of bandwidth twice, thus requiring shared memory, database, .. across the network while maintaining the need for global optimum of TE LSP computation.

3) The network dynamic behavior becomes extremely complex: in order to satisfy a new request, the statefull may require the displacement of a set of *X* primary TE LSPs. Thus new algorithms must be developed to take into account the traffic perturbation in terms of jitter due to path cost change/increase for the displaced TE LSPs when gracefully rerouted. Furthermore, situations where network connectivity may be interrupted between the set of impacted LSRs (acting as PCC) and the PCE during the operation consisting in displacing the set *X* of primary TE LSPs. Such a situation may become particularly challenging: this would unavoidably lead to put the network in some unknown state requiring complex state recovery.

The aforementioned technical challenges undoubtedly open the door for additional research in these areas, which has already started.

Push mode based PCE in the Internet of Things

The last chapter opens the door to the long-term use of PCE-based architectures in the Internet of Things (IoT). Although existing solutions and protocols are now in place and the IoT is being deployed in a number of verticals such as Energy, Industrial Automation, Connected Vehicles to mention a few with the existing solutions for years to come, new approaches consisting of using PCEs to assist in routing, traffic engineering and QoS in IoT will undoubtedly lead to new research.

List of Publications

Conferences

JeongGil Ko, Joakim Eriksson, Nicolas Tsiftes, Stephen Dawson-Haggerty, Mathilde Durvy, JP Vasseur, Andreas Terzis, Adam Dunkels, and David Culler. *Beyond Interoperability: Pushing the Performance of Sensornet IP Stacks*. In Proceedings of the ACM Conference on Networked Embedded Sensor Systems, ACM SenSys 2011, Seattle, WA, USA, November 2011.

J. Tripathi, J. C. de Oliveira, JP. Vasseur - *A performance evaluation study of RPL: Routing protocol for low power and lossy networks*, 44th Annual Conference on Information Sciences and Systems, CISS 2010; Princeton, NJ; 17 March 2010 through 19 March 2010.

J. Yu, Y. He, K. Wu, M. Tacca, A. Fumagalli, JP. Vasseur - *A Queueing Model Framework of PCE-based Inter-area Path Computation, Infocom 2009,* April 2009.

S. Dasgupta, J. C. de Oliveira, and J.-P.Vasseur, Krakow, Poland, *Trend Based Bandwidth Provisioning: An Online Approach for Traffic Engineered Tunnels*, Proceedings of the Next Generation Internet Networks (Euro-NGI 2008), April 28-30, 2008.

J. C. de Oliveira, and J.-P.Vasseur *A Performance Study of IP and MPLS Traffic Engineering Techniques under Traffic Variations*, S. Dasgupta, Proceedings of the IEEE Globecom 2007, Washington, DC, November 26-30, 2007.

S. Dasgupta, J. C. de Oliveira, and J.-P. Vasseur *A New Distributed Dynamic Bandwidth Reservation Mechanism to Improve Resource Utilization: Simulation and Analysis on Real Network and Traffic Scenarios in the Proceedings of IEEE INFOCOM 2006, Barcelona, Spain,* April 23-29, 2006.

Journals

K. Lee, R. Sudhaakar, J. Ning, L. Dai, S. Addepalli, JP Vasseur, M. Gerla, *A Comprehensive Evaluation of RPL under Mobility,* To be published in International Journal of Vehicular Technology.

S. Dasgupta, J. C. de Oliveira, and Jean-Philippe Vasseur, *Path-Computation-Element-Based Architecture for Interdomain MPLS/GMPLS Traffic Engineering: Overview and Performance*, in IEEE Network, Special Issue on Network Systems Architecture, vol. 21(4), pp. 38-45, July/August 2007.

Standards

T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, JP. Vasseur, R. Alexander, *RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks, RFC6550, March 2012*.

JP. Vasseur, M. Kim, K. Pister, N. Dejean, D. Barthel, *Routing Metrics Used for Path Calculation in Low-Power and Lossy Networks*, RFC 6551, March 2012.

J. Hui, JP. Vasseur, *The Routing Protocol for Low-Power and Lossy Networks (RPL) Option for Carrying RPL Information in Data-Plane Datagrams*, RFC 6553, March 2012.

J. Hui, JP. Vasseur, D. Culler, *An IPv6 Routing Header for Source Routes with the Routing Protocol for Low-Power and Lossy Networks (RPL)*, RFC 6554, March 2012.

JP Vasseur, Navneet Agarwal, Jonathan Hui, Zach Shelby, Paul Bertrand, Cedric Chauvenet RPL: *The IP routing protocol designed for low power and lossy networks*, IPSO alliance, April 2011.

Jean-Philippe Vasseur, JL. Le Roux, Y. Ikejiri, *A Set of Monitoring Tools for Path Computation Element (PCE)-Based Architecture*, RFC5886, June 2010.

Jean-Philippe Vasseur, JL, Roux, Y. Ikejiri, *A set of monitoring tools for Path Computation Element based Architecture*, RFC 5886, June 2010.

M. Meyer, Jean-Philippe Vasseur *MPLS Traffic Engineering Soft Preemption*, RFC 5712, January 2010.

Jean-Philippe Vasseur, JL. Le Roux, *Path Computation Element (PCE) Communication Protocol (PCEP)*, RFC 5440, March 2009.

Jean-Philippe Vasseur, R. Zhang, N. Bitar, JL. Le Roux , *A Backward-Recursive PCE-Based Computation (BRPC) Procedure to Compute Shortest Constrained Inter-Domain Traffic Engineering Label Switched Paths*, RFC 5441, April 2009

R. Bradford, Jean-Philippe Vasseur, A. Farrel, *Preserving Topology Confidentiality in Inter-Domain Path Computation Using a Path-Key-Based Mechanism*, RFC5520, April 2009.

JP. Vasseur, A. Ayyangar, R. Zhang, *A Per-Domain Path Computation Method for Establishing Inter-Domain Traffic Engineering (TE) Label Switched Paths (LSPs)*, RFC 5152, February 2008.

JL. Le Roux, JP. Vasseur, Y. Ikejiri, R. Zhang *OSPF Protocol Extensions for Path Computation Element (PCE) Discovery*, RFC 5088, January 2008.

JL. Le Roux, JP. Vasseur, Y. Ikejiri, R. Zhang *IS-IS Protocol Extensions for Path Computation Element (PCE) Discovery*, RFC 5089, January 2008.

J. de Oliveira, JP. Vasseur, L. Chen, C. Scoglio *Label Switched Path (LSP) Preemption Policies for MPLS Traffic Engineering,* RFC 4829, Avril 2007.

Jean-Philippe Vasseur, S. Previdi, Definition of an IS-IS Link Attribute Sub-TLV , RFC 5029, September 2007.

A. Farrel, J.-P. Vasseur, J. Ash, *A Path Computation Element (PCE)-Based Architecture*, RFC 4655 , August 2006.

P. Pan, G. Swallow, A. Atlas, Jean-Philippe Vasseur, Markus Jork, Der-Hwa Gan, Dave Cooper, *Fast Reroute Extensions to RSVP-TE for LSP Tunnels*, RFC 4090, May 2005.

Jean-Philippe Vasseur, R. Zhang, X. Vinet, S. Matsushima, A. Atlas *RSVP Path computation request and reply messages* - http://tools.ietf.org/html/draft-vasseur-mpls-computation-rsvp, June 2002.

Books

JP Vasseur, Adam Dunkels, *Interconnecting Smart Objects with IP: The next Internet* - Morgan Kaufman -, 400 pages, May 2010.

JP Vasseur, Jim Guichard et Francois Le Faucheur, *Definitive MPLS Network Designs* - Cisco Press -, 552 Pages, March 2005.

JP Vasseur, Mario Pickavet, and Piet Demeester *Network Recovery: Protection and Restoration of Optical, SONET-SDH, IP, and MPLS* (The Morgan Kaufmann Series in Networking) - Morgan Kaufmann, , 544 pages, July 2004.

Glossary

A

ABR: Area border Router

API: Application Programming Interface

ASBR: Autonomous System Border Routers

ATM: Asynchronous Transport Mode

B

BFD: Bidirectional Forwarding Detection

BNF: Backus-Naur Form

BRPC: Backward Recursive Path Computation

C

CAC: Call Admission Control

CSPF: Constrained Shortest Path First

CIC: Central Intelligence Controller

CoAP: Constrained Application Protocol

CORE: Constrained RESTful Environements

CSPF: Constrained Shortest Path First

D

DAO: DODAG Destination Advertisement Object

DIO: DODAG Information Object

DIA: Distributed Intelligence Agent

DIA-R: Distributed Intelligence Agent – Routing

DIA-TE: Distributed Intelligence Agent – Traffic Engineering

DIS: DODAG Information Solicitation

DODAG: Destination Oriented Directed Acyclic Graph

DPI: Deep Packet inspection

DSCP: DiffServ Code Point

DS-TE: DiffServ aware Traffic Engineering

E

ECN: Explicit Congestion Notification

ERO: Explicit Route Object

ETX: Expected Transmission

F

FSM: Finite State Machine

FBCM: Facility Based Computation Model

FRR: Fast Reroute

G

GRE: Generic Routing Encapsulation

I

ICMP: Internet Control Message Protocol

ID: IDentification

IP: Internet Protocol

IPHC: Internet Protocol Header Compression

IoT: Internet of Things

IRO: Include Route Object

IS-IS: Intermediate System to Intermediate System

L

LBR: Low power and lossy network Border Router

LLN: Low power and Lossy Network

M

MCO: Minimalist Connected Object

MBB: Make Before Break

MPLS: Multi Protocol Label Switching

MPLS VPN: MPLS Virtual Private Network

MPLS TE: MPLS Traffic Engineering

N

NCPC: Naïve CSPF-based Path Computation approach

NMS: Network Management System

MTR: Multi Topology Routing

O

OF: Objective Function

OSPF: Open Shortest Path First

P

PCFR: Path Computation Failure Rate

PCC: Path Computation Client

PDR: Packet Delivery Rate

PCC: Path Computation Client

PCE: Path Computation Element

PCEP: Path Computation Element signaling Protocol

PCFR: Path Computation Failure Rate

PCR: PCE Computation Resource

PCReq: Path Computation Request message

PCRep: Path Computation Reply message

PCNtf: Path Computation Notification message

PCErr: Path Computation Error message

PPEI PCE: Push-based Packet/Event Inspecting PCE

PLR: Point of Local Repair

PPNI: Private Network-to-Network Interface

PPEI: Push based Packet Event Inspecting

PRT: Predictive Response Time

Q

QoS: Quality of Service

L

LBR: Low power and lossy network Border Router

LSA: Link State Advertisement

LSP: Label Switch Path

LER: Label Edge Router

LFIB: Label Forwarding Information Base

LLN: Low power and Lossy Networks

LSDB: Link State DataBase

LSR: Label Switch Router

M

MCU: Micro Controller Unit

MIB: Management Information Base

MP: Merge Point

MinRT: Minimal Routing Topology

MRT: Maximum Response Time

P

PPEI: Push based Packet Event Inspecting

R

RED: Random Early Discard

RDM: Russian Doll Model

ROLL: Routing Over Low power and Lossy networks

RP: Routing Protocol

RPL: Routing in Low power and Lossy networks

RT: Routing Topology

RSVP-TE: Resource reservation Protocol – Traffic Engineering

RSSI: Received Signal Strength Indicator

RRO: Record Route Object

S

SN: Sensor Networks

SDLG: Shared SRLG Dependency Link Group

SLA: Service Level Agreement

SNMP: Simple Network Management Protocol

SPF: Shortest Path First

SRLG: Shared Risk Link Group

T

TDM: Time Division Multiplexing

TED: Traffic Engineering Database

TE LSP: Traffic Engineering Label Switch Path)

TCP: Transmission Control Protocol

TLV: Type-Length-Value

TE: Traffic Engineering

V

VSPT: Virtual Shortest Path Tree

Bibliography

[1] J. Moy *OSPF Version 2*, Internet Engineering Task Force RFC2328, April 1998.

[2] *ISO Intermediate System to Intermediate system routing information exchange protocol in use in conjunction with the protocol for providing the Connectionless-mode Network service (ISO 8473)"* ISO/IEC10589:1992.

[3] Dijkstra, E. W. *"A note on two problems in connexion with graphs"*. *Numerische Mathematik* **1**: 269–271. doi:10.1007/BF01386390., 1959.

[4] Zinky, J., Vichniac, G., and A. Khanna, *Performance of the Revised Routing Metric for ARPANET and MILNET*, Military Communications Conference, MILCOM '89, March 1989.

[5] Douglas R. Mauro & Kevin J. Schmidt. *Essential SNMP* (1st ed.). Sebastopol, CA: O'Reilly & Associates, 2001.

[6] Eric Osborne, Ajay Simha – *"Traffic Engineering with MPLS"* Cisco Press, 2002

[7] Jean-Philippe Vasseur, Jim Guichard and Francois Le Faucheur - *"Definitive MPLS Network Designs"* Cisco Press - March 2005.

[8] Emilio C.G. Wille Emilio Leonardi Marco Ajmone Marsan, *Algorithms for IP network design with end-to-end QoS constraints*, Elsevier, Computer Networks, Volume 50, Issue 8, 6 June 2006, Pages 1086–1103.

[9] Marco Ajmone Marsan, Claudio Casetti, Gianluca Mardente, Marco Mellia, *A Framework for Admission Control and Path Allocation in DiffServ Networks*, Computer Networks, Vol.5, No.10, pp.2738-2752, 11 July 2007.

[10] A. Cuadra, Felipe Mata, José Luis García-Dorado, Javier Aracil, Jorge E. López de Vergara, Francisco Cortés, Pablo Beltrán Pellicer, E. de Mingo, A. Ferreiro, *Traffic monitoring for assuring quality of advanced services in future internet*, Proceedings of the 9th IFIP TC 6 international conference on Wired/wireless internet communications (WWIC'2011), Vilanova i la Geltrú, Spain, June 15-17, 2011. Published in Lecture Notes in Computer Science, Vol. 6649, pp. 186-196, Springer Verlag.

[11] D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, G. Swallow, *RSVP-TE: Extensions to RSVP for LSP Tunnels*, Internet Engineering Task Force RFC3209, 2001.

[12] *"A New Distributed Dynamic Bandwidth Reservation Mechanism to Improve Resource Utilization: Simulation and Analysis on Real Network and Traffic Scenarios",* S. Dasgupta, J. C. de Oliveira, and J.-P. Vasseur *in the Proceedings of IEEE INFOCOM 2006, Barcelona, Spain, April 23-29, 2006*

[13] *"Trend Based Bandwidth Provisioning: An Online Approach for Traffic Engineered Tunnels", Proceedings of the Next Generation Internet Networks (Euro-NGI 2008),* S. Dasgupta, J. C. de Oliveira, and J.-P.Vasseur*,* Krakow, Poland, April 28-30, 2008

[14] *"A Performance Study of IP and MPLS Traffic Engineering Techniques under Traffic Variations",* S. Dasgupta, J. C. de Oliveira, and J.-P.Vasseur *,Proceedings of the IEEE Globecom 2007, Washington, DC, November 26-30, 2007.*

[15] J. de Oliveira, JP. Vasseur, L. Chen, C. Scoglio *Label Switched Path (LSP) Preemption Policies for MPLS Traffic Engineering,* RFC 4829, Avril 2007

[16] T. Li, H. Smit, *IS-IS Extensions for Traffic Engineering*, RFC5305, October 2008.

[17] D. Katz, K. Kompella, D. Yeung, *Traffic Engineering (TE) Extensions to OSPF Version 2*, RFC3630, September 2003.

[18] B. Jamoussi, L. Andersson, R. Callon, R. Dantu, L. Wu, P. Doolan, T. Worster, N. Feldman, A. Fredette, M. Girish, E. Gray, J. Heinanen, T. Kilty, A. Malis, *Constraint-Based LSP Setup using LDP*, RFC3212, January 2002.

[19] *Tunnels* D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, G. Swallow: *RSVP-TE: Extensions to RSVP for LSP*, RFC 3209, December 2001

[20] R. Braden, L. Zhang, S. Berson, S. Herzog, S. Jamin, *Resource ReSerVation Protocol (RSVP) - Version 1 Functional Specification*, RFC 2205, September 1997.

[21] L. Berger, D. Gan, G. Swallow, P. Pan, F. Tommasi, S. Molendini, *RSVP Refresh Overhead Reduction Extensions*, RFC 2961, April 2001.

[22] L. Berger, *Generalized Multi-Protocol Label Switching (GMPLS) Signaling Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Extensions*, RFC 3473, January 2003.

[23] Jean-Philippe Vasseur, Mario Pickavet, and Piet Demeester "*Network Recovery: Protection and Restoration of Optical, SONET-SDH, IP, and MPLS (The Morgan Kaufmann Series in Networking)"* Morgan Kaufmann - July 2004, 544 pages

[24] P. Pan, G. Swallow, A. Atlas, Jean-Philippe Vasseur, Markus Jork, Der-Hwa Gan, Dave Cooper, *Fast Reroute Extensions to RSVP-TE for LSP Tunnels*, RFC 4090, May 2005.

[25] D. Katz, D. Ward, *Bidirectional Forwarding Detection (BFD)*, RFC5880, June 2010.

[26] JP. Vasseur, A. Ayyangar, R. Zhang, *A Per-Domain Path Computation Method for*

Establishing Inter-Domain Traffic Engineering (TE) Label Switched Paths (LSPs), RFC 5152, February 2008.

[27] ATM Forum 94-0471R13 PNNI Draft Specification, PNNI SWG, ftp://ftp.fit.vutbr.cz/pub/doc/ATM/atmforum/contributions/atm94-0471.ps

[28] A. Farrel et al., *Crankback signaling extensions for MPLS and GMPLS RSVP-TE* Internet-Draft, Work in progress, Version 05, IETF, November 2005.

[29] A. Farrel, J.-P. Vasseur, J. Ash, *A Path Computation Element (PCE)-Based Architecture,* RFC 4655 , August 2006.

[30] V. Lopez, B. Huiszoon, J. Fernandez-Palacios, O. Gonzalez de Dios, J. Aracil, *Path computation element in telecom networks: Recent developments and standardization activities*,
14th Conference on IEEE Optical Network Design and Modeling (ONDM) conference, 2010

[31] H. Gredler, A. Farrel, J. Medved, S. Previdi, *North-Bound Distribution of Link-State and TE Information using BGP,* draft-gredler-idr-ls-distribution, Work in progress, March 2012.

[32] JL. Le Roux, JP. Vasseur, Y. Ikejiri, R. Zhang *OSPF Protocol Extensions for Path Computation Element (PCE) Discovery*, , RFC 5088, January 2008

[33] JL. Le Roux, JP. Vasseur, Y. Ikejiri, R. Zhang *IS-IS Protocol Extensions for Path Computation Element (PCE) Discovery,* RFC 5089, January 2008

[34] J. Sermersheim, *Lightweight Directory Access Protocol (LDAP): The Protocol*, RFC4511, June 2006.

[35] Y. Rekhter, T. Li, S. Hares, *A Border Gateway Protocol 4 (BGP-4)*, RFC4271, June 2006.

[36] Jean-Philippe Vasseur, R. Zhang, X. Vinet, S. Matsushima, A. Atlas *RSVP Path computation request and reply messages* - http://tools.ietf.org/html/draft-vasseur-mpls-computation-rsvp, June 2002.

[37] Chairs: Jean-Philippe Vasseur, Julien Meuric, "*Path Computation Element (pce)*" Working Group, IETF, http://datatracker.ietf.org/wg/pce/charter/.

[38] Jean-Philippe Vasseur, JL. Le Roux , *Path Computation Element (PCE) Communication Protocol (PCEP),* RFC 5440*,* March 2009

[39] J. Ash, J. Le Roux, *Path Computation Element (PCE) Communication Protocol Generic Requirements*, RFC 4657, September 2006.

[40] JL. Le Roux, *Path Computation Element Communication Protocol (PCECP) Specific Requirements for Inter-Area MPLS and GMPLS Traffic Engineering*, RFC 4927, June 2007.

[41] N. Bitar, R. Zhang, K. Kumaki, *Inter-AS Requirements for the Path Computation Element Communication Protocol (PCECP),* RFC 5376, November 2008.

[42] E. Oki, T. Takeda, JL Le Roux, A. Farrel, Fatai Zhang , *Extensions to the Path Computation Element communication Protocol (PCEP) for Inter-Layer MPLS and GMPLS Traffic Engineering*, draft-ietf-pce-inter-layer-ext , Work in progress.

[43] Jean-Philippe Vasseur, JL. Le Roux, Y. Ikejiri, *A Set of Monitoring Tools for Path Computation Element (PCE)-Based Architecture*, RFC5886, June 2010

[44] A Farrel, *Reduced Backus-Naur Form (RBNF) A Syntax Used in Various Protocol Specifications*", draft-farrel-rtg-common-bnf, Work in Progress, September 2009.

[45] Jean-Philippe Vasseur, S. Previdi, *Definition of an IS-IS Link Attribute Sub-TLV* , RFC 5029, September 2007

[46] Jean-Philippe Vasseur, JL, Roux, Y. Ikejiri, *A set of monitoring tools for Path Computation Element based Architecture*, RFC 5886, June 2010.

[47] Jean-Philippe Vasseur, R. Zhang, N. Bitar, JL. Le Roux **,** *A Backward-Recursive PCE-Based Computation (BRPC) Procedure to Compute Shortest Constrained Inter-Domain Traffic Engineering Label Switched Paths,* RFC 5441, April 2009

[48] R. Bradford, Jean-Philippe Vasseur, A. Farrel,*Preserving Topology Confidentiality in Inter-Domain Path Computation Using a Path-Key-Based Mechanism*, RFC5520, April 2009

[49] S. Dasgupta, **J. C. de Oliveira**, and Jean-Philippe Vasseur, *"Path-Computation-Element- Based Architecture for Interdomain MPLS/GMPLS Traffic Engineering: Overview and Performance", , in IEEE Network, Special Issue on Network Systems Architecture, vol. 21(4), pp. 38-45, July/August 2007.*

[50] Sebastian Gunreben, Franz Rambach, *Assessment and Performance Evaluation of PCE-based Inter-Layer Traffic Engineering*

[51] Alvarez, D.; Lopez, V.; Anamuro, J.L.; de Vergara, J.L.; de Dios, O.G.; Aracil, J., *Utilization of temporary reservation of path computed resources for multi-domain path computation element protocols in WDM networks*, IEEE International Conference on the Network of the Future (NOF), pp. 102-106,2011.

[52] M. Meyer, Jean-Philippe Vasseur *MPLS Traffic Engineering Soft Preemption*, RFC 5712, January 2010.

[53] Jean-Philippe Vasseur, Anna Charny, Francois Le Faucheur , Javier Achirica, *MPLS Traffic Engineering Fast reroute: backup tunnel path computation for bandwidth protection,* draft-vasseur-mpls-backup-computation-00.txt, June, 2002.

[54] "Northeast blackout of 2003", http://en.wikipedia.org/wiki/Northeast_blackout_of_2003

[55] M. Gagnaire, J. Kuri and E. Doumith,. *Grooming of Scheduled demands in Multi-Layer Optical Networks. Traffic Grooming for Optical Networks: Foundations, Techniques and Frontiers*, Collective book edited by Rudra Dutta, Ahmed E. Kamal and George N. Rouskas, Springer, pp. 253-278, 2008.

[56] E. A. Doumith, M. Gagnaire, O. Audouin, and N. Puech. *Traffic Engineering for Virtual Private Networks and Random Traffic Demands in WDM Optical Core Networks*. In Proc. of the IEEE ICCCN Conference, San Diego, California, USA, October 2005.

[57] E. A. Doumith, M. Gagnaire, O. Audouin, and R. Douville. *From Network Planning to Traffic Engineering for Optical VPN and Multi-Granular Random Demands*. In Proc. of the IPCCC Conference, Phoenix, Arizona, USA, April 2006.

[58] JL. Leroux, G. Calvignac, *A method for an Optimized Online Placement of MPLS Bypass Tunnels*, draft-leroux-mpls-bypass-placement-00.txt, February 2002.

[59] Kini et al, *Shared Backup Label Switched Path Restoration'*, draft-kini-restoration-shared-backup-01.txt, May 2001.

[60] The IETF Integrated Service Working Group - http://datatracker.ietf.org/wg/intserv/charter/

[61] Sally Floyd and Van Jacobson , *Random Early Detection Gateways for Congestion Avoidance*, IEEE/ACM Transactions On Networking, Vol1 No4, August 1993.

[62] J. Yu, Y. He, K. Wu, M. Tacca, A. Fumagalli, JP. Vasseur - *A Queueing Model Framework of PCE-based Inter-area Path Computation, Infocom 2009, Apr 2009*

[63] JP Vasseur, A. Dunkels *Interconnecting Smart Objects with IP: The next Internet* - - Morgan Kaufman - May 2010, 400 pages.

[64] IEEE IEEE 802.15 WPAN™ Task Group 4 (TG4), http://www.ieee802.org/15/pub/TG4.html

[65] Z. Shelby, K. Hartke, C. Bormann, B. Frank, *Constrained Application Protocol (CoAP)*, draft-ietf-core-coap-09, work in progress, March 2012.

[66] The IETF Constrained RESTful Environments (core) Working Group, http://datatracker.ietf.org/wg/core/charter/

[67] The IETF Routing Over Low power and Lossy networks (roll) Working Group, http://datatracker.ietf.org/wg/roll/charter/

[68] J. Kempf, R. Austein, *The Rise of the Middle and the Future of End-to-End: Reflections on the Evolution of the Internet Architecture*, RFC 3247, March 2004.

[69] Deep Packet Inspection - http://searchnetworking.techtarget.com/definition/deep-packet-inspection-DPI

[70] S. Kumar, J. Turner, J. Williams, *Advanced Algorithms for Fast and Scalable Deep Packet Inspection* in Proceedings of IEEE/ACM Symposium on Architectures for Networking and Communications Systems (ANCS), San Jose, California, December, 2006.

[71] J. Hui, P. Thubert *Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks*, September 2011.

[72] L. Andersson, I. Minei, B. Thomas, *LDP Specification*, RFC5036, October 2007.

[73] M. Dohler, T. Watteyne, T. Winter, D. Barthel, *Routing Requirements for Urban Low-Power and Lossy Networks*, RFC5548, May 2009.

[74] K. Pister, P. Thubert, S. Dwars, T. Phinney, *Industrial Routing Requirements in Low-Power and Lossy Networks*, RFC 5673, October 2009.

[75] A. Brandt, J. Buron, G. Porcu, *Home Automation Routing Requirements in Low-Power and Lossy Networks*, RFC 5826, April 2010.

[76] J. Martocci, P. De Mil, N. Riou, W. Vermeylen, *Building Automation Routing Requirements in Low-Power and Lossy Networks*, RFC 5867, June 2010.

[77] T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, JP. Vasseur, R. Alexander, *RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks*, RFC6550, March 2012

[78] JP. Vasseur, M. Kim, K. Pister, N. Dejean, D. Barthel, *Routing Metrics Used for Path Calculation in Low-Power and Lossy Networks*, RFC 6551, March 2012.

[79] P. Thubert, *Objective Function Zero for the Routing Protocol for Low-Power and Lossy Networks (RPL)*, RFC6552, March 2012.

[80] J. Hui, JP. Vasseur, *The Routing Protocol for Low-Power and Lossy Networks*

(RPL) Option for Carrying RPL Information in Data-Plane Datagrams, RFC 6553, March 2012.

[81] J. Hui, JP. Vasseur, D. Culler, *An IPv6 Routing Header for Source Routes with the Routing Protocol for Low-Power and Lossy Networks (RPL)*, RFC 6554, March 2012.

[82] O. Gnawali, P. Levis, *The Minimum Rank with Hysteresis Objective Function,* draft-ietf-roll-minrank-hysteresis-of-10, work in progress.

[83] JP Vasseur, Navneet Agarwal, Jonathan Hui, Zach Shelby, Paul Bertrand, Cedric Chauvenet *RPL: The IP routing protocol designed for low power and lossy networks*, IPSO alliance, April 2011

[84] J. Tripathi, J. C. de Oliveira, JP. Vasseur - *"A performance evaluation study of RPL: Routing protocol for low power and lossy networks" –,* 44th Annual Conference on Information Sciences and Systems, CISS 2010; Princeton, NJ; 17 March 2010 through 19 March 2010

[85] K. Lee, R. Sudhaakar, J. Ning, L. Dai, S. Addepalli, JP Vasseur, M. Gerla - *"A Comprehensive Evaluation of RPL under Mobility"*, To be published in International Journal of Vehicular Technology.

[86] A. Conta, S. Deering, M. Gupta, *Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification*, RFC 4443, March 2006.

[87] P. Levis, T. Clausen, J. Hui, O. Gnawali, J. Ko, *The Trickle Algorithm*, RFC6206, March 2011.

[88] P. Levis, N. Patel, D. Culler, S. Shenker, *Trickle: A Self-Regulating Algorithm for Code Propagation and Maintenance in Wireless Sensor Networks*. In *Proceedings of the First USENIX/ACM Symposium on Networked Systems Design and Implementation (NSDI 2004)*.

[89] P. Levis, N. Patel, S. Shenker, D. Culler, *Trickle: A Self-Regulating Algorithm for Code Propagation and Maintenance in Wireless Sensor Networks*. UC Berkeley Tech Report UCB//CSD-03-1290, November 2003.

[90] A. Atlas, A. Zinin, *Basic Specification for IP Fast Reroute: Loop-Free Alternates*, RFC 5286, September 2008.

[91] P. Psenak, S. Mirtorabi, A. Roy, L. Nguyen, P. Pillay-Esnault, *Multi-Topology (MT) Routing in OSPF*, RFC 4915 , June 2007.

[92] T. Przygienda, N. Shen, N. Sheth, *M-ISIS: Multi Topology (MT) Routing in Intermediate System to Intermediate Systems (IS-ISs)*, RFC 5120, February 2008.

[93] K. Nichols, S. Blake, F. Baker, *Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*, RFC 2474 , December 1998.

[94] J. Case, M. Fedor, M. Schoffstall J. Davin, *The Simple Network Management Protocol*, STD 15, RFC 1157, May 1990.

[95] J. Case, D. Harrington, R. Presuhn, B. Wijnen, *Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)*, RFC 2572, April 1999.

[96] D. Harrington, R. Presuhn, B. Wijnen, *An Architecture for Describing SNMP Management Frameworks*, RFC2571, April 1999.

[97] K. Ramakrishnan, S. Floyd, *A Proposal to add Explicit Congestion Notification (ECN) to IP*, RFC 2481, January 1999.

[98] K. Ramakrishnan, S. Floyd, D. Black, *The Addition of Explicit Congestion Notification (ECN) to IP*, RFC 3168, September 2001.

[99] Heffes H., Lucantoni D., *A Markov Modulated characterization of packetized voice and data traffic and related statistical multiplexer performance*, IEEE Journal on Selected Areas in Communications, Vol. 4, N. 6, pp. 856-868, September 1986.

[100] Chydzinski A., *Time to buffer overflow in an MMPP queue*, Proceedings of IEEE Networking 2007, pp. 879-889, 2007.

[101] Jean-Marie A., Liu Z., Nain P. Towsley D., *Computational aspects of the workload distribution in the MMPP/GI/1 queue*, March 6, 1998.

[102] Sykas E. D., Vlakos K. M., Protonotarios E. N., *Mathematical tools for analysis of ATM systems*, Teletraffic and Data Traffic in a period of change, ITC-12, Elsevier publishers, pp. 781-786, 1991.

[103] Garg S., Huang Y., Kintala C., Trivedi K. S., *Time and Load based software rejuvenation: policy, evaluation and optimality*, Fault Tolerant Systems, IIT Madras, December 20-22, 1995.

[104] Figueiredo D. R., Liu B., Feldmann A., Misra V. Towsley D. Willinger W., *On TCP and self-similar trafic*, Elsevier Science, 2004.

[105] Willinger W., Taqqu M. S., Erramilli A., *A Bibliographical Guide to Self-Similar Traffic and Performance Modeling for Modern High-Speed Networks*, Stochastic Networks: Theory and Applications, pages 339-366. Oxford University Press, 1996.

Lists of Patents

United State Patent – Vasseur et al. Patent Number US 7,230,913B1 – June 12, 2007 – “MPLS Fast Reroute without Full mesh Traffic Engineering”

United State Patent – Vasseur et al. Patent Number US 7,599,349 B2 – October 2009 – “Computing inter Autonomous systems MPLS Traffic Engineering LSP Paths”

United State Patent – Sivabalan et al. Patent Number US 7,801,048 B1 – September 2010 - “concurrent path computation using virtual shortest path tree”

United State Patent – Vasseur et al. Patent Number US 7,031,262 B2 – April 2006- “Reoptimization Triggering by Path Computation Elements”

United State Patent – Vasseur et al. Patent Number US 7,558,276 B2 – July 2009 – “System and Method for Retrieving Computed Paths from a Path Computation Element Using a Path Key”

United State Patent – Vasseur et al. Patent Number US 7,623,461 B2 – November 2009 – “Trigger for Packing Path Computation Request”

United State Patent – Vasseur et al. Patent Number US 7,515,529 B2 – April 2009 – “Efficient Mechanism for Fast Recovery In Case of Border Router Node Failure in a Computer Network”

United State Patent – Vasseur et al. Patent Number US 7,684,351 B2 – Mach 2010 – “Inter-domain optimization trigger in PCE-based environments”

United State Patent – Vasseur Patent Number US 7,710,872 B2 – May 2010 – “Technique for Enabling Traffic Engineering on CE-CE path across a Provider Network”

United State Patent – Vasseur Patent Number US 7,522,603 B2 – April 2009 – “Technique for Efficiency Routing IP Traffic on CE-CE paths across a Provider Network”

United State Patent – Vasseur et al. Patent Number US 7,668,971 B2 – February 2010 – “Dynamic Path Computation Element Load Balancing with Backup Path Computation Elements”

United State Patent – Vasseur et al. Patent Number US 7,995,593 B2 – August 2011 – “System and Method for Retrieving Computed Path from a Path Computation Element Using Encrypted Objects”