Towards flexible QoS provisioning for inter-provider services

Gilles Bertrand

N° d'ordre : 2009telb0135

Thèse

Présentée à

l'École Nationale Supérieure des Télécommunications de Bretagne

en habilitation conjointe avec l'Université de Rennes 1

pour obtenir le grade de

Docteur de Télécom Bretagne

Mention : *Informatique*

par

Gilles Bertrand

Mécanismes de routage inter-domaine multi-critère. Vers des services inter-opérateurs à performances garanties.

Soutenue le 8 Décembre 2009, devant la Commission d'Examen :

Composition du Jury :

Rapporteurs :	Prof. Dominique Barth, Lab. PriSM, Université de Versailles
	Prof. Catherine Rosenberg, University of Waterloo
Examinateurs :	Mohamed Boucadair, Orange Labs
	Prof. Xavier Lagrange, Télécom Bretagne (Directeur de thèse)
	Dr. Miklós Molnár, IRISA–INSA de Rennes
	Dr. Géraldine Texier, Télécom Bretagne (Encadrante de thèse)

Cho người tôi yêu.
Em luôn bên tôi, cùng tôi qua những ngày bão giông để đón chờ một ngày nắng mới.

Contents

List of Figures

List of Tables

CHAPITRE 1

Résumé

1.1 Introduction

1.1.1 Problématiques étudiées

La qualité de service (quality of service, QoS) revêt une importance critique dans les accords contractuels entre les fournisseurs de services et leurs clients. Par conséquent, les opérateurs utilisent divers mécanismes pour garantir que le réseau fournit les niveaux de service spécifiés dans leurs contrats. Dans la thèse, nous introduisons deux problèmes cruciaux liés aux technologies de QoS existantes : (1) la configuration d'un réseau pour fournir un niveau spécifique de service et (2) la fourniture de QoS de bout en bout pour les flux qui traversent plusieurs réseaux.

Les mécanismes de QoS disponibles actuellement permettent aux opérateurs de gérer la QoS dans leur réseau, et donc, de créer des « ilots de QoS ». Néanmoins, il est difficile de garantir que les performances du réseau sont conformes à une spécification de niveau de service (service level specification, SLS) particulière. Par exemple, la difficulté à mettre en œuvre des comportements saut-par-saut (per hop behavior, PHB) a conduit l'organisme de normalisation de l'Internet, l'IETF, à publier des directives de configuration pour l'architecture DiffServ [10] de gestion de la QoS en 2006, huit ans après la publication des normes DiffServ initiales.

La QoS doit être considérée comme un problème de bout en bout, car les utilisateurs observent les performances de bout en bout de leurs services. Toutefois, le contrôle du trafic est partagé entre les réseaux traversés, qui gèrent leur trafic localement et sans perspective globale. C'est pourquoi la gestion de bout-en-bout du trafic inter-domaine avec QoS est intrinsèquement difficile et reste un problème ouvert, même si des technologies sont disponibles pour la gestion de QoS intra-domaine.

Récemment, plusieurs étapes importantes vers la fourniture de qualité de service de bout en bout pour les flux inter-domaines ont été réalisées au sein de l'IETF. Ce dernier a notamment publié une série d'exigences pour l'ingénierie de trafic inter-domaine avec MPLS [189] en 2005. Les travaux de l'IETF pour définir une architecture qui répond à cette série d'exigences ont abouti à la publication de plusieurs RFCs qui décrivent des éléments de calcul de chemin (path computation element, PCE) [66]. En parallèle, l'IETF a étendu les protocoles de réservation de ressources pour la gestion des flux inter-domaines [64]. Ces protocoles représentent un élément important des architectures d'ingénierie de trafic : leur extension est donc un progrès majeur vers la fourniture de QoS inter-domaine et vers une ingénierie fine du trafic inter-domaine.

Plus précisément, elle a permis au groupe de travail PCE de l'IETF de fournir aux opérateurs un cadre pour la gestion du trafic inter-domaine [175, 176].

De nombreux problèmes techniques doivent être résolus pour permettre la fourniture de qualité de service de bout en bout pour les flux inter-opérateurs. En particulier, la négociation d'accords d'interconnexion nécessite généralement beaucoup de temps et d'efforts de la part des opérateurs. En outre, le niveau de performance des chemins inter-domaines dépend de la configuration de tous les domaines traversés. Enfin, la détermination d'un chemin performant et la configuration d'un comportement saut-par-saut approprié au sein de tous les domaines traversés sont des tâches complexes.

Le problème de la fourniture de QoS devient de plus en plus important pour les opérateurs. En effet, un grand nombre de fournisseurs de services envisagent de migrer certaines applications, tels que la téléphonie, d'une architecture supportant la QoS vers une architecture Internet, traditionnellement best-effort. Les architectures considérées pour cette migration ont principalement été étudiées du point de vue des couches hautes, telles que la couche de fourniture de service. En revanche, nous pensons que les mécanismes réseaux utilisés pour mettre en oeuvre les services et notamment pour garantir la qualité de service au niveau du réseau demandent des études supplémentaires.

Dans ce contexte, nous pouvons résumer nos contributions de la façon suivante : nous proposons des extensions des protocoles et des architecture existants, ainsi que de nouveaux algorithmes de calcul de chemins pour mieux répondre aux exigences des problèmes de calcul de chemins inter-domaines, par rapport à l'existant. Nos solutions représentent un élément fondamental de futures architectures de réseau supportant la QoS.

1.1.2 Plan de la thèse

Le reste de la thèse se divise en deux parties. La partie I présente le contexte technologique de nos études et les éléments nécessaires pour comprendre le travail effectué durant cette thèse.

Nous commençons la partie I en expliquant l'évolution récente d'Internet vers une architecture unifiée supportant de multiples services. Nous introduisons l'architecture la plus connue pour les réseaux convergents, dans le chapitre 3. La présentation de cette architecture et de ses limites est importante pour expliquer les problèmes de gestion de trafic auxquels les opérateurs réseau sont actuellement confrontés et qui sont à la racine de nos travaux.

Dans le chapitre 4, nous présentons un bref rappel sur l'organisation des réseaux des fournisseurs d'accès et sur leur interconnexion à travers Internet. Ce rappel nous permet d'introduire les limitations des technologies de réseaux existantes et les principaux problèmes de qualité de service qui se produisent dans divers types de réseaux. Nous présentons ces problèmes après avoir défini la qualité de service et nous introduisons le vieux débat sur la nécessité de la QoS dans les réseaux de télécommunication. Plusieurs modèles de QoS existent pour l'Internet : nous détaillons leurs principes. Le routage et l'ingénierie de trafic ont une importance cruciale pour garantir que les ressources du réseau sont utilisées efficacement et que le réseau fournit le niveau de service souhaité. Ainsi, nous introduisons les techniques de routage et d'ingénierie de trafic que les opérateurs de réseaux utilisent de nos jours. Le chapitre décrit d'abord les méthodes d'ingénierie de trafic basées sur les protocoles de routage IP, puis celles basées sur MPLS.

La qualité de service de bout en bout peut apporter d'importants avantages aux utilisateurs finaux, aux fournisseurs d'accès, et aux fournisseurs de service. Dans le chapitre 5, nous mettons en évidence deux problèmes essentiels : la négociation de QoS inter-opérateur et la

mise en œuvre des niveaux de service négociés. En outre, nous introduisons des directions de recherche qui ont été développées dans divers projets et qui visent à résoudre ces problèmes. Le chapitre se poursuit avec la présentation de technologies de gestion de trafic qui constituent la base technologique pour l'approvisionnement inter-domaine dynamique et pour la création de tunnels inter-domaines statiques avec garanties de performance.

La deuxième partie du manuscrit détaille les principaux résultats obtenus au cours de la thèse.

Le chapitre 6 décrit notre travail sur le dimensionnement et l'allocation des ressources pour les réseaux DiffServ. Nous proposons une modélisation des problèmes d'ingénierie de trafic dans ces réseaux. Puis, nous présentons notre algorithme d'allocation de bande passante et des résultats de simulation. Cette étude nous permet de démontrer la complexité des opérations de dimensionnement du réseau pour les réseaux DiffServ et de proposer un modèle simple d'allocation de bande passante qui peut être utilisé pour gérer le trafic de toutes les classes de service mises en œuvre. Dans ce chapitre, nous nous focalisons sur des mécanismes hors ligne d'allocation de bande passante, pour chaque classe de service, dans un réseau DiffServ. En revanche, dans les chapitres suivants, nous étudions des problèmes d'allocation dynamique de bande passante pour fournir des garanties strictes de qualité de service.

Dans le chapitre 7, nous présentons le problème qui consiste à calculer des chemins appropriés pour des flux inter-domaines critiques. Nous décrivons le problème mathématique sous-jacent, qui est appelé le problème de calcul de chemin multi-contraint (MCP). Ensuite, nous étudions sa complexité : nous montrons qu'il est $\mathcal{NP}$-complet dans le contexte considéré et nous expliquons l'origine de cette complexité prohibitive. Plusieurs solutions ont été proposées pour résoudre le problème MCP ; nous montrons qu'elles ne sont pas applicables pour les problèmes de routage inter-domaine, tel que le placement de réseaux privés virtuels (virtual private networks, VPNs) inter-opérateurs, et que des solutions nouvelles sont donc nécessaires. Par conséquent, nous proposons une approche pour résoudre le problème MCP inter-domaine : en raison des contraintes du problème, nous considérons que des algorithmes distribués sont requis. Enfin, nous décrivons l'architecture de telles solutions distribuées et les possibilités existant pour leur mise en œuvre.

Nous exposons les principes des algorithmes de routage contraint dans le chapitre 8. Notre but dans cette démarche consiste à déterminer les éléments qui peuvent être réutilisés pour résoudre le problème de routage *inter-domaine* contraint. Nous expliquons d'abord comment un problème d'optimisation complexe avec plusieurs objectifs peut se traduire en un problème d'optimisation mono-objectif simple, à travers l'utilisation de fonctions de coût. Ensuite, nous analysons séparément deux familles d'algorithmes existants : ceux qui résolvent le problème MCP exactement et ceux qui le résolvent rapidement, mais pas exactement. Nous terminons le chapitre par une brève synthèse des algorithmes les plus connus.

Nous décrivons notre solution exacte pour le problème de routage inter-domaine contraint dans le chapitre 9. Cette solution garantit de trouver les chemins qui sont les plus éloignés des contraintes de routage considérées. Nous intégrons notre algorithme dans une architecture d'ingénierie de trafic qui a été récemment normalisée et nous déterminons les extensions de protocole requises. Puis, nous évaluons les performances de notre algorithme par simulation, dans des scénarios variés. Malgré l'intérêt évident d'une solution exacte pour fournir une base théorique pour des solutions au problème considéré et pour trouver les chemins optimaux, la complexité des solutions exactes est prohibitive dans les réseaux réels. Par conséquent, nous étendons notre algorithme exact afin de scinder les opérations de calcul de chemin en deux étapes : (1) des pré-calculs et (2) des opérations en ligne simples. Cette division est

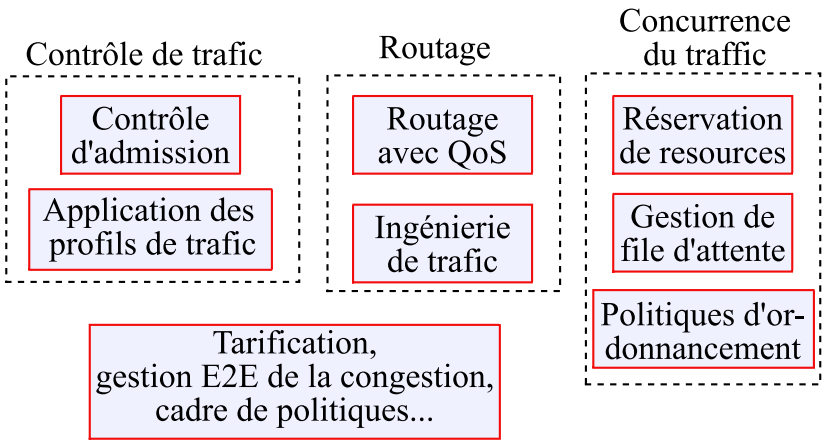

FIG. 1.1 – Une classification des principaux mécanismes de QoS dans les réseaux

intéressante pour diminuer la charge de travail des entités de calcul de chemin, dans des scénarios spécifiques.

Dans le chapitre 10, nous présentons une approche qui donne de bonnes solutions au problème MCP inter-domaine, en temps polynomial. Nous résumons les travaux précédents sur des algorithmes d'approximation pour le problème MCP. Ensuite, nous proposons un nouvel algorithme pour résoudre un problème spécifique dans chaque domaine. La combinaison des résultats obtenus dans chaque domaine pour ce problème permet de trouver des chemins de bout en bout qui sont aussi proches que désiré d'un objectif d'optimisation. Dans ce chapitre, nous montrons que le temps d'exécution dans le pire des cas des algorithmes d'approximation est trop grand pour des calculs de routage en ligne. Par conséquent, nous concluons que des mécanismes plus rapides, tels que des heuristiques, doivent être étudiés.

Nous décrivons un algorithme rapide basé sur notre approche exacte dans le chapitre 11. Cet algorithme est très efficace : dans la plupart des situations, il retourne rapidement un chemin qui satisfait les contraintes de routage. Nous décrivons la méthode que nous mettons en œuvre dans cet algorithme rapide pour réduire à la fois la complexité du calcul et le surcout de signalisation. Ensuite, nous illustrons notre algorithme rapide sur un exemple simple et nous fournissons son pseudo-code. Enfin, nous comparons sa performance à celle d'algorithmes exacts, à la fois de manière analytique et par simulation.

Finalement, la conclusion (chapitre 12), synthétise les contributions de la thèse et propose des orientations pour de futurs travaux.

1.2 Garanties de performances

1.2.1 Vue d'ensemble des mécanismes de QoS

Les solutions de qualité de service impliquent de nombreux mécanismes qui opèrent dans des couches diverses, afin de fournir les niveaux de service désirés. Soldatos *et alii* [157] fournissent une classification intéressante de ces mécanismes et définissent des blocs de gestion de trafic au niveau (1) des paquets, (2) des flux, et (3) du réseau. En s'inspirant de cette classification, la figure 1.1 représente les principaux éléments qui influent sur les performances des services. Nous la décrivons ci-dessous.

Dans les réseaux à commutation de paquets, les flux se disputent les ressources du réseau : le taux d'arrivée des paquets dans un routeur peut dépasser la capacité de sortie de ce routeur. Par conséquent, les routeurs utilisent des files d'attentes pour stocker temporairement les paquets qu'ils ne peuvent pas servir immédiatement, en cas d'arrivées en rafale. Les routeurs utilisent les fonctions de gestion de file d'attente et les politiques d'ordonnancement pour partager la bande passante disponible entre les flux de trafic admis et pour différencier les comportements saut par saut pour chaque classe de service. La configuration de ces fonctions peut être déclenchée par des techniques de réservation de ressources.

Le contrôle d'admission (CAC) est une procédure qui accepte ou rejette les flux entrants, en fonction de l'utilisation du réseau, afin d'éviter la congestion. Le CAC n'accepte un flux que si le réseau dispose de ressources suffisantes pour fournir le niveau de qualité de service requis pour le nouveau flux tout en maintenant la qualité de service convenue pour les flux précédemment admis. Les décisions du CAC dépendent du profil de l'utilisateur et de son droit à utiliser les ressources du réseau. Les contrats de service spécifient un profil de trafic que l'utilisateur doit respecter pour bénéficier du niveau de service convenu. Les profils de trafic peuvent être imposés par l'opérateur, à l'entrée de son réseau, grâce à diverses opérations appelées *policing* (rejeter le trafic en excès), *shaping* (retarder certains paquets en cas d'arrivée de rafales), et *marking* (marquage du trafic en excès de sorte que, en cas de congestion, les routeurs de cœur de réseau rejette ce trafic en priorité).

Le routage et l'ingénierie de trafic jouent un rôle prépondérant pour permettre aux opérateurs d'utiliser efficacement les ressources de leurs réseaux. Premièrement, ils permettent de répartir le trafic afin d'éviter la création de goulots d'étranglement. Deuxièmement, lorsqu'ils prennent en compte les besoins en QoS des applications, le routage et l'ingénierie de trafic offrent la possibilité de router le trafic sur des chemins qui offrent un niveau de performance adapté. Par exemple, le trafic de voix peut-être routé sur des chemins qui offrent un délai de propagation aussi petit que possible. Ces deux fonctionnalités confèrent un rôle important aux fonctions de routage pour garantir la performance d'un réseau.

Les opérations à long terme que les opérateurs effectuent pour garantir que les ressources réseau disponibles à l'intérieur de leur réseau sont suffisantes pour la charge de trafic attendue sont appelées *planification du réseau* et *dimensionnement*. Elles consistent en la conception d'une architecture de réseau et la gestion des ressources du réseau pour supporter des prévisions de trafic. Ces opérations sont complétées par l'*ingénierie de trafic hors-ligne*, qui calcule un routage optimal des flux de trafic prévus dans l'architecture réseau existante. Ce routage prend généralement en compte les besoins des services, l'équilibrage de charge, la protection des ressources réseaux contre les pannes, et des considérations d'optimisation du réseau. L'*ingénierie de trafic en ligne* est utilisée pour déterminer dynamiquement un routage acceptable pour un agrégat de trafic spécifique.

Pour compléter la présentation des mécanismes de QoS, nous devons aussi mentionner l'effet des prix sur les modèles de trafic, l'importance des techniques de gestion de la congestion de bout en bout (*e.g.,* configuration de TCP [5], ECN, PCN [34]) pour adapter le comportement des sources de trafic à l'état du réseau, et le rôle des politiques de gestion pour atteindre une configuration réseau cohérente.

1.2.2 Limitations des technologies actuelles de QoS

La migration des réseaux de télécommunications vers une architecture de prochaine génération (NGN) représente une évolution majeure de l'Internet. Ce dernier devient l'architecture

commune pour un ensemble de services très variés. Les contraintes de qualité de service prennent donc de plus en plus d'importance dans l'Internet, pour permettre de déploiement de services à forte valeur ajoutée. Cependant, des problèmes techniques significatifs[1] doivent être résolus pour permettre d'offrir des garanties de qualité de service de bout en bout dans les réseaux de prochaine génération. La thèse présente l'état actuel des technologies de QoS et les restrictions de ces dernières. En particulier, elle détaille les solutions existantes pour gérer le trafic.

Le routage et l'ingénierie de trafic jouent un rôle fondamental dans la gestion du trafic et la fourniture de QoS. Avant tout, ils permettent d'éviter la congestion en garantissant une utilisation efficace des ressources des réseaux. Par ailleurs, le routage avec QoS et les technologies de réservation de ressources permettent de fournir des garanties strictes de performance. Deux solutions principales existent pour la gestion du routage : l'une se base sur les fonctionnalités du routage IP et l'autre utilise les capacités de l'architecture MPLS. L'ingénierie de trafic basée sur IP est largement répandue, mais elle souffre de quelques restrictions. MPLS bénéficie de fonctionnalités plus étendues et peut être utilisé en conjonction avec DiffServ afin de réaliser un outil puissant de gestion de la QoS.

Globalement, les technologies actuelles permettent aux opérateurs réseau de créer des « ilots de QoS » qui correspondent aux domaines de routage. Dans la thèse, nous décrivons les avancées vers l'interconnexion de ces ilots afin de garantir la qualité de service de bout en bout. La gestion de bout en bout de la QoS et la gestion du trafic inter-domaine sont des sujets importants : ils sont étudiés par de nombreux projets de recherche et par plusieurs organisations industrielles. L'introduction de la gestion de QoS de bout en bout pourrait avoir des conséquences importantes sur les modèles commerciaux des opérateurs. En particulier, de nouvelles architectures, telles que la technologie PCE, fournissent un cadre solide pour l'ingénierie de trafic inter-domaine et pourraient constituer un élément important des futures solutions de QoS inter-opérateur.

La thèse illustre les restrictions des technologies actuelles de qualité de service intra-domaine. Nous décrivons les solutions existantes pour mettre en œuvre la différentiation de qualité de service en se fondant sur des prévisions de trafic et sur un dimensionnement du réseau. Plus précisément, nous proposons un cadre conjoint pour l'allocation de bande passante et l'ingénierie de trafic dans les réseaux DiffServ. En tant que méthode d'allocation de bande passante, notre modèle permet aux opérateurs d'utiliser efficacement les ressources de leur réseau. En tant que modèle de simulation, notre modèle est intéressant pour évaluer la capacité d'un réseau à admettre des flux de trafic supplémentaires. La conclusion principale de notre étude des mécanismes intra-domaine de QoS est que dimensionner le réseau d'un opérateur et provisionner une quantité appropriée de ressources à l'avance pour les flux critiques sont des tâches difficiles.

Les technologies d'approvisionnement dynamique représentent un outil intéressant pour la mise en œuvre de niveaux spécifiques de qualité de service. Nous complétons notre étude des mécanismes de QoS par l'examen des techniques d'allocation dynamique de ressources pour obtenir des garanties strictes de QoS au niveau inter-domaine.

[1] par exemple, les interactions entre opérateurs, les problèmes inter-couches, *etc*

Notation	Signification
$\mathbb{N}$	ensemble des entiers naturels
$[1..n]$	ensemble des entiers compris entre 1 et n
$\mathbb{R}$	ensemble des réels
$G = (V, E)$	un graphe
$G = (V, E, \vec{w})$	un graphe valué
V	ensemble des nœuds
E	ensemble des liens
$\|V\|$ or V	nombre de nœuds
$\|E\|$ or E	nombre de liens
$\vec{w}$	la fonction de coût des liens
l	un lien
$\vec{w}(l)$	les poids d'un lien
$w_k(l)$	le k-ième poids d'un lien
K	le nombre de métriques de lien
$P_{s\to t}$	l'ensemble des chemins de s vers t
$\mathbf{p}$	un chemin
$\vec{W}$	un ensemble de contraintes sur les poids des chemins
W_k	une contrainte sur le k-ième poids
D	un domaine
$\mathbf{S}$	une séquence de domaines

TAB. 1.1 – Résumé des notations employées dans le manuscrit

1.3 Routage inter-domaine multi-contraint

1.3.1 Problème

Dans la thèse, nous utilisons le terme MCP pour désigner un chemin soumis à de multiples contraintes liées à plusieurs métriques. Le problème de calcul de MCP est nommé *problème MCP* et intervient lorsqu'un système doit déterminer un chemin, avec des garanties sur plusieurs critères de performance, entre une source donnée et une destination donnée, dans un graphe.

Nous considérons un réseau représenté par un graphe orienté $G = (V, E)$ où V est l'ensemble des sommets ou *nœuds* et E est l'ensemble des arêtes ou *liens*. Pour modéliser de multiples paramètres de QoS, un vecteur $\vec{w}(l)$ de $K \in \mathbb{N}$ poids réels non-négatifs $w_k(l) \in \mathbb{R}^+$, avec k dans $[1..K]$ est associé à chaque arête l dans E. Les poids $w_k(l)$ représentent la valeur de chaque métrique considérée pour le lien l, donc, nous utilisons les termes *poids de lien* et *valeur de métrique* indifféremment. Nous supposons qu'au moins l'un des K poids associés à un lien l est différent de zéro.

Nous associons chaque chemin $\mathbf{p}$ avec un vecteur $\vec{w}(\mathbf{p})$ de K poids réels positifs $w_k(\mathbf{p}) \in \mathbb{R}^+$ avec k dans $[1..K]$. Les poids du chemin représentent la valeur de métriques additives, nous les définissons donc par :

$$w_k(\mathbf{p}) \equiv \sum_{l \in \mathbf{p}} w_k(l),\ k \in [1..K]. \tag{1.1}$$

Pour définir le problème MCP, nous considérons une *requête de calcul de chemin*. Cette requête précise une source s et une destination t dans V, ainsi que K contraintes W_k dans $\mathbb{R}^{+*}$ avec k dans $[1..K]$ sur le chemin $\mathbf{p}$ demandé. Les contraintes considérées représentent la limite maximum W_k sur chaque poids $w_k(\mathbf{p})$: pour être acceptable un chemin $\mathbf{p} \in P_{s\to t}$ doit vérifier $w_k(\mathbf{p}) \leq W_k$ pour tout k dans $[1..K]$. Nous appelons *chemin faisable* tout chemin qui répond aux contraintes de la demande, et le problème MCP consiste à déterminer un chemin faisable. Par exemple, le problème consistant à trouver un chemin dont le délai de propagation de bout en bout est inférieur à cinquante millisecondes et qui traverse moins de quinze liens est un problème MCP.

Problème (MCP)**.** *Étant donné une source s et une destination t dans un graphe valué $G(V, E, \vec{w})$, un entier $K \in \mathbb{N}$ tel que $K \geq 2$, et K contraintes W_k avec k dans $[1..K]$, trouver un chemin $\boldsymbol{p}$ dans $P_{s\to t}$ qui vérifie $w_k(\boldsymbol{p}) \leq W_k$, pour tout k dans $[1..K]$.*

Déterminer des MCPs nécessite des calculs difficiles : Wang et Crowcroft ont prouvé que le problème MCP avec deux contraintes additives ou plus est $\mathcal{NP}$-complet [180].

Le problème inter-domaine MCP est une généralisation du problème MCP : dans le problème inter-domaine, la source et la destination sont des nœuds dans des domaines différents, alors que le problème MCP ne considère pas la division d'un réseau en domaines.

Nous introduisons quelques définitions et notations supplémentaires afin de formaliser le problème MCP inter-domaine. Nous considérons un réseau représenté par un graphe valué $G = (V, E, \vec{w})$, et nous définissons une partition de l'ensemble des nœuds V. Nous appelons *domaine* tout élément de cette partition. Nous disons qu'un lien $l = (u, v)$ dans E est *intra-domaine*, s'il existe un domaine $D \subset V$ tel que $u \in D$ et $v \in D$. En outre, nous disons qu'un lien $l = (u, v)$ dans E est un lien *inter-domaine* s'il existe un domaine $D \subset V$ tel que $u \in D$ et $v \notin D$. Nous disons que deux domaines sont *voisins* si un lien inter-domaine les relie. Nous appelons *nœud de bordure d'entrée* d'un domaine D tout nœud v dans D pour lequel il existe un nœud u dans $V \setminus D$ tel que le lien inter-domaine (u, v) existe dans E. Nous notons $P^{\mathbf{S}}_{s\to t}$ l'ensemble des chemins d'une source s vers une destination t qui traversent une séquence de domaines $\mathbf{S}$ particulière.

Problème (InterMCP)**.** *Étant donné une séquence de domaines finie et sans boucle $\boldsymbol{S} = (D_1, D_2, \ldots)$ qui contient $|\boldsymbol{S}|$ domaines, une source $s \in D_1$ et une destination $t \in D_{|\boldsymbol{S}|}$, un entier $K \in \mathbb{N}$ tel que $K \geq 2$ et K contraintes $W_k \in \mathbb{R}^+$ avec k dans $[1..K]$, trouver un chemin $\boldsymbol{p}$ dans $P^{\boldsymbol{S}}_{s\to t}$ tel que, pour tout k dans $[1..K]$, $w_k(\boldsymbol{p}) \leq W_k$.*

Le problème MCP inter-domaine est une généralisation du problème MCP, qui est $\mathcal{NP}$-complet. C'est pourquoi le problème MCP inter-domaine est $\mathcal{NP}$-complet lui aussi. Cependant, certaines instances du problème peuvent être résolues en temps polynomial. Nous détaillons ces instances dans la section 7.3.4.

Nous analysons les solutions existantes pour le problème MCP. En particulier, le tableau 1.2 fournit une synthèse des algorithmes les plus connus et de leurs propriétés. Une technique courante pour résoudre les problèmes MCP consiste à transformer un problème d'optimisation multi-objectif en un problème mono objectif (plus simple) grâce à des fonctions de coût linéaires et non linéaires. Spécifiquement, les fonctions de cout linéaires permettent d'utiliser des algorithmes simples de calcul de plus court chemin. Pour réduire la complexité des opérations de calcul, plusieurs techniques sont employées. Par exemple, il est possible de discrétiser l'espace des valeurs de métriques et de borner arbitrairement le nombre de chemins intermédiaires mémorisés afin de réduire la complexité temporel des calculs dans le pire des cas.

Algorithme	Catégorie	Complexité temporelle dans le pire des cas	Référence
Jaffe	Heuristique pour le problème MCP	$\mathcal{O}(V \log V + KE)$	[98]
TAMCRA	Heuristique pour le problème MCP	$\mathcal{O}(\alpha_{\max} V \log(\alpha_{\max} V) + \alpha_{\max}^2 KE)$	[58, 59, 110]
SAMCRA	Exacte pour le problème MCP	$\mathcal{O}(\alpha V \log(\alpha V) + \alpha^2 KE)$	[172, 110]
H_MCOP	Heuristique pour le problème MCP	$\mathcal{O}(V \log V + KE)$	[109]
Yuan's	Mécanisme d'approximation pour le problème de décision MCP	$\mathcal{O}\left(EV\left(\frac{V}{\epsilon}\right)^{K-1}\right)$	[188]
FAST-DMCP	Mécanisme d'approximation pour le problème de décision MCP	$\mathcal{O}\left(E\left(\frac{V}{\epsilon}\right)^{K-1}\right)$	[185]

TAB. 1.2 – Synthèse des solutions les plus connues au problème MCP intra-domaine

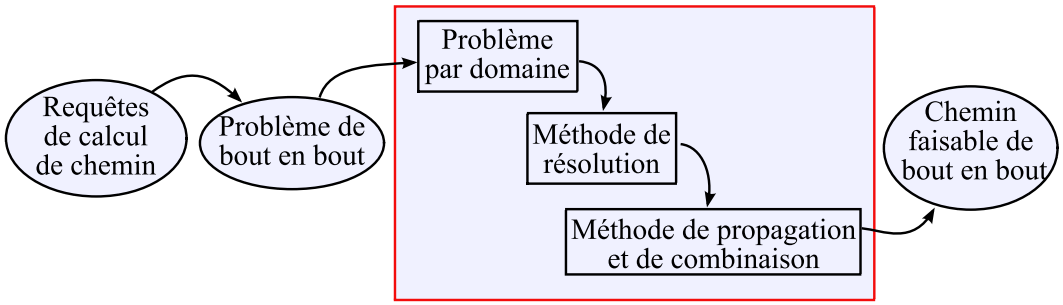

FIG. 1.2 – Structure proposée pour des solutions distribuées au problème MCP inter-domaine

Nous expliquons pourquoi de nouvelles solutions adaptées aux contraintes du problème MCP interdomaine sont requises. En effet, de nombreuses solutions ont été proposées pour résoudre le problème MCP, mais elles ne peuvent pas être appliquées pour résoudre les problèmes de routage inter-domaine contraint, pour des raisons de sécurité (confidentialité), d'autonomie des domaines et pour des contraintes d'extensibilité.

1.3.2 Architecture

Nous proposons une approche générale pour résoudre les problèmes de routage inter-domaine contraint. Notre méthode repose sur *des calculs distribués* qui impliquent la résolution de problèmes par domaine et *des opérations de propagation et de combinaison* des résultats locaux pour obtenir des chemins de bout en bout faisables, à partir des résultats de chaque domaine. Afin de permettre des calculs distribués, nous proposons trois éléments de base,

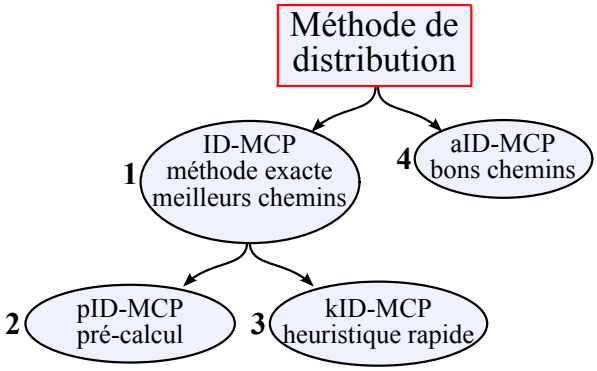

FIG. 1.3 – Synthèse des algorithmes de calcul de chemin inter-domaine que nous avons proposés

qui sont représentés dans la figure 1.2 : (1) la formulation d'un problème local pour chaque domaine traversé, (2) un algorithme qui résout le problème par domaine, et (3) une méthode pour propager et combiner les résultats des problèmes par domaine pour obtenir des chemins faisables de bout en bout. Ces éléments constitutifs donnent naissance à des algorithmes variés, notamment en fonction la formulation considérée pour le problème par domaine. Dans la thèse, nous décrivons plusieurs algorithmes, qui sont tous basés sur la méthode de distribution des calculs mentionnée ci-dessus. Les caractéristiques de ces algorithmes sont décrites dans la figure 1.3.

1.3.3 Algorithmes

Dans les réseaux orientés connections, l'état d'un réseau peut changer entre le calcul et la configuration d'un chemin. Par conséquent, il est intéressant de trouver des chemins qui ont de grandes chances de rester faisables jusqu'à ce qu'ils soient configurés. Dans cette optique, nous proposons une solution nommée ID-MCP [22], qui remplit les exigences de sécurité et d'autonomie des domaines et qui trouve les meilleures solutions possibles pour le problème inter-domaine MCP. Toutefois, la complexité de cette solution est prohibitive pour certaines instances du problème (*e.g.,* réseaux de grande taille). Par conséquent, nous avons conçu une méthode de pré-calcul, appelée pID-MCP, qui permet le calcul hors ligne de représentations des domaines et effectue en ligne seulement des opérations simples [21]. Ce nouvel algorithme permet de réduire la complexité des calculs effectués à la demande pour chaque requête, dans des situations spécifiques. Nous intégrons nos deux méthodes exactes de calcul dans le cadre de l'architecture PCE [66].

Les approches exactes nécessitent des calculs intensifs, car le problème est $\mathcal{NP}$-complet. Par conséquent, des algorithmes d'approximation et des heuristiques sont nécessaires. Dans le chapitre 10, nous avons étudié les méthodes d'approximation pour le problème MCP inter-domaine. Nous avons décrit aID-MCP, un algorithme qui calcule les meilleurs chemins faisables, en temps polynomial et avec une précision garantie. Cette solution montre qu'il est possible d'adapter les techniques d'approximation existantes pour calculer efficacement des chemins inter-domaines contraints. La complexité temporelle de aID-MCP dans le pire des cas est polynomiale, alors que celle des méthodes exactes est significativement plus grande (problème $\mathcal{NP}$-complet). Néanmoins, la complexité des méthodes d'approximation est prohibitive dans certains scénarios, par conséquent, des solutions plus rapides (heuristiques) doivent être étudiées.

Algorithme	**Propriétés**	**Complexité temporelle dans le pire des cas**	**Référence**
ID-MCP	lent, exact	$\mathcal{O}(D\alpha^2K(\alpha V + E + V^2))$	[22], p. 98
pID-MCP	pré-calcul, exact	$\mathcal{O}(\alpha^2K(E + MV^2))$ hors-ligne et $\mathcal{O}(\alpha^4{N_{\text{BN}}}^3)$ en ligne, dans chaque domaine	[21], p. 102
aID-MCP	temps polynomial, calculs approchés	$\mathcal{O}\left(D \cdot E \cdot (\lfloor\theta\rfloor + 1)^{K-1}\right)$	p. 107
kID-MCP	le plus rapide, heuristique	$\mathcal{O}(D \cdot K \cdot V^2)$	[23], p. 125

TAB. 1.3 – Synthèse des performances de nos solutions au problème MCP inter-domaine

Nous avons proposé une heuristique rapide et efficace, appelée kID-MCP [23]. Nous présentons cette heuristique dans le chapitre 11. L'heuristique kID-MCP trouve rapidement des chemins inter-domaines avec garanties de performance. Sa complexité temporelle dans le pire des cas est du même ordre que celle de l'algorithme de Dijkstra. Une vaste étude par simulations montre que notre solution rapide fonctionne bien, à la fois dans des scénarios réalistes et dans les pires cas : kID-MCP trouve un chemin faisable dans la plupart des situations et la performance des chemins calculés est proche de l'optimum.

Le tableau 1.3 synthétise nos algorithmes pour résoudre les problèmes de routage inter-domaine avec contraintes multiples. Les algorithmes d'approximation fournissent des garanties tant sur le succès des opérations de chemin de calcul que sur la qualité des chemins calculés. Toutefois, leur complexité est sensiblement supérieure à celle de notre heuristique. Par conséquent, nous pensons que, dans la plupart des situations, notre heuristique représente un meilleur choix que les algorithmes d'approximation. Le compromis entre la rapidité des heuristiques et la précision des algorithmes d'approximation nécessite toutefois des études complémentaires. Pour conclure, toutes les solutions proposées ont leurs forces et leurs faiblesses : la sélection d'un algorithme spécifique dépend essentiellement du contexte d'application (*e.g.,* taille et topologie du réseau considéré, exigences de réactivité).

1.4 Conclusion et perspectives

Dans la présente thèse, nous avons étudié le problème de la fourniture de services avec performances garanties de bout en bout. Ce problème implique des enjeux importants pour les opérateurs de réseaux et pour les fournisseurs de services parce que les services à valeur ajoutée (*e.g.,* VPN, IPTV) exigent de la qualité de service de bout en bout et parce que de nombreux opérateurs souhaitent migrer leur services de téléphonie sur une infrastructure Internet.

Notre travail a illustré la complexité de la gestion de la QoS : cette dernière exige que de nombreux mécanismes, appartenant à différentes couches réseau, interagissent. Nous avons montré que, de nos jours, il est difficile de garantir un niveau spécifique de QoS pour des services critiques, à l'intérieur du réseau d'un seul opérateur. Fournir des garanties de QoS de bout en bout pour le trafic qui traverse plusieurs domaines de routage, exploités par des entités

diverses, est également compliqué. Aujourd'hui, la coopération et les échanges d'informations entre les domaines traversés sont limités parce que tous les opérateurs essaient d'optimiser les performances de leur réseau et leurs recettes. Nous pensons que l'extension des relations inter-domaines pour permettre une gestion plus efficace du trafic serait bénéfique pour toutes les parties prenantes [20]. Des évolutions du modèle organisationnel originel de l'Internet, selon lequel les réseaux sont autonomes et divulguent peu d'information aux réseaux adjacents, sont déjà observables dans des forums de pré-standardisation et dans certains organismes de normalisation. Notre travail contribue à cette évolution qui vise à permettre de fournir de la qualité de service pour des flux inter-opérateurs.

Nos solutions de gestion du trafic inter-domaine permettent de déterminer dynamiquement les chemins les plus appropriés pour l'acheminement d'agrégats de trafic critique. Nous sommes donc convaincus que nos contributions représentent une étape importante vers une gestion du trafic plus efficace dans les réseaux. Plus précisément, nos algorithmes contribuent à faciliter la transition vers un monde tout-IP grâce à la fourniture de garanties de QoS inter-domaine et grâce à l'ingénierie de trafic inter-domaine.

Notre travail a ouvert de nombreuses perspectives pour des travaux futurs : par exemple, il a conduit à un projet de collaboration fructueux (CITRIC) et a engendré de multiples activités de recherche dont un post-doc., deux stages de master, et deux futures thèses. Beaucoup d'applications de notre travail seraient intéressantes à étudier. Par exemple, nos algorithmes peuvent être adaptés pour calculer des chemins contraints utilisant des liens inter-domaines divers. Cela permettrait aux opérateurs d'équilibrer la charge de trafic sur les liaisons inter-domaine. Cette question est critique pour les opérateurs de réseau parce que les liens inter-opérateurs sont connus pour représenter un point potentiel de congestion. De plus, nos algorithmes pourraient s'avérer utile dans les réseaux militaires, car ils permettent le routage contraint dans des réseaux où l'information de topologie est fragmentée pour des raisons de sécurité.

Notre travail pourrait être étendu dans plusieurs directions de recherche. Il serait intéressant d'analyser plus profondément les compromis entre les solutions heuristiques et les algorithmes d'approximation pour le problème MCP. Une solution idéale devrait fournir des garanties solides de performances (comme le font les algorithmes d'approximation) et avoir une faible complexité de calcul (comme le font heuristiques). En outre, l'étude de scénarios dans lesquels les domaines traversés utilisent différents algorithmes de calcul de chemins (par exemple des algorithmes exacts et des heuristiques) serait utile, par exemple pour permettre un déploiement progressif et adapter la précision du routage à la charge de calcul dans les PCEs. Du point de vue architectural, les compromis entre l'ingénierie de trafic basée sur les protocoles de routage IP et sur PCE pourraient être étudiés de manière plus détaillée. À plus long terme, étudier la fourniture de QoS inter-domaine sous l'angle de la théorie des jeux serait intéressant. Plus précisément, il est nécessaire de fournir des incitations aux entreprises afin qu'elles offrent le niveau de QoS convenu. Des mécanismes de réputation représentent une solution possible à ce problème et devraient être envisagés.

CHAPTER 2

Introduction

2.1 The Internet: a Multi-Service Network Architecture

The early Internet was designed to carry small delay-tolerant flows related to file transfers or to text-based services such as electronic mail. By contrast, today's Internet is the unifying architecture for multiple services including telephony and massive video transfers. For example, IP television introduces large flows (several Mbps) with strict quality of service (QoS) requirements. In addition, more and more operators are interested in migrating telephony flows from the public switched telephony network to new architectures based on the Internet [1], for instance the IP multimedia sub-system (IMS) of the next generation network (NGN) architecture [165]. These tendencies put new requirements on the Internet in terms of service availability, of end-to-end QoS performance, and of network management.

Best-effort service is not acceptable for the most-demanding network applications. For example, frequent packet losses and transmission delays affect the quality of streaming videos, and thus, decrease the end-user quality of experience. Therefore, the use of the Internet to carry critical services requires the deployment of appropriate QoS technologies. Moreover, the rise in traffic volumes and in service diversity increases the cost of over-dimensioned networks. As a result, more sophisticated QoS management strategies become interesting in various scenarios [100]. Several tools exist today, to manage the level of network performance provided by a single operator. However, managing the end-to-end performance of the traffic that traverses the network of several providers is a largely unresolved problem. The present thesis studies these subjects, as explained in the next section.

2.2 Problems Studied in the Thesis

In this section, we briefly introduce the main problems that are studied in the thesis. In particular, Section 2.2.1 presents some important issues faced by network operators to support QoS-demanding services. In addition, we describe our vision of a QoS-enabled Internet, through simple case studies, in Section 2.2.2.

2.2.1 From QoS Islands to End-to-End QoS

QoS has a critical importance in the contractual agreements among service providers and their customers. Therefore, network operators use various mechanisms to guarantee that

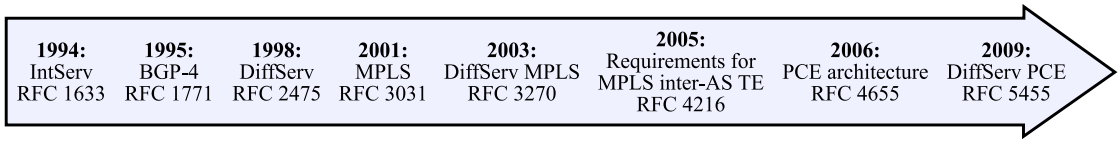

Figure 2.1: A timeline of the standardization work on QoS and on inter-provider traffic management inside the IETF working groups

their network provides the contracted service levels. In this section, we introduce two critical problems of existing QoS technologies:

1. configuring a network to provide a specific level of service
2. providing end-to-end QoS for flows that traverse several networks

Fulfillment of Service Level Agreements

Internet service providers (ISPs) are bound to their customers by particular contracts, named *service level agreements* (SLAs), which describe the performance requirements for the provided services. The technical part of the SLA is called the *service-level specification* (SLS). In the SLS, the contracted level of QoS is often described in terms of classes. For example, a provider might offer a premium service class that guarantees a transmission delay lower than ten milliseconds, a jitter lower than five milliseconds, and a packet-loss rate lower than one percent, for a given traffic volume. Network operators need appropriate mechanisms to guarantee that their network provides a performance level that fulfills the contracted SLAs.

Congestion is a major cause for packet losses and queuing delays. Therefore, the provision of a high level of QoS to specific traffic flows requires that the network always allocates sufficient resources to the data flows of critical services, to avoid congestion phenomena. Network operators must dimension their networks with a sufficient capacity to fulfill the contracts with their customers. This operation involves installing appropriate network equipments, configuring them to use the network capacity efficiently, and contracting agreements with other network providers for the traffic toward external destinations.

The long-term operations that the ISPs perform to guarantee that sufficient network resources are available inside their network are called *network planning* and *dimensioning*. They consist in designing a network architecture and managing the network resources to support a forecasted traffic load. These operations are complemented with *offline traffic engineering*, which computes an optimal routing of the forecasted traffic flows in the existing network architecture. This routing usually integrates service performance, load balancing, protection, and network optimization considerations. *Online traffic engineering* is used to determine dynamically an acceptable routing for a specific traffic aggregate.

Several methods have been standardized to provision appropriate resources for QoS flows. Figure 2.1 presents a few milestones in the standardization work on QoS. It describes standards developed in the main Internet standardization group, the Internet engineering task force (IETF) [87]. The first approach for providing QoS is to reserve resources for every critical flow. It has been developed in IntServ [31], the first QoS model for the Internet. The second possibility is to prioritize critical traffic and to discard first the non-critical traffic, in case of congestion. The DiffServ model [132, 27], which was described in 1998, follows this idea. The third method consists in adapting the routing to the flow requirements, so that critical flows

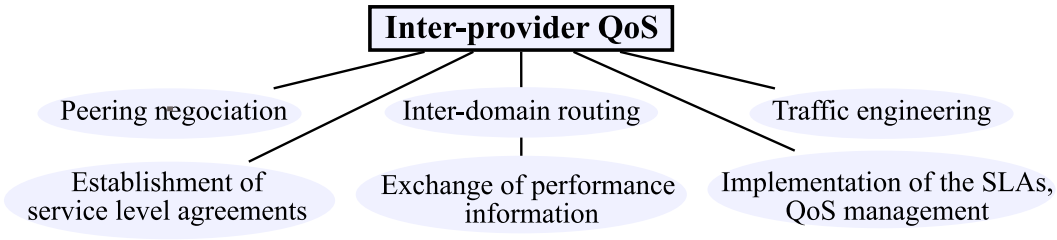

Figure 2.2: Some domains where work is required to facilitate QoS operation across multiple provider networks

follow paths with guaranteed performance. The MPLS traffic-engineering architecture [145], described in 2001 by IETF, provides the technology to implement this third solution. Usually, these three approaches are mixed, to provide flexible QoS management.

The available QoS mechanisms enable operators to administrate the QoS inside their network, and thus, to create "QoS islands". Nevertheless, it is hard to guarantee that the performance of the network will meet a specific SLS. For example, the difficulty of implementing particular per-hop behaviors led IETF to publish DiffServ configuration guidelines [10] in 2006, eight years after the publication of the initial DiffServ standards.

Provision of End-to-End QoS for Inter-Domain Flows

The provision of QoS is required for many applications. The most well-known telecommunications application that needs QoS is probably the telephony: to limit the mouth-to-ear delay and the jitter, VoIP networks require QoS mechanisms. Another important domain where QoS is required is the provision of VPN services, which enable a company to interconnect its sites in a secure and cost-effective manner, using a converged IP network rather than expensive private wires. VPN offers are widely adopted by most medium-sized and large businesses in several developed countries, and especially, in the United States [104]. These technologies necessitate QoS guarantees to support business-critical services, which demand a better performance level than the one provided by the public Internet. VPNs provide a similar service as leased-line offers for critical flows.

QoS must be considered as an end-to-end problem, because end-users observe the end-to-end performance level experienced by their services. However, the control of the traffic is shared among the traversed networks, which manage the traffic locally and without end-to-end perspective. Thus, managing the end-to-end QoS of inter-domain traffic is intrinsically hard and is still a largely unresolved problem. The "Workshop on Revisiting IP QoS" at SIGCOMM 2003 revealed the doubts of many researchers about the deployment of end-to-end QoS in the Internet. One of the main problems that hinder the development of QoS in the public Internet is probably the lack of a clear business model with a demonstrated profitability. The incentives to deploy QoS technologies are quite limited because several technical problems must be solved before end-to-end QoS can be supported in production networks. For instance, network operators require technologies to ease the coherent treatment of inter-domain QoS flows.

Figure 2.2 depicts several domains where work is still required to enable the provision of inter-provider QoS. We list below some of the problems that must be solved.

- **Selfishness:** Carriers are selfish entities bound by competitive relationships; nevertheless, the provision of end-to-end QoS requires a certain level of inter-provider cooperation as well as enhanced information exchanges among the involved network domains.
- **Business incentives:** The business cases for inter-provider QoS must be clarified. The main issue is to define a commercial model that allows varied revenue-sharing and cost-sharing strategies for the involved providers. Several business cases can be imagined; for example, national operators might ally to extend their footprint and to compete with larger carriers.
- **Peering connections and inter-domain provisioning:** The establishment of peering connections among operators and the provisioning of specific QoS levels for inter-provider services require time-consuming human interventions. Introducing an acceptable level of automation in these processes might simplify the operation of inter-provider services.
- **Traffic management:** Implementing a per-hop behavior and providing a transit performance that fulfills a contracted service level agreement is difficult.
- **Coherence:** Providing a coherent packet treatment in several networks, administrated by different operators, is complex and requires uniform QoS policies.
- **Traffic separation:** flows from customers of other networks shall not degrade the performance of a network's own customers.

Recently, several important steps toward the provision of end-to-end QoS for inter-domain flows have been achieved inside the IETF bodies. For example, IETF issued a set of requirements for inter-domain traffic engineering with MPLS [189] in 2005 (Figure 2.1). The IETF's work on an architecture that fulfills this set of requirements resulted in the publication of several RFCs that describe the path computation element (PCE) framework [66]. In parallel, the IETF has extended resource reservation protocols for managing inter-domain flows [64]. These protocols represent an important element of traffic-engineering architectures, and thus, their extension is a major advance toward the provision of inter-domain QoS and toward fine-grained inter-domain traffic engineering. Specifically, it has allowed the PCE working group of IETF to provide ISPs with an inter-domain traffic management framework [175, 176].

2.2.2 Our Vision of a QoS-Enabled Internet

In this section, we illustrate our vision of a QoS-enabled Internet through simple examples that describe the coordinated provision of a particular end-to-end service level for inter-domain traffic.

Access with Guaranteed Performance to Third-Party Services

Figure 2.3 presents a case figure where an operator guarantees the network performance for its users when they access a specific service provided by another ISP. Conversely, an operator might sell a specific performance level for third-party service providers that want to reach its subscribers. The implementation of these scenarios would open the way to novel business models for the access providers. ISPs could seize this opportunity to:

- **differentiate their offers to the end-users by the provided QoS level for specific services.** For instance, an access provider could guarantee an excellent quality of

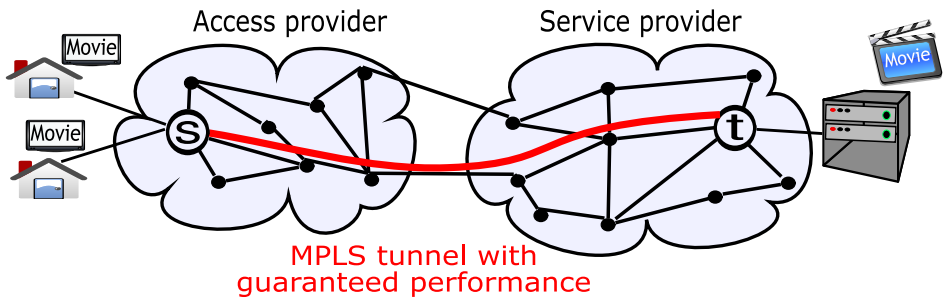

Figure 2.3: Provision of a path with guaranteed performance by an access operator for a specific service

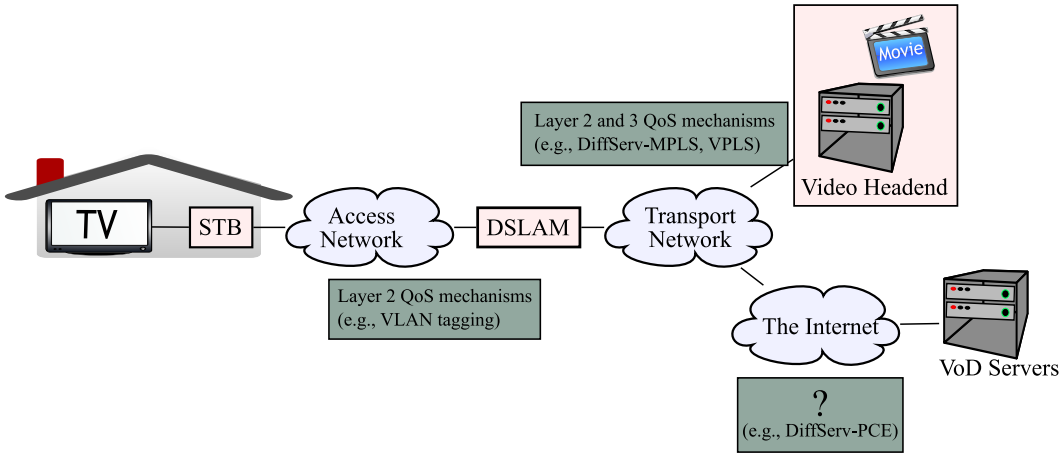

Figure 2.4: A practical example: QoS issues in IPTV networks

experience for popular third-party services to attract and retain customers (*e.g.,* access the `http://hulu.com` free online video service with a superior quality of experience). Conversely, it could guarantee this quality of experience to the service providers in a wholesale scenario.

- **extend their footprint.** In particular, national service providers could compete with larger carriers and provide an international VPN service thanks to their agreements with other providers. Moreover, a national access provider could provide a service with guaranteed performance for customers in foreign countries.

Another important scenario occurs when the backbone and the service-provision network of an operator are operated independently by different entities. In this situation, the network operator might setup a few static inter-domain paths with guaranteed performance between various locations of its network to provide performance guarantee for specific services.

Figure 2.4 illustrates some of the aforementioned scenarios on a practical case: IPTV networks. Several technologies are available to provide QoS in access and backbone networks. However, the mechanisms to provide end-to-end QoS for flows that cross the Internet (*e.g.,* in the figure, the flows from the VoD server to the customer household) are not mature. Specifically, it is hard to ensure that sufficient resources are provisioned end-to-end for critical flows and that those flows are routed along high-performance paths.

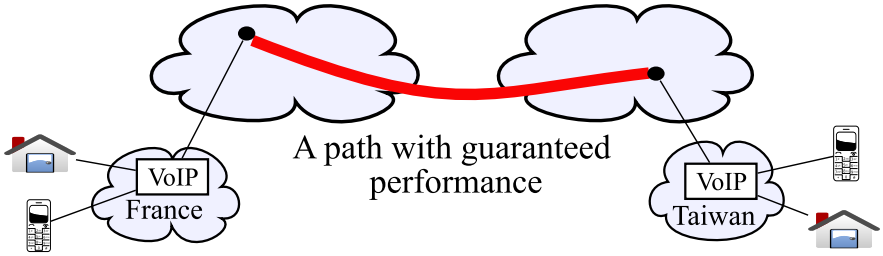

Figure 2.5: Provision of a path with guaranteed performance through two transit operators in the best-effort Internet

To make the provision of end-to-end QoS possible for inter-carrier flows, many technical problems must be solved. In particular, the negotiation of inter-connection agreements typically requires much time and efforts from the operators. In addition, the performance level of inter-domain paths depends on the configuration of every traversed domain. Finally, determining an appropriate route and configuring an appropriate per-hop behavior inside every traversed domain are complex tasks.

QoS-Enabled Transit Provider Inter-Connection

Figure 2.5 presents a second example: here, an intermediate QoS path is setup between two transit operators. Their inter-connection through a QoS path enables these transit operators to provide enhanced service level agreements to their customers and to their peers. In particular, configuring paths with different performance levels allows transit carriers to differentiate their offers and to answer the varied needs of their customers. For example, consider two ISPs with nation-wide networks that want to inter-connect their telephony over IP (ToIP) gateways. These ISPs need paths through the network of their providers and with a reduced delay, a limited jitter, and a guaranteed availability. Thus, they might request a QoS enabled inter-connection from their providers.

Alternatively, the stub domains might represent two indirectly-connected IMS domains that establish video sessions. Due to the absence of a direct inter-connection, the video traffic must transit through third-party networks, which would usually treat it as best effort. As video traffic is typically vulnerable to losses and delays, the service performance might be unacceptable. To avoid this problem, the stub domains would probably desire to setup an inter-domain connection with guaranteed performance [137].

Several technical and business issues must still be investigated before the dynamic provision of inter-domain transit paths with guaranteed performance becomes possible in production networks. For instance, the negotiation of service level agreements among the transit carriers takes time and requires human interventions, as Matos *et alii* [126] explain. Furthermore, network operators often use different class-of-service definitions, which complicates the coherent treatment of the flows inside every traversed domain.

2.3 Scope and Objectives of the Thesis

2.3.1 Path Computation Algorithms

The MPLS-TE mechanisms enable routing traffic aggregates along specific paths with guaranteed performance. Recent extensions of the MPLS-TE technologies (*e.g.,* [176, 64]) enable network operators to configure constrained paths that traverse domains managed by different operators and to reserve resources along these paths. The *algorithms* to compute these paths in a way that optimizes service and network performance are not studied in IETF, which focuses on the development of *protocols* and *architectures*. One of the purposes of this thesis is, thus, to complement the IETF work on inter-domain traffic engineering by proposing appropriate path computation algorithms.

Constrained path computation algorithms provide a path that matches a set of constraints. Network operators require such algorithms for several applications. For example, they use constrained path computation to:

- adapt the routing of incoming traffic flows to the current network condition dynamically (online traffic engineering),
- manage the resource allocation in their networks on the admission of new traffic flows (dynamic network provisioning),
- route tunnels (*e.g.,* VPNs) with performance criteria [126].

In all these cases, the considered path constraints often include both QoS (*e.g.,* delay, jitter, losses, and reliability) and operational considerations (*e.g.,* resource consumption, load balancing).

The present thesis addresses the fundamental problem of constrained routing: the computation of paths subject to constraints on multiple metrics, such as delay, number of traversed links, and packet-loss probability. As explained in subsequent chapters, this problem has already been well investigated, because of its important applications. However, previous work did not consider the inter-domain routing problem, which is subject to several specific requirements, such as the confidentiality of topology information and the autonomy of every domain. Thus, existing constrained routing algorithms cannot be applied to compute inter-domain constrained paths. Consequently, the present thesis investigates the constrained path computation problem in the inter-domain case.

2.3.2 Main Contributions

Our main domain of activity involves the study of inter-domain routing problems [25, 26, 23, 22]. Figure 2.6 depicts the context of our work in this area. Our studies include the three following aspects. (1) *We propose novel path computation algorithms.* These algorithms must be fed with information about the network state and topology. Thus, (2) *we evaluate the requirements introduced by novel inter-domain traffic management mechanisms on the routing protocols.* Connection-oriented architectures, such as the PCE framework, require path-setup (signaling) operations before traffic can transit through a new path. Consequently, (3) *we also examine the available signaling mechanisms.* At a more abstract level, we analyze the current trend toward enhanced inter-carrier interactions and their consequences [20].

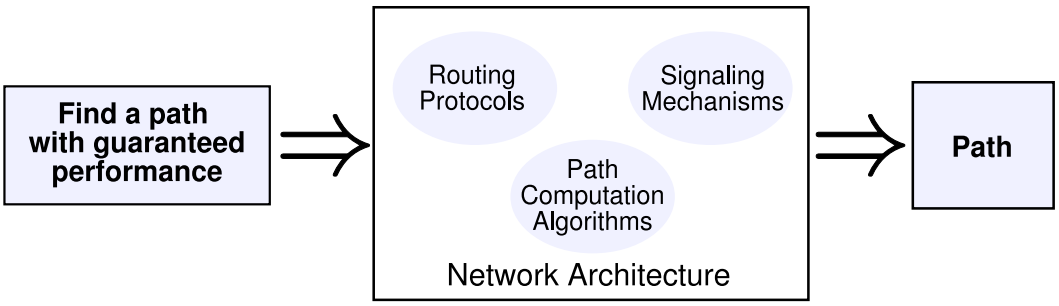

Figure 2.6: Schematic representation of the thesis work on inter-domain traffic management

To synthesize our contributions, we propose protocol and architecture extensions as well as novel path computation algorithms that better fulfill the requirements of the inter-domain path computation problem, compared to existing solutions. Our proposed solutions represent a fundamental element of future QoS-enabled network architectures.

2.4 Organization of the Manuscript

The remainder of the present dissertation is divided into two parts. Part I introduces the technological context for our studies and presents the required background to understand the work carried out during this thesis.

We begin Part I by explaining the recent evolution of Internet toward a unified multi-service architecture. We introduce the most well-known architecture for convergent networks, in Chapter 3. The presentation of this architecture and of its limitations is important to understand the traffic management problems that network operators currently face and that motivate our work.

In Chapter 4, we present a short reminder on the organization of access provider's networks and of their inter-connection through the Internet. This reminder is important to introduce business-related limitations of existing networking technologies and the main QoS problems that occur in varied types of networks. We present some of these problems after defining QoS and introducing the long-lasting debate on the need for QoS in telecommunications networks. Several QoS models try to solve these problems in the Internet: we detail their principles. Moreover, routing and traffic engineering have a critical importance to guarantee that network resources are used efficiently and that a network provides the desired service level. Thus, we introduce the routing and traffic-engineering techniques that network operators use nowadays. The chapter first describes intra- and inter-domain traffic-engineering methods based on IP-routing protocols and continues with the ones based on MPLS.

End-to-end QoS can bring important benefits to end-users, access-providers, and service providers. In Chapter 5, we give a synthesis of the end-to-end QoS management issues in the current Internet. Specifically, we highlight two essential problems: the negotiation of inter-provider QoS and the enforcement of the negotiated service level. Furthermore, we introduce research directions that have been developed in various projects to solve these problems. The chapter continues with the presentation of recent inter-domain traffic management technologies, which provide the technological basis for provisioning inter-domain resources dynamically and for setting up static inter-domain tunnels with guaranteed performance.

The second part of the dissertation details the main results obtained during the thesis.

Chapter 6 describes our work on network dimensioning and resource allocation in DiffServ networks. We propose a modeling of traffic-engineering problems for DiffServ networks. Then, we introduce our bandwidth-allocation algorithm and the simulation results. This study enables us to demonstrate the complexity of the network dimensioning operations for DiffServ networks and to propose a simple bandwidth allocation model that can be used to engineer the traffic of all the supported classes of service.

In Chapter 7, we present the problem of computing appropriate paths for inter-domain critical flows. We describe the underlying mathematical problem, which is called the multi-constrained path (MCP) problem. Then, we study its tractability: we show that it is $\mathcal{NP}$-complete in the considered context and we explain the origin of this prohibitive complexity. Several solutions have been proposed for the MCP problem; we argue that they are not applicable for inter-domain routing problems, such as the placement of inter-provider VPNs, and that novel solutions are needed. Thus, we propose our approach to solve the inter-domain MCP problem: because of the problem requirements, we consider that distributed algorithms are the most appropriate for our problem. Finally, we describe the building blocks for distributed solutions and the possible implementation directions.

We exhibit the principles of existing constrained routing algorithms in Chapter 8. Our purpose in this process is to show the elements that can be reused to solve the inter-domain constrained routing problem. We first explain how a complicated multi-objective optimization problem can be translated into a simpler single-objective optimization problem through the use of path-length functions. Then, we analyze separately two families of existing algorithms: the ones that solve the MCP problem exactly and the ones that solve it rapidly but not exactly. We conclude the chapter with a short comparison of the most well-known algorithms.

We describe our exact solution to the inter-domain constrained routing problem in Chapter 9. This solution guarantees to find the paths that are the furthest from the considered routing constraints. We integrate our algorithm in a recently-standardized traffic-engineering framework and we determine the required protocol extensions. Then, we evaluate the algorithm performance in various simulation scenarios. Despite the clear interest of an exact solution to provide a theoretical foundation for future solutions to the problem and to find the optimal paths, the complexity of exact solutions is prohibitive in real networks. Therefore, we enhance our exact algorithm to split the path computation operations into two stages: (1) pre-computations and (2) simple online operations. This division is interesting to decrease the per-request computational burden on the path computation entities, in specific scenarios.

In Chapter 10, we present an approach that provides good solutions to the inter-domain MCP problem, in polynomial time. We detail related work on approximation algorithms for the MCP problem. Then, we provide a novel algorithm to solve a specific per-domain problem. The combination of the per-domain results provides feasible end-to-end paths that are as close as desired to an optimization objective. In that chapter, we show that the worst-case running time of approximation algorithms is prohibitive for online routing computations. Therefore, we conclude that faster mechanisms, such as heuristics, must be studied.

We describe a fast algorithm based on our exact approach in Chapter 11. This algorithm is very efficient: in most situations, it provides a path that satisfies the routing constraints and it computes this path rapidly. We describe the method that we implement in this fast algorithm to reduce both the computational complexity and the signaling overhead. Then, we illustrate our fast algorithm on a simple example and provide its pseudo-code. Finally, we compare its performance to the one of exact algorithms, both analytically and by simulation.

Finally, the concluding chapter, Chapter 12, synthesizes the contributions of the present thesis and proposes directions for future research.

Part I

Background and Technological Context

CHAPTER 3

Evolution Toward Convergent Network Architectures

3.1 Introduction

The convergence of telephony and data networks is one of the most important evolutions of telecommunications networks in the past few years. This change has introduced many new requirements on the Internet to support the varied needs of the supported services. In particular, it has brought the need for novel network mechanisms, such as the ones presented in our thesis, to support end-to-end QoS and to enhance routing and traffic management.

We present the architecture of next-generation convergent networks, in Section 3.2. One of the purposes of convergent network architectures is to support services with varied requirements on a common architecture. In particular, the support of QoS-demanding applications, such as telephony or IPTV, requires advanced QoS management techniques. We describe the NGN mechanisms for QoS management in Section 3.2.2. The IP multimedia sub-system (IMS) is an important element of the NGN architecture; we present it in Section 3.3. We introduce the limitations of IMS, and more generally, of the NGN architecture in Section 3.4.

3.2 The Next-Generation Network Architecture

3.2.1 A Unifying Infrastructure

The recent evolution toward the offering of multiple telecommunications services has complicated network architectures. In particular, diverse access technologies, such as DSL, Ethernet, UMTS, are used for the same services. Therefore, network operators have designed novel architectures to support heterogeneous services on a single infrastructure based on IP, the *next generation network* (NGN) architecture [40, 95, 94].

As depicted in Figure 3.1, the NGN architecture divides network functions into three planes: service, control, and transport[1]. This formalism can be seen as a simplification of the well-known OSI model [92]. The main idea behind the separation of the three planes is to enable services to be technology-independent and networks to be service-independent. The

[1]The term *transfer plane* is also commonly used to refer to *transport plane*.

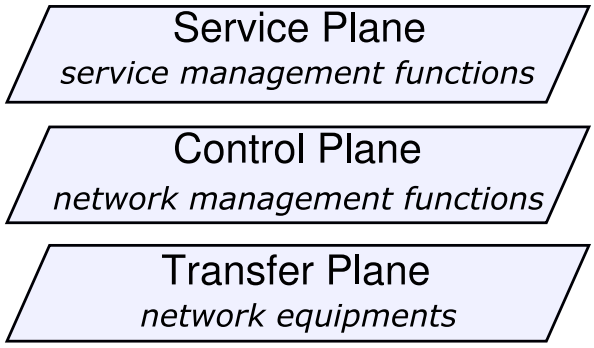

Figure 3.1: The three-plane formalism of NGN architectures

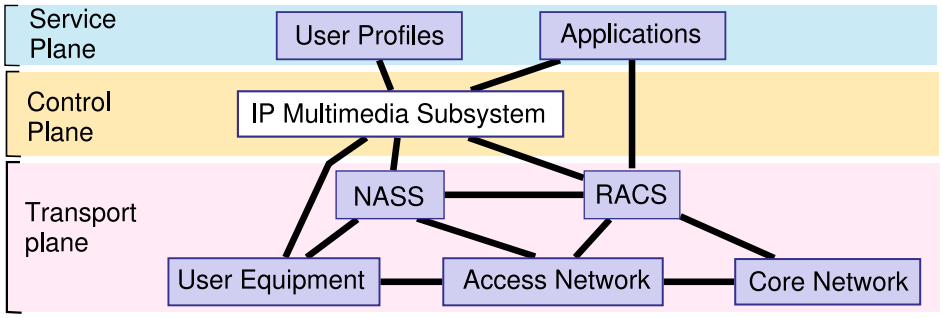

Figure 3.2: The NGN architecture and the IP multimedia sub-system

service plane consists of service management functions, the *control plane* is used to manage the network and should be independent of the networking technologies, and the *transport plane* is made up of the network equipments [29].

Figure 3.2 presents an overview of the NGN architecture. The core of NGNs is the *IP multimedia sub-system* (IMS) [165, 96, 18], a set of functions standardized mainly by ETSI and that enable the transport of various multimedia contents. The IMS is complemented by several additional sub-systems, which take care of specific aspects of the service provision. For instance, ETSI describes a resource admission control sub-system (RACS) [166], which controls the admission of new calls. Similarly, ITU defines the resource admission control function (RACF) that enables real-time, session-based resource control for various services and various networking technologies [97].

3.2.2 QoS Management in Next-Generation Networks

The level of QoS that can be provided in NGNs determines the services that can be deployed. As a result, QoS delivery is critical in such networks and QoS management functionalities are integrated at several levels of the NGN architecture.

- In the service and the control plane, the session initiation protocol (SIP) enables the negotiation of communications parameters before a call actually starts. The end-user terminals exchange SIP messages with SDP contexts that describe their communication capabilities, to agree on a set of mutually acceptable settings.

- The service-level QoS mechanisms interact with the control-plane equipments to check the admissibility of the calls considering the operator procedures. In particular, the RACS can retrieve information from transfer-plane equipments to take policy decisions regarding the admissibility of new calls. In addition, it influences the marking of the packets in differentiated services (DiffServ) networks, and is able trigger the reservation of specific resources in the network with the resource reservation protocol (RSVP) [158], to support the admitted calls.

- The enforcement in the transfer plane of the QoS management decisions relies mainly on the IP edge nodes and on the border nodes. These equipments use various mechanisms, such as the ones described in Section 4.2, to provide the required performance to the service flows. We refer the interested readers to Reference [164], which provides a more extended synthesis of the QoS mechanisms for IMS networks than we could provide in the scope of the present thesis.

3.3 The IP Multimedia Subsystem

3.3.1 Main Components

The IMS is used for session and media control in NGNs. Its structure is depicted in Figure 3.3 [166] and its functional components are the following:

- The call session control function (CSCF) establishes, monitors, supports and releases multimedia sessions and manages the user's service interactions. It can play three different roles: serving-, proxy- or interrogating- CSCF (S, P and I-CSCF) [166]. The S-CSCF is the proxy server controlling the communication session. It invokes the applications servers related to the requested services. It is always located in the home network [38]. The P-CSCF is the IMS contact point for the SIP user agents.

- The multimedia resource function controller (MRFC) is used for controlling a multimedia resource function processor (MRFP) that essentially provides transcoding and content adaptation functionalities [51].

- The breakout gateway control function (BGCF) "selects the network in which PSTN breakout is to occur and—within the network where the breakout is to occur—selects the MGCF" [165]. This means that it is used for inter-working with the circuit switched domain.

- The media gateway controller function (MGCF) is used to control a media gateway.

3.3.2 Strengths of the IMS Architecture

IMS is designed to allow substantial network infrastructure and management savings, therefore improving cost effectiveness. It should decrease the investment threshold for new service deployment thanks to a uniform service delivery platform: it provides a set of common functions called *service enablers* that can be used by several services (*e.g.,* group/list management, presence, provisioning, operation and management, billing...). This makes service implementation much easier and faster. Moreover, it allows a tight interaction between several services.

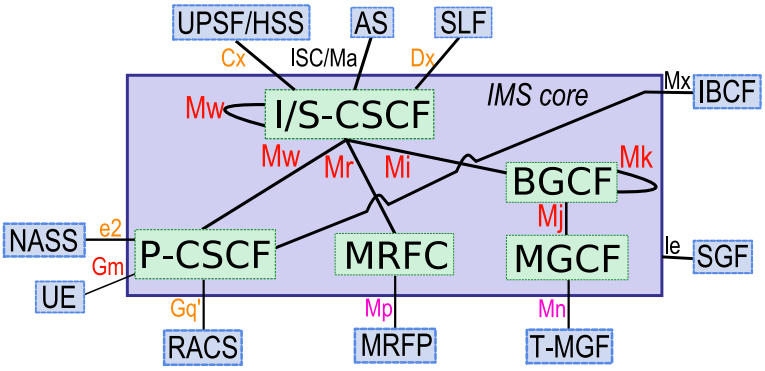

Figure 3.3: IMS internal structure and interfaces [166]

While the average revenue per user is decreasing for several network operators, IMS is seen by many as a solution for network operators to be "more than bit pipes", as explained in an eponymous paper [51]. Indeed, it allows the network operator to play a central role in service delivery, and to bundle attractive services with their basic-access offer. The combination of several services in one session, the single sign-on and unified billing are expected to raise customer's interest and to increase the revenue opportunities. In IMS, the operator is aware of the actual services that the customer is using. Therefore, appropriate billing schemes can be developed [38].

The RACS/RACF Component

Policy-based QoS control allows a network operator to configure easily the network equipments. It is essentially used to define admission control rules and to facilitate the translation of business level contracts such as service-level specifications (SLSs) and service level agreements (SLAs) into network level policies. The network policy rules are defined by the operator in the policy decision point (PDP). This network element is used for taking policy decisions. It answers the requests emitted by a policy enforcement point (PEP).

The main network functions involved in QoS provision in IMS network are:

- The P-CSCF, which is aware of the SDP context of the session, and thus, of the resources required for this session.
- The RACS, which takes policy decisions and interfaces with transfer functions.
- The transport-layer components, which apply the policy decisions.

The main transport-layer functions are listed below:

- The *resource control enforcement function* (RCEF) enforces policies under the control of the A-RACF. It opens and closes unidirectional filters called *gates* or *pinholes*, polices traffic and marks IP packets [166].
- The *border gateway function* (BGF) performs policy enforcement and network address translation (NAT) functions under the control of the S-PDF. It operates on unidirectional flows related to a particular session (micro-flows) [166].

- The *layer-two termination point* (L2TP) terminates the layer two procedures of the access network [166].

The admission control usually follows a three-step procedure:

1. authorization of resources
2. resource reservation
3. resource commitment

3.4 Limitations of Next-Generation Networks

3.4.1 From a Functional Architecture to a Real Network Infrastructure

It is to be noted that the NGN standards describe essentially a functional architecture. Therefore, the ISPs require to map the network functions to existing equipments and to map the specified interfaces to appropriate protocols. Moreover, the capabilities of the architecture NGN are limited by the ones of existing equipments and protocols.

Below, we provide a description of existing protocols that are used in deployed NGN/IMS architectures. Most of these protocols are standardized by IETF.

- The main signaling protocol used in IMS is called the session initiation protocol (SIP) [148]. The main purpose of SIP is the establishment, modification, and termination of multimedia sessions between two terminals. The body of SIP messages is described using the session description protocol (SDP). The SDP is a syntax for describing media flows (address, port, media type, encoding, *etc.*) standardized in [146]. The IMS SIP is an enhanced version of SIP that includes several extensions to handle subscriber management, service control, single sign on, QoS authorization, billing, resource management, *etc*, as described in the 3GPP TS.24.229 standard [2]. Most of these numerous extensions have led to IETF RFCs (*e.g.,* [39, 147, 124, 81]).

- Diameter is a recent authentication, authorization, and accounting protocol that replaces Radius. It is defined in [37]. In the IMS service framework, Diameter is used mainly by the I-CSCF, S-CSCF and the application servers in their exchanges with the element that contains the user profiles: the HSS.

- The common open policy service (COPS) is a protocol defined in [62]. It supports policy control over QoS signaling protocols (*e.g.,* RSVP) and is used to convey policy requests and decisions between policy decision points (PDPs) and policy enforcement points (PEP).

- MeGaCo, also called H.248, is a successor of the media gateway control protocol (MGCP) used for controlling media serving functions in an IMS environment. It is specified in ITU-T Recommendation H.248.

- The real-time protocol (RTP) provides transport functions for transmitting real-time data. It is specified in [152] and is used in conjunction with a control protocol called real-time control protocol (RTCP) to allow monitoring the data delivery.

It is to be noted that the use of IPv6 is mandatory in standard-compliant IMS networks, but several equipment vendor implementations support both IPv4 and IPv6.

3.4.2 The Difficulty of Implementing End-to-End QoS Management

The provision of end-to-end QoS guarantees is a highly desirable feature in NGNs, for several reasons. First, the idea of transforming a best-effort IP network by introducing end-to-end QoS guarantees is an important driver for the development of IMS: convergent networks aim at supporting QoS services, such as telephony or video-streaming. This is a key consideration because the services that can be deployed on the IMS architecture are restricted by the level of QoS, and the value is assumed to reside in QoS demanding services, such as real-time multimedia applications.

One of the main motivations for IMS networks is to facilitate the development and the deployment of third-party services, which can be hosted in other networks. However, the provision of end-to-end QoS guarantees for flows traversing several domains remains a largely unresolved problem in NGNs (as explained in the introduction). Some advances on application-layer mechanisms for managing inter-domain QoS can be cited. For instance, the most recent standard versions, such as [167], provide various scenarios in which service-based policy decision and admission control functions are distributed across several domains. In addition, they describe the requirements on network functions to implement such scenarios. Nevertheless, the required transfer-plane technologies for providing end-to-end QoS for inter-provider flows are not available yet.

3.5 Conclusion

In the chapter, we have presented the NGN architecture for convergent networks. In particular, we have described the NGN functions involved in the provision of QoS. These functions play a fundamental role to enable the deployment of QoS sensitive services, such as telephony or television, on the Internet. We have explained the strengths and weaknesses of the NGN and IMS architectures. In particular, the provision of end-to-end QoS is an essential requirement for enabling the deployment of value-added services in these architectures. Nevertheless, the technologies for providing end-to-end QoS for flows that traverse several domains, belonging to a single or to several operators, are in their infancy. One of the purposes of the present thesis is; therefore, to provide solution elements toward the provision of QoS performance guarantees for services hosted in distant networks and reachable trough one or more transit domains. We present related work on this topic in the next chapter, before describing our contributions in the remainder of the dissertation.

Key points of Chapter 3

- The migration toward NGN architectures represents a major evolution of the Internet.
- End-to-end QoS is an important requirement of NGNs, to enable deploying value-added services.
- Significant technical problems must be solved to enable the provision of end-to-end QoS in NGNs (*e.g.,* inter-carrier interactions, cross-layer issues).
- We present relevant QoS technologies in the next chapter. In the next part, we enhance existing traffic management mechanisms to support end-to-end QoS.

CHAPTER 4

Existing QoS and TE Solutions for the Internet

4.1 Introduction

The main purposes of the present chapter are to provide the readers with the required background to understand our contributions, and to show some limitations of existing technologies that our work tries to solve.

We first remind a few important facts about the organization of the Internet, define QoS, and describe the most well-known QoS models for the Internet, in Section 4.2. The available QoS models are effective when the network is stable (absence of failures) and the links are not congested. However, both congestion and service disruption occur frequently in the Internet [4, 162]. Therefore, routing and traffic engineering play an important role in network operation. They ensure network connectivity along paths that use the network resources efficiently and that provide the right level of QoS. For example, traffic related to bandwidth-demanding services should be routed along paths with sufficient available bandwidth and delay-sensitive traffic should be routed along paths with low transmission delays.

In Section 4.3 and Section 4.4, we provide an overview of the routing and of the traffic-engineering techniques used in the Internet. We show that the traffic-engineering techniques that base on IP routing suffer from several limitations. One of the purposes of our work is to extend these mechanisms for providing QoS guarantees to inter-domain flows. Thus, we describe MPLS, which provides enhanced traffic management functionalities. Our solutions build up on this technology.

4.2 Quality of Service

4.2.1 Short Reminder on the Organizational Model of the Internet

From Access Networks to the Internet

In the Internet, a set of routers and of hosts administrated with homogeneous policies by an ISP is called a *domain*, or more specifically, an *autonomous system* (AS).[1] This name highlights the distributed nature of the Internet, which is a set of inter-connected independent

[1] An ISP can divide its network into several ASes and in smaller routing areas for administrative reasons. Routing areas are often named *domains* too.

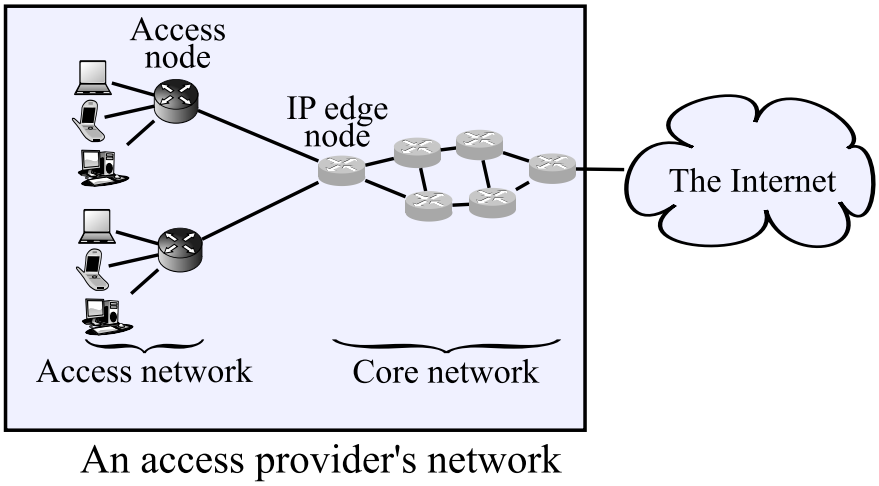

Figure 4.1: Segmentation of access provider's networks

networks without central authority. The Internet is made up of numerous domains, which are identified by their AS number; at the time of writing, about 31000 ASes are advertised and 15000 more are allocated but not advertised by their owner [91].

Access provider's typically structure their networks in *segments*, as depicted in Figure 4.1 and in the example of Figure 2.4. *Access* networks connect the end-users to the provider's equipments. *Core* networks represent the center of ISP's networks and carry highly aggregated traffic flows. The intermediate segment that connects access and core networks has various names depending on the profile of the access provider. For example, it can correspond to metropolitan and regional aggregation networks.

Inter-Provider Connectivity

Three main entities can be involved in the provision of communication services in the Internet [29]. *Network providers* administrate a network to provide connectivity among distant locations. *Service providers* use this connectivity to provide various services to their *customers*. A single entity can fulfill various roles, for example, access providers often deploy services for their customers and buy a transit service from other ISPs. Therefore, the generic term *ISP* is often used to refer to both network providers and service providers.

ISPs inter-connect routers of their core network to routers of other ISPs to ensure global Internet connectivity to their customers. The physical infrastructure that enables peering ISPs to exchange Internet traffic between their ASes is often called an *Internet exchange point*[2] (IXP). Network operators typically use their inter-connections with other ISPs in the IXPs to improve fault-tolerance and to reduce their dependency on their upstream providers.

The relationships between inter-connected ISPs are typically described by contractual agreements. Gao [80] lists the simplest forms of inter-connection agreements between two neighboring ASes. Customer-provider relationships bind a *provider* that provides a paid transit service to a *customer* ISP. Peering relationships describe the agreement of two ISPs (*peers*) to exchange traffic between their networks free of charge for mutual benefit. Figure 4.2

[2]The alternative terms *network access point* (NAP) and metropolitan area exchange (MAE) are often used to refer to IXPs.

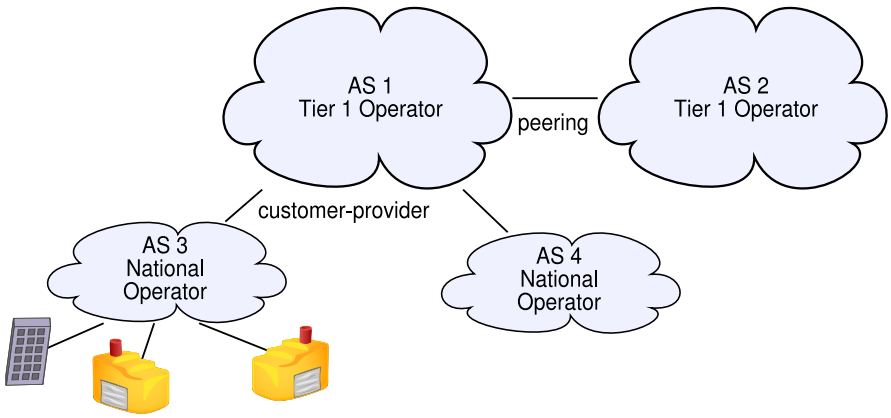

Figure 4.2: The hierarchical organization of the Internet

illustrates the two kinds of relationships. In particular, it shows that customer-provider relationships are oriented and typically occur between a large (*e.g.,* international or national) and a smaller size network (*e.g.,* regional).

A few Internet domains appear in a large fraction of the inter-domain routes. These domains are densely inter-connected and represent the core of the Internet, through which most traffic transits. In particular, a few carriers, called *tier-1* operators, have a global connectivity to the Internet; that is, they do not need to buy transit services from any other carriers to reach any other connected domain of the Internet. Such domains are represented at the top of Figure 4.2. For example, a simple analysis (Figure 4.3) of inter-domain routing tables shows that the French research and education network, RENATER, is reachable through Sprint and Verizon, which are two well-known tier-1 operators.

4.2.2 Do We Need QoS?

Quality of service can be defined as the set of elements that determine the performance of a service and its ability to fulfill the user expectations. In this definition, the user can be a service provider, an ISP that buys bandwidth for transit traffic, or an end-user. For example, the concept of *quality of experience* (QoE) corresponds to QoS from the point of view of the end-user.

Access represents a network segment where congestion is likely to occur, because the network capacity is limited and expensive. Thus, it is a widely-acknowledged fact that QoS mechanisms are required for access networks. However, there is a periodical debate about the opportunity of implementing QoS technologies inside aggregation and core networks. The most frequent criticism asserts that QoS mechanisms could introduce more complexity in the network management and increase the operational costs. Thus, network over-provisioning appears as an attractive solution to guarantee that, statistically, the traffic flows experience an appropriate level of QoS. However, over-provisioning is not a panacea and does not solve the following problems.

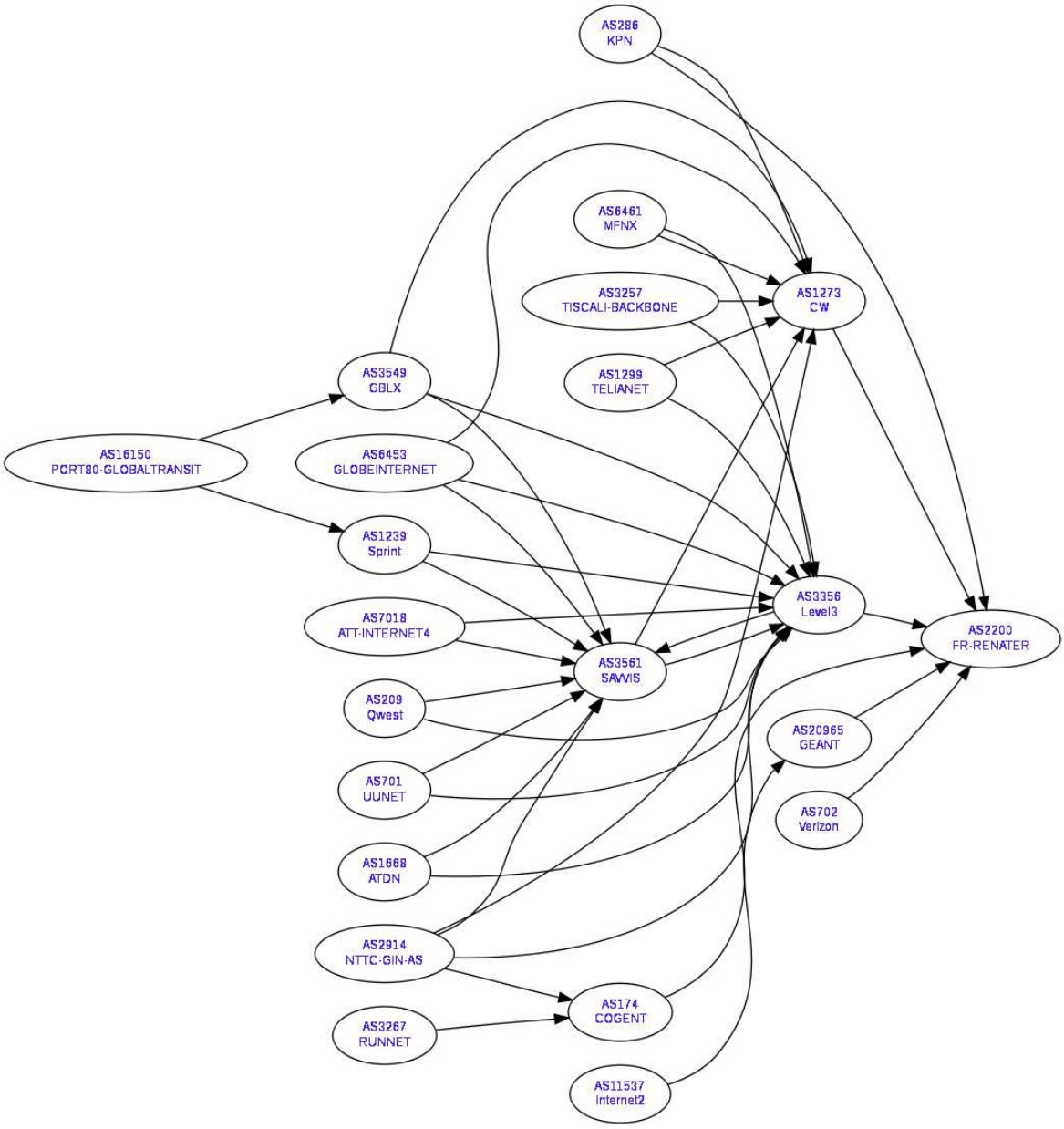

Figure 4.3: Example of AS-paths toward RENATER (reproduced with permission from `robtex.com`)

- Over-provisioning alone does not provide hard QoS guarantees because of the best-effort handling of IP traffic: for instance, it is possible (although unlikely) that congestion occurs on bottleneck links, if the traffic load is not balanced over the available links. Consequently, over-provisioning strategies must be complemented with efficient traffic-engineering mechanisms.

- Over-provisioning has a non-negligible cost: the bandwidth consumption is rapidly growing because of the evolution of the Internet-access rates sold to the end-users. For instance, passive optical networks (PONs) [41] provide bit-rates that exceed hundred megabits per second for the end-users. In the case of a wide adoption of these access-

rates, frequent increases of the core network capacity may be required to maintain an appropriate over-provisioning margin. Specifically, adding bandwidth on certain links, such as ISPs inter-connections in network access points, is expensive. Therefore, capacity expansion operations could require prohibitive investments for the network operators.

- Some consumers are ready to pay for inter-connections with better QoS levels (*e.g.,* multi-site companies, regional VoIP providers).

For all these reasons, and because of the proliferation of mission critical IP applications, it is now widely agreed that over-provisioning must be complemented with appropriate network management mechanisms for providing end-to-end QoS guarantees at an affordable price [34, 157].

4.2.3 Quantifying QoS: QoS Metrics and Classes of Service

Various indicators called *QoS metrics* measure the level of QoS that service requires or experiences from the network point of view. Several QoS metrics can be used to adapt the routing of critical flows. They can be classified as bottleneck, additive, and multiplicative [45]. A metric is *bottleneck*[3] if its value for a path is the value on a specific bottleneck link: the link with the minimum value of the metric. For example, bandwidth is a bottleneck metric. A metric is *additive* if its value for a path is obtained by adding the values for each link of the path. For example, the propagation delay and the hop-count are additive. Finally, a metric is *multiplicative* if its value for a path is obtained by multiplying the values for each link of the path. In particular, quantities related to event probabilities can often be expressed as multiplicative metrics. For instance, if we assume that the packet-loss probabilities on the links of a path are independent, then the probability of successful transmission is the product of one minus the loss probability on every link of the path. Note that bounded multiplicative metrics can be translated into additive metrics through simple operations, with logarithm functions.

Most services can be regrouped in *classes of service* (CoS), which describe their constraints on the value of relevant QoS metrics, according to their QoS requirements. For example, many services featuring streaming video content with a specific codec have similar needs in terms of bandwidth, delay, jitter, and packet loss. Several definitions have been proposed for usual classes of service, as explained in [10, 154].

4.2.4 The Diversity of the QoS Problems

End-users observe the end-to-end performance level experienced by their services. Therefore, QoS must be considered as an end-to-end problem. However, ISPs face various problems to guarantee QoS in all network segments. In particular, bandwidth is usually a scarce resource in the access segment (*e.g.,* wireless networks). Hence, ISPs require QoS mechanisms in access networks to guarantee that appropriate resources are available for every service. On the contrary, core networks are often over-dimensioned: they offer a bandwidth that represents several times the expected amount of traffic. Thus, the likelihood that the packet-forwarding queue of a router is full when a packet arrives is small, which decreases the packet loss ratios. In addition, the average queue occupation is low in most situations, which decreases the queuing delays.

[3] Wang [180] uses the term *concave*.

Due to the diversity of the QoS problems, network operators provide various types of QoS guarantees. In particular, they provide strict or relative guarantees on the values of QoS metrics. *Strict QoS* gives absolute bounds on certain QoS metrics. For instance, strict QoS enables operators to guarantee that critical services are not affected by congestion. The provision of such QoS guarantees relies on several mechanisms, such as admission control and per-flow resource reservations. *Relative QoS* ensures that in average the value of specific QoS metrics is better for certain flows compared to others. In particular, relative QoS relies on the differentiation of the flows and on specific router configurations to provide a preferential treatment to certain flows in case of congestion. For example, ISPs can use packet marking to identify the class of service of the packets, traffic policing to limit the traffic amount that is admitted into the network for a specific class of service, as well as appropriate scheduling and queue management policies to share the network resources among the supported classes of service.

4.2.5 QoS Models for the Internet

The Internet protocol (IP) is designed to provide *best-effort* connectivity among various locations. This means that IP does not provide any guarantee that packets reach their destination or that they experience a specific QoS level. In particular, traffic flows are aggregated and compete for shared link resources. For example, when the traffic load is high, packets can be dropped or can experience long queuing delays in routers. Guarantees about the successful delivery of data can nevertheless be provided using higher-layer technologies like the transmission control protocol (TCP) [140].

The most well-known QoS solutions for the Internet are the integrated services (IntServ) [31] and the differentiated services (DiffServ) [27] models. IntServ enables resources reservations to guarantee strict QoS. DiffServ defines mechanisms for relative QoS thanks to service differentiation. The main idea behind these models is that an operator that requires a high level of QoS for specific flows does not necessarily need to dimension his network to give this QoS level to all traffic. DiffServ and IntServ offer the means to provide a privileged treatment only to the flows that require QoS, and thus, to save some expensive network investments.

The Integrated Services Model

IntServ defines mechanisms to express service types, to quantify resource requirements for every service and to determine the availability of the requested resources in the routers [17]. In particular, it supports two main classes of service: guaranteed service [155] and controlled load [181]. The former provides strict bounds on end-to-end *queuing* delays and enables ISPs to provide both delay and bandwidth guarantees. The latter uses admission control to protect the services from network overload.

The IntServ architecture uses the resource reservation protocol (RSVP) [32, 182] to request resources from the network. RSVP signaling relies on messages that follow the same path as the data traffic and create a flow state, which describes the traffic characteristics of every data flow. As the Internet uses a non-connected approach, re-routing can occur during a communication. Therefore, RSVP uses a soft-state approach: the state must be periodically refreshed to ensure that it is still valid on the current path.

Through signaling and resource reservation at flow scale, the IntServ model offers a fine-grained control over QoS; however, the number of managed states rises with the number of flows. In addition, each reservation requires a non-negligible amount of message exchange. Furthermore, in a network with many flows, maintaining queues for every micro-flow and assigning packets to these queues introduces a non-negligible processing overhead [47]. Hence, the IntServ model suffers from scalability limitations, which motivate Baker *et alii* [12] to extend it for the management of *aggregate* flows instead of single micro-flows. Similarly, Ping Pan [135] describes a hierarchical reservation model that aggregates the resource reservations at application-layer and at provider-level. Aggregating the flows enables IntServ to maintain a lower number of states, and thus, enhances the scalability of this QoS model.

The Differentiated Services Model

In the DiffServ model, IP flows are identified at the entry of the network. Usually, this identification is based on the contents of the IP header in the incoming packet: the source and the destination address, the value of the DiffServ field, and the upper layer protocol [44]. It can also take into account the source and destination ports in the transport header of the incoming packets. Then, the packets are marked with a DiffServ code point (DSCP) that indicates their QoS class and that influences their treatment in the routers.

Compared to IntServ, DiffServ suppresses per-flow state management and per-flow signaling. Consequently, it is more appropriate than IntServ for core networks, where the number of flows is large. More precisely, DiffServ relies on the classification of flows into *classes of service*. The DiffServ model supports three main families of service classes: the default *best-effort* (BE) behavior, *expedited forwarding* (EF) [57], and *assured forwarding* (AF) [85]. AF is divided into various service classes to enable different treatments inside each family. For example, critical data traffic could be mapped to AF1x and streaming-video traffic to AF2x. Every group is again sub-divided into several drop precedence values (*e.g.,* AF11, AF12, AF13), to differentiate the priority of the traffic inside a class.

ISPs configure scheduling and queue management so that the packets belonging to every class of service experience a particular packet forwarding performance, named *per-hop behavior* (PHB). For instance, an ISP can reserve a bandwidth share in the routers for premium traffic. Nevertheless, configuring routers to implement a particular PHB is non-trivial [10]. For instance, best-effort traffic must not be starved because of excessive resource reservations for QoS traffic, while, QoS traffic must be favored in presence of congestion. To solve these problems, the network operators require either to over-dimension their network or to supervise the resource allocations in their network. In particular, Nichols *et alii* [133] have proposed the use of a *bandwidth broker* (BB) to allocate and to control the bandwidth shares. A BB is an agent that monitors the resource allocations inside a domain and accepts or rejects new requests for using network resources. Several papers, such as [191, 121, 60], have described architectures based on BBs. The BBs of neighboring domains might be inter-connected to provide end-to-end resources allocations, as proposed in [136]. However, Cuevas [52] states that there are no clear inter-bandwidth broker standards.

The service differentiation principle of the DiffServ QoS model is used in varied technologies. For example, in access networks, ISPs can use a particular VLAN identifier or the 802.1p priority field, instead of the DSCP, to tag critical Ethernet flows. Then, they can differentiate the traffic treatment in the switch queues depending on the frame tagging. Similarly, the 802.11e standards define service differentiation mechanisms for WLAN networks [157].

Synergy of IntServ and DiffServ Principles

The IntServ and the DiffServ QoS models are complementary. IntServ enables end-to-end reservations of quantifiable amounts of resources for every flow and provides feedback on the admissibility of the reservation requests. DiffServ considers larger traffic aggregates and scales better. As a result, Bernet *et alii* [17] have proposed to combine both models to reserve resources along data paths for aggregate flows.

It is important to understand that the definition of expected packet treatments through PHBs in the DiffServ model is on a *per-hop basis*; on the contrary, the IntServ approach defines *end-to-end* performance requirements [155]. This represents a fundamental difference between the two approaches and can be considered as a weakness of DiffServ for providing end-to-end QoS, because network operators can use different PHB definitions for the same class of service. Thus, it is difficult to guarantee that inter-domain QoS flows receive a coherent treatment inside every traversed network. To solve this problem, Levis and Boucadair have proposed the use of meta-QoS-classes. These classes enable different providers to certify the support of a set of applications that bear similar network QoS requirements [114].

Both IntServ and DiffServ require that the network is appropriately dimensioned: the network capacity must be sufficient to support the provided services and must be efficiently used. Nevertheless, there is a fundamental difference between these two QoS models. IntServ reserves network resources when they are need for a flow. Thus, IntServ can provide guaranteed QoS. On the contrary, DiffServ relies on dimensioning operations that the operator performs in advance. If this dimensioning is appropriate for the actual resource usage, then, the QoS can be guaranteed for premium traffic.

If the dimensioning is inappropriate, if the network resources are inefficiently used, or if a failure occurs, then, the IntServ and DiffServ QoS can fail to provide an acceptable level of QoS for critical traffic. In fact, several studies have shown that congestion and failures occur frequently in the Internet [4, 162, 89]. As a result, ISPs require network management techniques to efficiently use the network capacity and to delay their investments while maintaining the QoS. This idea motivates the concept of traffic engineering (TE), which describes the set of tools and of techniques that ISPs use to optimize resource usage in their network without degrading QoS. We describe the main existing TE methods in subsequent sections. Their presentation is important for dissertation, because our work focuses on resource provisioning and TE mechanisms.

4.3 IP-Based Traffic Engineering

Traffic engineering is an essential aspect of network management: it consists in all the tasks that network operators do to maintain the performance of their network while avoiding unnecessary expenses. As it tries to reach network performance objectives, TE is closely related to QoS. Some TE methods even enable operators to allocate resources for specific flows, as in the IntServ model. In the present section, we describe the TE methods that base on IP routing protocols.

4.3.1 Fundamental Properties of IP-Routing

In this section, we present some fundamental properties of IP-routing. We describe the two-level routing hierarchy of the Internet, which splits intra-domain and inter-domain operations.

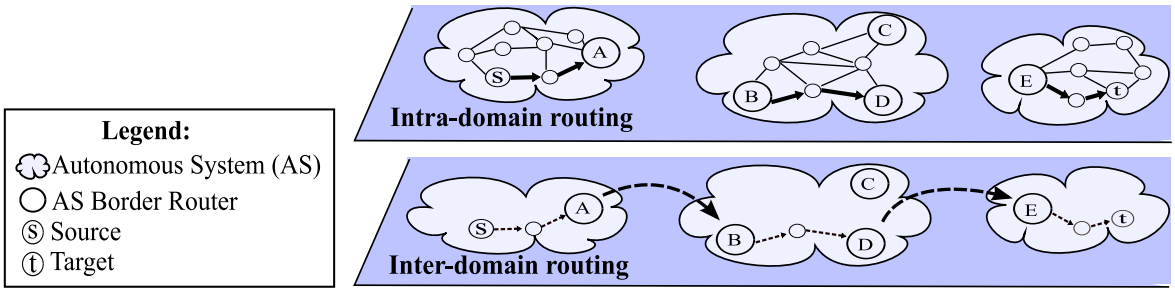

Figure 4.4: The two-level hierarchy of IP routing

This separation structures the IP-based traffic-engineering methods: intra-domain and inter-domain traffic management rely on different techniques. We present both types of techniques and their limitations in subsequent sections.

The Two-Level Routing Hierarchy

In the Internet, routing operations are conceptually separated into two levels, as represented in Figure 4.4. *Intra-domain routing* determines a route toward every destination in the considered domain. *Inter-domain routing*[4] determines the domain sequence and the border nodes that the packets must traverse to reach a specific destination network outside the considered domain. Intra-domain routing protocols are often called interior gateway protocols (IGPs), whereas inter-domain routing protocols are commonly named exterior gateway protocols (EGPs). There are some fundamental differences between intra- and inter-domain routing. Intra-domain routing depends only on the destination of the traffic, and provides the same service to all intra-domain nodes. On the contrary, inter-domain routing is selective because, at inter-domain level, enforcing business policies is more important than obtaining connectivity with all neighboring ASes. Thus, an AS rarely provides a transit service for all the adjacent ASes and toward all destinations.

Routing Algorithms

With IP routing, a router that receives a packet retrieves the destination address from the IP header and looks up the outgoing interface that is registered in its routing table for this destination with the *longest prefix matching* algorithm [11, 168]. After determining the relevant outgoing interface, the router forwards the packet. As the same forwarding operations are repeated for every hop, this process is described as *hop-by-hop* forwarding.[5] This procedure is called the *destination-based forwarding* paradigm.

The routers exchange information that enables them to select an appropriate route toward every destination. The procedure that they use to dialog is the *routing protocol*. *Routing algorithms* use the information advertised by the routing protocols to compute the routes and to populate the routing tables. The information advertised by the routing protocols depends

[4]The exact term is *Inter-AS* routing because the word "domain" can also refer to routing areas, but the denomination inter-domain routing is more common.

[5]IP also supports source routing, mainly for diagnostic analysis, but for security reasons many operators block all IP traffic that employs the source route option [139, 63]. In addition, a technique named *equal-cost multi-path* (ECMP) routing enables routers to use several next-hops for a given destination to provide load balancing among redundant paths [163].

on the routing algorithm used by the routers. There are essentially three different categories of routing algorithms [128]: distance-vector, path-vector, and link-state. We provide examples for each of these categories in the next paragraph.

Several IGPs, such as the routing information protocol (RIP) [123] and the enhanced interior gateway routing protocol (EIGRP) [48], implement a *distance-vector* algorithm at intra-domain level. This algorithm requires that the routers advertise the distance and the next-hop to reach every destination.

The border gateway protocol (BGP) [144], the *de facto* standard inter-domain routing protocol of the Internet, uses a *path-vector* algorithm. This algorithm implies the advertisement of a next-hop and of a domain sequence to reach each destination network.

With the distance-vector and the path-vector algorithms, the routers do not need to know all the router relationships for the entire network. By contrast, the open shortest path first (OSPF) protocol [131] relies on a *link-state* algorithm at intra-domain level. This algorithm requires that every router maintains a representation of the complete topology of the considered domain, including a cost for each link.

4.3.2 Intra-Domain Traffic Engineering

In this section, we present the main TE methods based on IP routing protocols. Network operators use two main techniques to introduce QoS and TE concerns in intra-domain routing. We describe both of them in the following paragraphs. As IP routing typically uses a single path for every destination network, the main purpose of IP-based TE is to calculate this path in a way that preserves the capacity of the network.

An appropriate *metric choice* influences the routing decisions and can be used to select paths that provide the best trade-off between performance and usage of critical resources. For example, ISPs often define the link weights for OSPF as inversely proportional to the nominal link capacity[6], to advantage the use of the links with a large nominal capacity. EIGRP [48] extends this approach and optimizes a composite metric that combines delay, bandwidth, load, and reliability metrics.

ISPs commonly use a second method to improve the routing. They *compute a set of link weights that leads to optimal routing decisions* with the considered routing algorithm.[7] Then, they inject these weights into the routing protocol. This way, the routing can be adapted periodically to the actual condition of the network. Wang *et alii* [179] proved that any routing configuration can be reached by appropriately setting the link metric values. Nevertheless, this method suffers from significant limitations: optimizing link weights requires hard computations: Fortz and Thorup [74, 75] have proved that optimizing OSPF link weights to balance the traffic load for a given set of traffic demands is $\mathcal{NP}$-hard[8]. Furthermore, the adjustment of the IGP weights triggers routing updates and can introduce instability in the routing.

[6] The nominal capacity of a link is the capacity of the link interfaces in the absence of traffic load.

[7] Clearly, the definition of the optimality depends on the operator's objectives.

[8] We refer readers to [82] for definitions of "NP-hard" and other concepts in complexity theory that are not defined here.

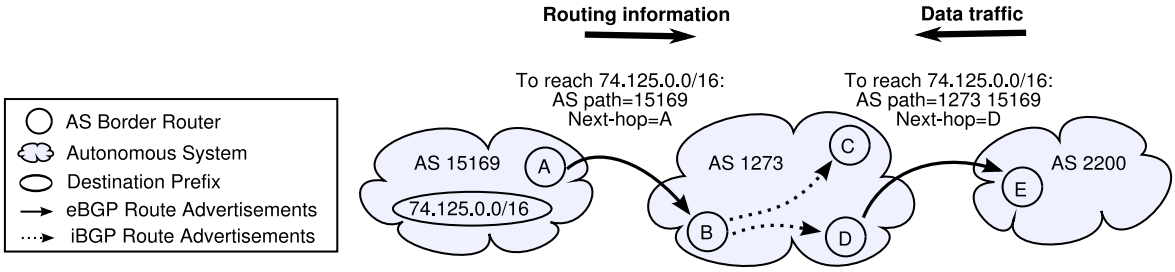

Figure 4.5: Fundamental principles of BGP

4.3.3 Inter-Domain Traffic Engineering

The Border Gateway Protocol

The variety of the commercial relationships among Internet providers has several consequences on networking technologies. For example, inter-connected ISPs must exchange information so that end-hosts can learn routes to distant networks. However, the exchanges of information among the ISPs must be reduced for several reasons. Primarily, the associated communication overhead must be limited to operate the networks in a scalable and efficient way. Moreover, ISPs have a restricted trust in each other, and thus, want to preserve the confidentiality of sensitive information. For instance, an ISP should not advertise routes through its network to domains from which it is not supposed to receive transit traffic. In addition, the ISPs usually refuse to disclose information about the organization of their network (*e.g.,* its topology). As a result, every domain possesses only partial information about the inter-network. Furthermore, this partial information depends on the inter-connection agreements of the domain with other domains and on the policies of the neighboring domains. In summary, each domain has only a *local visibility* on the Internet and inter-domain routing protocols provide *policy-based routing*.

Inter-domain routing in the Internet relies on BGP [144]. To allow communications between hosts in different ASes, this protocol uses a *path-vector* routing algorithm, which is illustrated in Figure 4.5. The BGP border routers of every AS advertise network reachability information to a selection of neighboring ASes, which is determined in accordance with the operator's *route export policies*. A BGP router advertises a route in two situations: (1) the destination of this route is located inside its domain and it wants to make it reachable from other domains, (2) it has received the route from another domain to which it provides a transit service.

BGP route advertisements include the *AS-path*, which consists in a list of ASes that can be followed to reach a particular destination network, and several *path attributes*. In the AS-path, every traversed AS is identified by its unique *AS number*. Route advertisements describe each destination network by a block of adjacent IP addresses, represented by a *network prefix* and a network mask (or prefix length). For instance, in the figure, AS 15169 advertises a route for the prefix 74.125.0.0/16, which represents a block of IP addresses from 74.125.0.1 to 74.125.255.255. The neighboring domain, AS 1273, forwards this route both internally to its BGP routers and externally to neighboring ASes. As a result, AS 2200 learns the advertised route and can use the corresponding AS-path to send traffic toward the hosts in the network 74.125.0.0/16.

As represented in Figure 4.5, BGP relies on two different mechanisms named iBGP (internal) and eBGP (external) to convey the route advertisements inside an AS or to neighboring

ASes. These advertisements rely essentially on the BGP `UPDATE` message and uses TCP as transport protocol, to manage the retransmission of lost messages. In particular, iBGP maintains a full-mesh of TCP sessions among the internal BGP routers, to enable these routers to exchange BGP messages. This requirement clearly introduces scalability concerns, and thus, various alternatives to full-mesh inter-connections have been proposed. For example, route reflectors [15] can be used to limit the number of connections among the BGP routers.

A BGP router can receive several routes, from various routers, for the same destination prefix. Therefore, BGP routers uses route filters and a sophisticated decision process to select the routes that are introduced into the routing table and advertised to neighboring BGP routers. The best-path selection process is based on the calculation of a *degree of preference* for each route. This preference degree usually depends on both the path attributes and the router configuration.

When several routes for the same destination exist and they have the same degree of preference, the BGP routers apply multiple tie-breaking rules successively to decide which route should be advertised [144]. These rules begin by considering all equally preferable routes to the same destination, and then, selecting routes to be removed from consideration. The tie-breaking algorithm terminates as soon as only one route remains in consideration. A commonly used tie-breaking rule specifies that the routes which do not have the lowest value of the multi-exit discriminator (MED) attribute must be discarded. The lowest-MED rule enables ISPs to control how traffic leaves their networks, as explained by McPherson and Gill in Reference [127]. In addition, ISPs frequently use a tie-breaking rule that removes from consideration the AS-paths that do not have the lowest number of hops. This rule enables these ISPs to favor the shortest domain-level paths, which usually perform better than longer paths.

BGP-Based Traffic Engineering

Engineering inter-domain traffic is a strategic problem for network operators. For example, stub domains[9] want to minimize the cost that they pay to their transit providers. Thus, they need to optimize the routes followed by their outgoing traffic. On the other hand, transit providers wish to balance the traffic load that they exchange with their peers, to use efficiently the available network capacity and to avoid congestion. Conversely, a transit provider might want to divert all traffic from a backup path that must be used only in case of failures. Nowadays, network operators rely essentially on BGP features to engineer their inter-domain traffic. Usually, the methods based on adapting BGP configurations to engineer traffic are called *BGP tuning*. The engineering of the incoming and of the outgoing traffic relies on different mechanisms that we present in the following paragraphs.

ISPs can tune the configuration of their BGP routers to engineer their outgoing traffic. The main tools for such tasks are the inbound route filters (Figure 4.6) and the BGP decision process. For example, an ISP can support the forwarding of its transit traffic through a domain A rather than through another domain B by allocating a higher degree of preference to the routes received from A than to those received from B. This ability is particularly important to enforce commercial relationships with adjacent ISPs.

BGP enables ISPs to engineer the traffic that enters in their network, through the adaptation of the route attributes for the outgoing route advertisements and the outbound route filters. For example, Quoitin *et alii* [142] describe *path prepending*, a technique that uses

[9] We call *stub domains* the domains that do not provide a transit service.

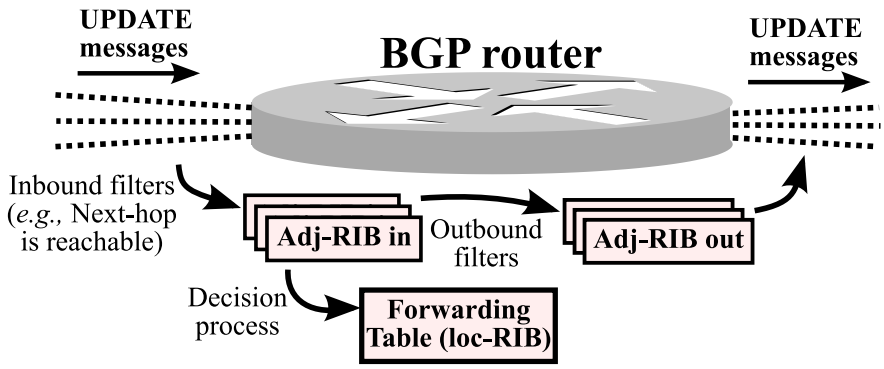

Figure 4.6: Route filtering and route selection operations in a BGP router

the shortest AS-path tie-breaking rule and increases artificially the AS-Path length of certain routes to make them less preferable. Many other techniques are also available: an operator might advertise different routes on different links to control the inbound traffic more accurately [177], or agree with neighboring domains on the definition of BGP communities [43].

For preserving the scalability of inter-domain routing, it is desirable to reduce the amount of information exchanged by the routers. Therefore, BGP advertises limited information about the available inter-domain paths. However, the reduction of routing information limits the QoS routing and TE capabilities of BGP. In particular, the route advertisements include little information about the QoS level of the paths. In addition, each router typically advertises only a single route to a particular destination network and this route can differ from the one that best fulfills the QoS constraints for a given traffic class.

BGP relies on unidirectional information exchanges: an AS can pick an inter-domain path only among the paths advertised by downstream ASes; it cannot indicate its routing preferences to the downstream domain. Consequently, BGP tuning is not sufficient to enable ISPs to take mutually beneficial routing decisions, as observed by Mahajan, Wetherall, and Anderson [122]. A promising solution to this problem would be to enhance the domains interactions for cooperative network engineering. *Specifically, in the book chapter [20], we have presented the emerging collaborative network management models as well as related technologies. In particular, we have argued that network operators will enhance the interactions among their networks to facilitate inter-domain traffic management and to improve the level of end-to-end QoS that they can provide to their customers.*

4.4 The MPLS-TE Architecture

MPLS [145] is an architecture that is based on *label switching* and that operates between the data-link and the network layer of the OSI model [92]. It was invented to enable fast switching operations thanks to a simplified header compared to IP. Nowadays, MPLS is mainly used for its traffic-engineering capabilities. In the present section, we introduce MPLS technologies and their interest for network operators.

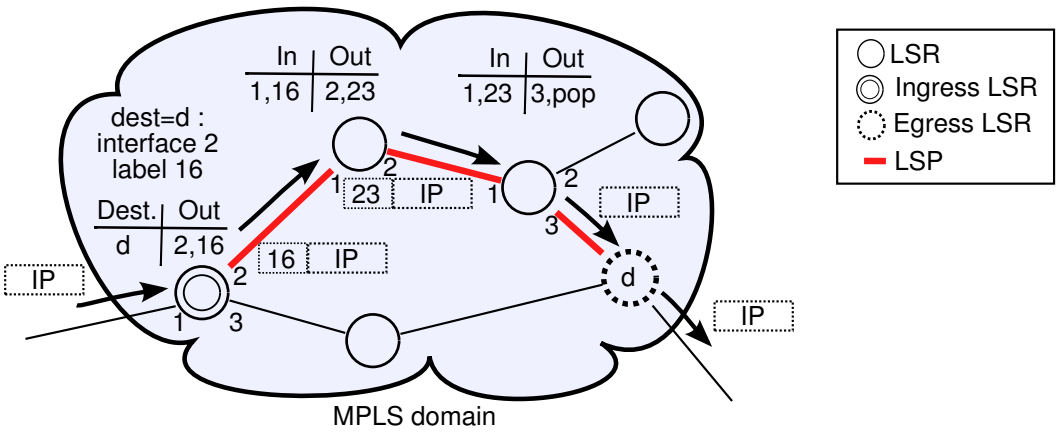

Figure 4.7: Classification into FECs and label switching in MPLS-TE networks

4.4.1 Label Switching Mechanisms

Figure 4.7 illustrates the main mechanisms of label switching in MPLS networks. A route inside an MPLS network is called a label switched path (LSP) and all MPLS-enabled routers of a domain are called label switch routers (LSRs).

The entry points of an MPLS domain, named *ingress* LSRs, prefix the packets with an MPLS header. To determine the contents of this header, ingress LSRs classify packets into sets of packets called *forwarding equivalence classes* (FECs). Typically, the FEC of a packet depends on its source and its destination address, and on the upper layer protocol indicated in the IP packet header. The FEC can also depend on the source and destination port, which are indicated in the header of the transport protocol (*e.g.,* TCP [30] or UDP [140]). Every traversed LSR must forward all the packets of a FEC along the same path.

The MPLS header contains a short path identifier, named *label*, which allows each LSR to forward the packets, based solely on the packet label. More precisely, every traversed LSR uses the label as an index into a table that specifies the outgoing interface, and a new label. The LSR replaces the old label with the new label, and forwards the packet on the outgoing interface to the next-hop [145]. Consequently, the initial label of every packet determines the route on which the packet is forwarded. This packet-forwarding mechanism enables network operators to setup alternate paths in addition to the usual IP routes. Finally, the MPLS header is removed with the *pop* operation before the packets exit the MPLS domain.

In Figure 4.7, an IP packet arrives in an ingress LSR, which determines that it must forward the packet to the egress router *d*. The ingress LSR searches the outgoing interface and the label corresponding to this destination in its MPLS switching table, the label forwarding information base (FIB). Then, it can determine the relevant outgoing interface (*2*) and the new label (*16*). Then, the ingress LSR forwards the labeled packet to the next LSR, which receives the packet with label *16* and on interface *1* and performs the label switching operations.

4.4.2 Traffic Engineering with MPLS

MPLS-TE allows ISPs to set up particular LSPs, named TE LSPs, that do not necessarily follow IP routing. As illustrated in Figure 4.8, the configuration of TE LSPs is interesting to forward packets that belong to certain FECs along specific routes with appropriate per-

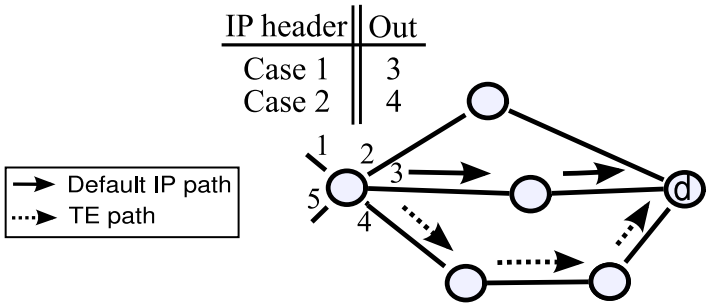

Figure 4.8: TE path and default path in a traffic-engineered network

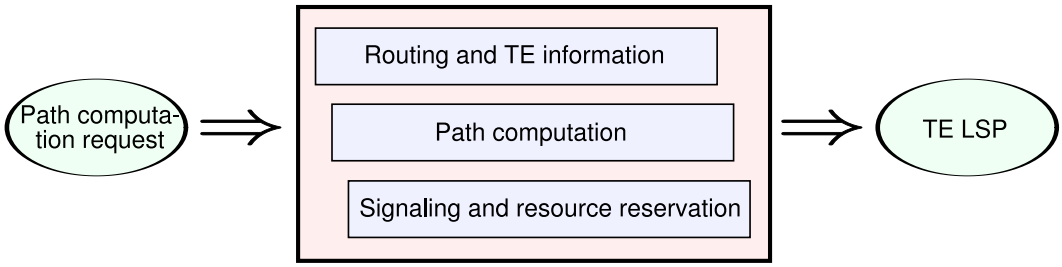

Figure 4.9: Main network functions involved in the computation and in the configuration of TE LSPs

formance, whereas the remainder of the traffic follows the usual IP routes. With TE LSPs, an ISP can route premium traffic on paths with high performance (*e.g.,* protected paths), whereas best-effort traffic follows the default IP routes.

Alternatively, an operator can route several traffic classes on a single LSP thanks to the DiffServ extensions for MPLS [71, 68, 70, 69]. The main mapping method of DiffServ to MPLS is called E-LSP (EXP-inferred LSP). When calculating the path for an E-LSP, the LSRs use a constrained shortest path first algorithm to ensure that the constraints are met for all the class types carried by the E-LSP. The path computation operations base on the topology and the link-state information (*e.g.,* available unreserved bandwidth) advertised by intra-domain routing protocols. The computed paths are signaled with RSVP: every LSR of the path verifies that the requested bandwidth for every class of service is actually available and establishes the path state.

Figure 4.9 depicts the main network functions that interact for configuring TE LSPs.

1. The first function is the *advertisement of routing and TE information.* In MPLS-TE networks, routing protocols have been enhanced to advertise extended link state information [102, 115]. For instance, the TE extension of OSPF [102] allows routers to represent and to disseminate network-state information. The disseminated data enable the routers to build an extended link state database, named traffic-engineering database (TED), which can include information about the unreserved bandwidth on the links.

2. The second function is the *path computation*, which uses TE information to determine paths with appropriate QoS and TE performance [128]. For example, the *constrained shortest path first* (CSPF) algorithm uses the information of the TED to run a shortest-path algorithm after pruning the links that violate the LSP constraints. This mechanism

enables CSPF to compute a shortest path fulfilling a set of constraints, like a minimum available bandwidth and a node exclusion list.

3. The third function is *the configuration* of the computed TE-LSPs. The most widely-deployed signaling protocol for configuring LSPs is RSVP-TE [8], the TE extension of RSVP.[10] During the signaling of a TE LSP with RSVP-TE, the source and destination LSRs of the TE LSP exchange signaling messages along the considered path to trigger the TE-LSP establishment in the routers. If a TE-LSP cannot be setup, because insufficient network resources are available, then, a crankback mechanism presented in Reference [65] enables trying alternative paths for the LSP.

RSVP-TE signaling messages typically carry an explicit route object (ERO) that describes the sequence of routers traversed by the TE LSP. This object can include *strict* hops that define the next-hop explicitly. In addition, it can include *loose* hops: components that rely on the routing table of an LSR or on path computation operations to determine the next-hop. The mechanism to resolve a loose hop is called *ERO expansion* [174]. Strict and loose hops provide two options for route selection in MPLS-TE networks: *hop-by-hop* routing and *explicit routing* [145, 8]. Hop-by-hop routing allows each LSR to choose independently the next-hop for each FEC. On the contrary, explicit routing allows a single LSR, typically the LSP ingress, to specify several or all the LSRs traversed by the LSP. The setup of TE LSPs relies on explicit routing: a path computation entity determines the path and triggers its signaling. It is possible to fix the loosely routed segments of an LSP so that the path used by the LSP does not change even when a better next hop becomes available at some LSR along the loosely routed segment. This mechanism is called *route pinning* and is used to avoid delay variations in case of routing changes [99].

4.4.3 Support of DiffServ in MPLS

Although MPLS supports several features that ISPs can use to engineer their traffic efficiently, MPLS must be complemented with QoS mechanisms for guaranteeing bandwidth, policing traffic flows and so on [157]. For example, MPLS-TE enables reserving bandwidth for an LSP with RSVP-TE [8], but the bandwidth reservation serves only for preventing link over-subscription by MPLS-TE paths and does not imply the actual reservation of resources in the routers. Thus, network operators use additional resource management mechanisms for the actual reservation of router resources. For example, they can combine the features of MPLS with the ones of DiffServ to provide desired service levels.

MPLS supports DiffServ traffic differentiation mechanisms based on classes of services. Ingress LSRs can associate packets with a particular PHB and a drop-precedence. In addition, they can use a specific LSP to forward certain packets. All the LSRs in the MPLS domain must agree on a mapping of the MPLS header information (label, traffic-class field) into well-defined PHBs and drop-precedence values. Furthermore, to avoid congestion, the ingress points must control the admission of new traffic flows for each PHB and for each destination. For instance, they can remark or drop packets that violate the traffic contracts.

The combination of MPLS traffic-engineering features with DiffServ QoS abilities provides a powerful traffic management scheme. For instance, network operators can use it to implement a VoIP service with a guaranteed bandwidth, latency, and jitter. In this scenario, ingress

[10] MPLS standardization bodies have stopped the development of the alternative signaling protocol, the constraint-based label distribution protocol (CR-LDP), in 2003.

LSRs uses the traffic-engineering database to compute explicit paths that fulfill the service specifications. The routing protocols must provision the TED with the amount of available bandwidth of each class type on every link. Then, the LSRs trigger the configuration of LSPs along these paths. Moreover, they mark the admitted VoIP packets so that these packets experience the expedited forwarding (EF) PHB [57]. The QoS requirements of the EF traffic are satisfied along the LSP thanks to appropriate queuing and dropping policies. A correct provisioning and admission control guarantee that the network provides the desired performance for every hop and that a suitable QoS level (*e.g.,* bandwidth, delay, jitter, and packet loss) is reached.

4.5 Comparison of IP and MPLS Traffic Engineering

The most common IP-based TE methods, for example adjustments of OSPF weights or BGP tuning, suffer from several limitations listed below. MPLS-TE provides enhanced traffic capabilities, which solve most of these limitations. However, MPLS is essentially used for intra-domain traffic engineering, even if recent technological advances, described in Chapter 5, enable setting up inter-domain LSPs. Thus, IP-based methods provide a major tool for engineering inter-domain traffic.

- The most obvious problem of IP-based traffic-engineering methods is that they are limited by the destination-based forwarding paradigm, which forbids routing differentiation based on QoS requirements. On the contrary, label switching in MPLS-TE networks allows routing critical traffic on paths with appropriate performance.

- IP routing typically uses relatively static information, which does not represent the current state of the network accurately, to compute the route for a destination. This feature minimizes the frequency of the routing information exchanges. However, it complicates network management tasks that depend on the state of the network, like load balancing. In MPLS-TE networks, TE-enabled routing protocols [102, 115] can advertise information about the available bandwidth on the links, which facilitates the adaptation of the paths to the actual traffic load.

- With IP routing, the path used to forward data packets can vary during a session, which can introduce out-of-order packet arrivals as well as unpredictable delay variations. The route pinning feature of MPLS-TE solves these problems.

- The MED attribute introduces a relationship between the weight setting of the IGP and the route choices of BGP, as described in Reference [127]. Therefore, interactions between intra- and inter-domain routing can introduce instability in the network: a re-optimization of the IGP weights can induce a change in inter-domain routing. Consequently, traffic management in backbone networks requires a complex coordination between intra- and inter-domain routing decisions. This problem could be avoided by robust traffic management methods, as Cerav-Erbas *et alii* [42] propose.

4.6 Conclusion

In this chapter, we have provided an overview of the QoS and the traffic management mechanisms for the Internet, with a particular focus on QoS routing and TE. Specifically, we have

introduced the most well-known QoS models for the Internet: IntServ and DiffServ. These technologies are limited (*e.g.,* IntServ scalability, DiffServ network dimensioning) but enable ISPs to create "QoS islands" that correspond to routing domains.

We have explained the importance of traffic engineering to operate a network efficiently. Then, we have detailed the traffic-engineering features of IP routing and of MPLS-TE. The presentation of these techniques enabled us to conclude that IP-based traffic-engineering mechanisms suffer from significant limitations. The MPLS-TE architecture solves most of these limitations and can be used in conjunction with DiffServ to provide a powerful traffic management framework.

Today, MPLS-TE is used mostly for traffic management inside a single domain, thus, it does not resolve the problem of providing end-to-end QoS. Therefore, in the next chapter, we present recent extensions of MPLS-TE that enable operators to manage inter-domain flows.

Key points of Chapter 4

- Routing and TE play an important role in traffic management and in QoS provisioning: most importantly, they enable avoiding congestion by using network resources efficiently. In addition, QoS routing and resource reservation make it possible to give strict performance guarantees.
- IP-based traffic engineering is widely used but suffers from significant limitations. MPLS provides enhanced TE capabilities and can be used in conjunction with DiffServ to provide a powerful QoS framework.
- Existing technologies enable ISPs to create "QoS islands" that correspond to routing domains. In the next chapter, we describe the advances toward interconnecting these QoS islands to guarantee end-to-end QoS.

CHAPTER 5

End-to-End Quality of Service

5.1 Introduction

The provision of end-to-end QoS is one of the main subjects studied in the present thesis. We think that end-to-end QoS would create several new business opportunities for ISPs, as explained in the introduction of the dissertation. Nevertheless, guarantees on end-to-end transit performance are difficult to provide for the traffic that traverses the network of several operators.

We detail relevant work on end-to-end QoS management and present some problems that hinder the deployment of QoS technologies for inter-carrier traffic management. We focus our presentation on two aspects: the negotiation of inter-provider QoS (Section 5.2.1) and the enforcement of inter-provider SLAs (Section 5.2.2). MPLS-based inter-domain traffic engineering is a promising method to configure inter-domain paths with adequate performance. Therefore, in Section 5.3, we describe the recent advances of path computation and signaling technologies in MPLS networks.

5.2 Related Work on End-to-End QoS

End-users experience a level of QoS that depends on the performance of the end-to-end path followed by service flows. Thus, QoS must be enforced on all the traversed network segments (access, aggregation, and core), so that ISPs can successfully deploy applications with stringent QoS requirements, like real-time multimedia services. In addition, the end-to-end QoS performance experienced by inter-domain flows depends on the way the packets are processed inside each crossed domain. Consequently, efficient traffic management is required inside every traversed domain to protect critical services, as well as to control network costs.

Numerous studies propose solutions for QoS routing and for TE *inside a single domain*, based on enhancements of IP routing or on the MPLS architecture. Wang *et alii* [178] present a good overview of these solutions. Providing QoS for *inter-domain* flows is a more complex problem. In particular, the absence of coordination among the network operators complicates the provision of end-to-end performance guarantees. As ISPs employ various traffic management policies, the main problematic is to guarantee that a packet receives the right treatment in every traversed domain. Outside its own network, an ISP cannot easily ensure that traffic receives an appropriate level of QoS. Therefore, inter-domain traffic flows require mechanisms

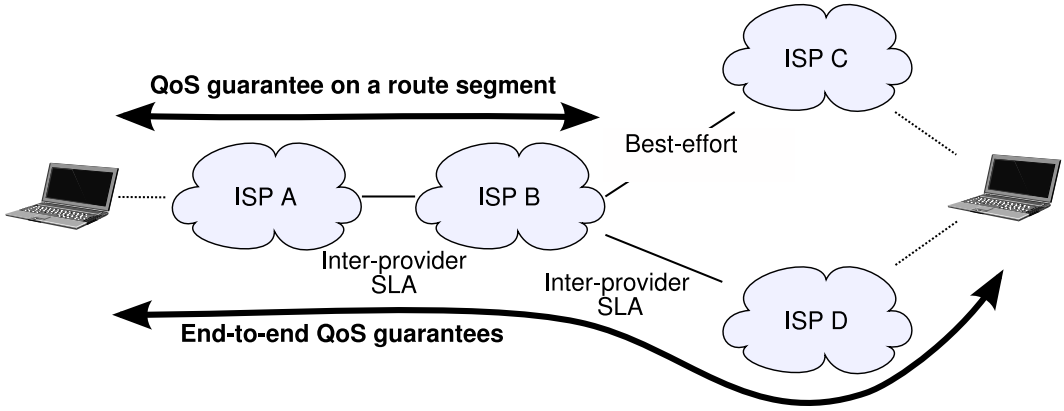

Figure 5.1: Bilateral inter-provider SLAs for end-to-end QoS

for QoS management that involve several domains. The present thesis contributes to the development of such mechanisms.

5.2.1 Negotiation of Inter-Provider QoS

Various technical perspectives are proposed for the future Internet to provide performance guarantees across AS boundaries. From the contractual point of view, QoS is guaranteed by establishing relevant service level agreements (SLAs) with neighboring ASes. An SLA is a contract that describes the expected performance for transit traffic and the agreed price for the transit service. SLAs enable the provision of end-to-end service-level guarantees through chains of bilateral SLAs between adjacent providers.

Providing QoS through bilateral SLAs requires that networks with numerous interconnections manage many SLAs. In addition, Mahajan, Wetherall, and Anderson [122] note that most ISPs appear reluctant to provide end-to-end guarantees for traffic transiting through domains belonging to other ISPs. Thus, the scope of the possible performance guarantees that can be obtained through bilateral SLAs is limited.

The example in Figure 5.1 illustrates the problem of providing end-to-end QoS through a chain of SLAs between adjacent providers. A computer accesses a video content on a server of a service provider. The video traffic between the client and the video server successively crosses the networks of three providers A, B, and D. If A and B, as well as B and D are bound by appropriate SLAs, then it is possible to provide end-to-end QoS guarantees for the considered traffic. However, if two successive providers, like B and C, are not bound by an appropriate SLA, then it is not possible to provide any end-to-end QoS guarantee.

Several issues must be solved to facilitate the establishment of a chain of providers' agreements to provide a specific end-to-end service. We describe two of these issues in the present paragraph.

1. Today, the negotiation of peering agreements and related SLAs between two providers requires human operations. Thus, the configuration of SLA chains for end-to-end QoS takes time.

2. The determination of a chain of providers to reach a specific destination is a second problem. When several provider sequences are available and offer varied transit perfor-

mance for varied prices, the discovery of the preferred sequences by the stub domain is complicated.

There are several initiatives to solve these two problems. For example, the IPSphere forum (IPSF) describes an enhanced commercial framework (*business layer*) that supports flexible business relationships for IP service creation. This business layer enables the provision of services through the networks of multiple operators [141]. Several international projects try to address similar problems. For instance, the IST project OpenNet [126] investigates solutions to the lack of predictable quality of service, when packets have to cross domains operated by different providers. The fact that several industry forums and research projects investigate the provision of inter-domain QoS reveals the strategic importance of this topic for network operators.

The mechanisms to compute a sequence of providers, and thus, a sequence of transit domains, are out of the scope of the present thesis and involve complex operations. For example, in [25, 26], we have evaluated network exploration mechanisms for discovering appropriate domain sequences for QoS flows. In particular, we have considered an example application of a simple exploration algorithm, including complexity reduction mechanisms, to simultaneously minimize the traffic forwarding cost, guarantee a minimum bandwidth and optimize an additive metric related to load balancing. A simulation study has demonstrated that our complexity reduction mechanisms were efficient in the considered setting. In the same references, we have proposed that the path computation entities in a domain be aware of the economical traffic-forwarding cost (price) of the inter-domain links connected to this domain. We have shown that this additional knowledge allows the implementation of interesting economical strategies. For that, we have implemented a detailed example.

5.2.2 Enforcement of Inter-Provider SLAs

The enforcement of the contracted inter-provider SLAs relies on various traffic management mechanisms. In particular, the ability to compute constrained paths that provide the QoS guarantees required by a service and that satisfy network management constraints is fundamental.

Numerous publications have proposed to add QoS capabilities to IP inter-domain routing based on the border gateway protocol (BGP). For example, several Internet drafts, such as [28, 16, 105, 153], have been proposed at IETF in the inter-domain routing (IDR) working group. In particular, Boucadair [28] proposes to advertise QoS-enabled reachability information between service providers to enable the use of an enhanced route selection process that takes the QoS performance of the paths into account. Furthermore, several European projects, such as AQUILA [106], EuQoS [186, 73, 125, 61], TEQUILA, MESCAL [88], and AGAVE [29], have studied the problem of inter-domain QoS management as well as possible extensions of BGP QoS capabilities.

Modifying BGP is difficult because this protocol is widely deployed and its behavior is complex. In particular, the scalability, the stability, and the convergence time of BGP must not be affected by the protocol modifications [90]. Consequently, to the best of our knowledge, the QoS extensions of BGP have not been widely deployed yet.

In addition to the QoS extensions of BGP, a promising approach for inter-domain QoS consists of deploying connection-oriented mechanisms based on the MPLS architecture at the inter-domain level. For example, Barth *et alii* [14] describe mechanisms for configuring QoS inter-domain paths inside an alliance of ASes. They assume that the networks can reserve

resources for these paths, for instance, thanks to MPLS features. Inter-domain deployments of MPLS allow network operators to setup inter-domain TE LSPs with guaranteed performance, and thus, to provide end-to-end QoS guarantees for the service flows routed on such TE LSPs. However, providing LSPs with performance guarantees across domain boundaries brings novel requirements, described in [149, 189], on the MPLS-TE architecture. In this context, the present thesis describes several contributions for the computation of inter-domain paths subject to multiple constraints in MPLS-TE networks.

5.3 Extensions of MPLS for Inter-Domain Resource Provisioning

The present section describes the enhancement of MPLS-TE to fulfill the requirements for *inter-domain* traffic engineering [149, 189]. They concern the main building blocks of the MPLS-TE architecture, namely (1) TE-enabled routing protocols, (2) path computation algorithms, and (3) path signaling techniques.

5.3.1 TE-Enabled Routing Protocols

Scalability is a critical concern in inter-domain routing because of the size of the Internet. In addition, information about the topology and the state of every domain is typically considered as confidential. As a result, most existing techniques for establishing inter-domain LSPs do not require the advertisement of TE information at inter-domain level. This information remains confined inside every domain and the path computation procedures are distributed to take the TE information of every domain into account.

5.3.2 Path Computation Procedures

Two main techniques for computing *inter-domain* TE-LSPs have been proposed: the *per-domain method* [173] and the *path computation element (PCE)-based technique* [66]. They permit the computation of constrained inter-domain paths in a distributed manner to overcome the visibility limitations introduced by the presence of multiple domains.

The Per-Domain Method

The per-domain method [173] relies on the juxtaposition of paths computed by each domain of a predetermined domain sequence. Every domain computes a path segment to reach the next domain, without using any information shared by other domains. The first domain computes the first segment to reach an exit border node (BN). Then, the second one computes a path segment to the next exit BN, and so on until the destination is reached. The concatenation of the computed segments provides an end-to-end path. As the segments are computed by entities with local visibility, and without collaboration between the domains, the concatenation does not necessarily lead to an optimal end-to-end path. To determine the next-hop BNs, the domains can use an auto-discovery mechanism based, for example, on routing information [173].

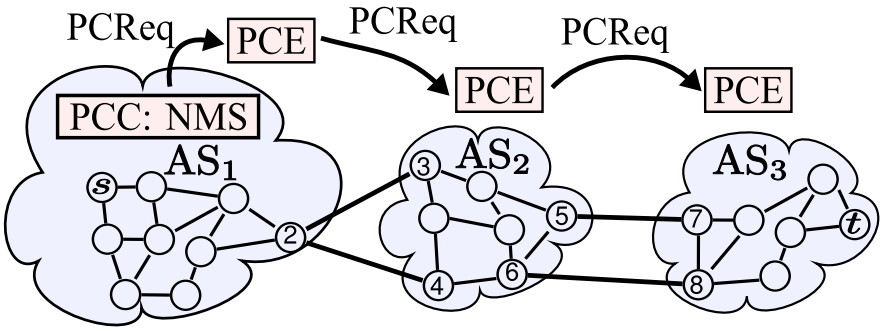

Figure 5.2: Forwarding of a path computation request from the PCC to a PCE of the destination domain

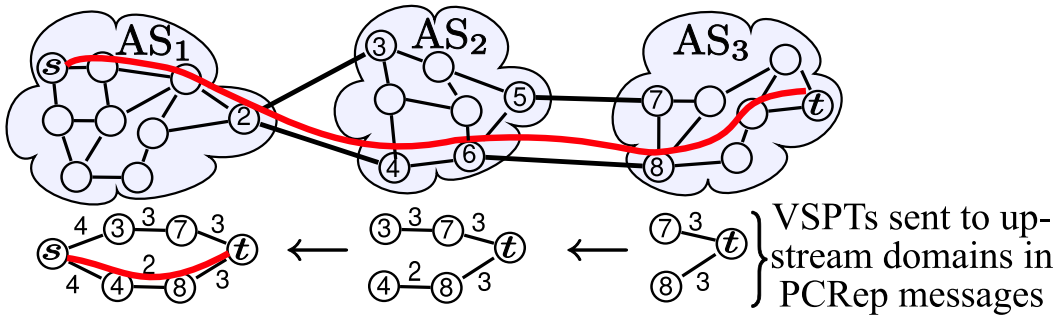

Figure 5.3: Operations of BRPC for computing a path from s to t with the minimum number of traversed links along the sequence AS_1-AS_2-AS_3

The Path Computation Element Techniques

A PCE is an entity that receives and answers requests from path computation clients (PCCs). Every path computation request specifies a source, a target, and a set of path properties. The PCE communication protocol (PCEP) conveys the path computation requests and replies exchanged during the path computation operations [175].

According to the PCE-based path computation technique [66], a PCE that receives a request and that has enough information to answer this request can compute an end-to-end path. Otherwise, the PCE collaborates with other PCEs, potentially responsible for other domains, to determine intermediate loose hops, or a full explicit path.

Reference [176] specifies the backward recursive PCE-based computation (BRPC) procedure. BRPC is a collaboration method to compute constrained inter-domain paths across a specific sequence of traversed domains in PCE-enabled networks. The domain sequence is either administratively pre-determined or discovered by some means that is outside of the scope of Reference [176].

Figure 5.2 and Figure 5.3 illustrate the operations of the BRPC procedure to compute a path with the minimum number of traversed links from a source node s to a target node t. As illustrated in Figure 5.2, the first stage of the BRPC procedure consists in forwarding the path computation request from the PCC to a PCE of the destination domain [7, 175, 176]. The request can be carried in a PCReq message of PCEP [175]. The actual path computation can start after a PCE the destination domain has received the path computation request.

We call *entry BN* every node that connects a domain to the previous domain in the considered sequence of traversed domains. A PCE of each domain computes a shortest-path tree whose leaves are the entry BNs of the domain and whose root is the target of the considered path computation request (Figure 5.3). This *virtual shortest-path tree* (VSPT) is forwarded to the previous domain, which uses it to compute a similar VSPT from its own entry BNs to the target. The process is repeated until a PCE of the source domain computes an end-to-end path. BRPC allows finding constrained paths that optimize a single additive metric, along a given AS-path, while avoiding signaling crankbacks [65, 55].

In Figure 5.3, the first computed VSPT contains a path that starts from node 7 and that traverses three links. In addition, the VSPT includes another path, from node 8 to t, with the same length. AS_2 uses the information about these paths to compute shortest paths from its entry BNs to t. It finds a path that traverses six links to reach t from node 3 and a path that traverses five links to reach t from node 4. Finally, AS_1 uses the information about the paths advertised by AS_2 to determine the shortest end-to-end path from s to t.

The per-domain method and the BRPC procedure focus on the case where a domain sequence is predetermined before the actual router-level path is computed. This division between the determination of inter-domain and intra-domain level paths is commonly adopted in the Internet (see Section 4.3.1), to simplify the path computation problems and to reflect the policy-based routing decisions inside every AS. However, it is possible that a path satisfying the constraints of a request cannot be found, because all solutions follow a domain sequence that differs from the considered domain sequence.

5.3.3 Inter-Domain Path Signaling

IETF has specified mechanisms to support the establishment and the maintenance of LSPs that cross domain boundaries [64] and has extended RSVP-TE to support the signaling of inter-domain LSPs [8, 108, 9].

Reference [67] identifies three main options for signaling inter-domain TE LSPs. The first one involves carrying the inter-domain TE LSP within another LSP inside every traversed domain and is named *LSP nesting* [108]. One advantage of this approach is that each intra-domain LSP is fully managed by the domain owner and may nest multiple inter-domain LSPs. The second technique uses a single signaling exchange to establish an end-to-end LSP that is named *contiguous inter-domain LSP*. Finally, with the third method, the domains establish separate LSPs and "stitch" these LSPs together in the data plane to form a single end-to-end LSP. Therefore, this method is called *LSP stitching* [9]. These three types of signaling can also be mixed to form hybrid-signaling methods.

5.4 Conclusion

The provision of end-to-end QoS is a well-identified requirement for enabling the provision of value-added services across the Internet. Thus, it receives much attention from both industry forums and collaborative research projects. The main work directions try to facilitate the establishment and the selection of chains of transit providers that guarantee a specific transit performance for a particular price. These subjects are intrinsically complex, because they involve both business and technical considerations.

Some progress toward the provision of end-to-end QoS for inter-domain flows can be noticed, for example, the IPSphere forum provides interesting contributions. From the technical point of view, offering end-to-end guarantees for flows that traverse several domains administrated by varied ISPs is difficult because of the autonomy of every ISP. In this context, recent advances on the configuration of constrained inter-domain paths in MPLS networks enable network operators to setup inter-domain paths with guaranteed performance and to provision dynamically resources for inter-domain flows. Therefore, they provide novel possibilities for inter-domain QoS aware routing and TE.

The present chapter concludes the background part of the thesis. In the next part, we describe our main contributions toward the provision of end-to-end QoS. Specifically, in Chapter 6, we present a study of traffic-engineering and dimensioning-evaluation mechanisms for DiffServ-MPLS networks.

Key points of Chapter 5

- End-to-end QoS and inter-domain traffic management are important topics: they have been investigated in numerous research projects and in several industry organizations. The introduction of end-to-end QoS management may have significant consequences on operator's business models.
- The PCE architecture provides a powerful framework for inter-domain traffic engineering and could be an important element of future inter-carrier QoS solutions.

Part II

Contributions

CHAPTER 6

Bandwidth Allocation in DiffServ-TE Networks

6.1 Introduction

Network operators face various problems to implement specific service levels. For example, in DiffServ networks, resources are typically not reserved dynamically at flow level but rather provisioned in advance at class-of-service level. The amount of resources that is allocated for every CoS is based on traffic expectations and must be relevant for the actual network conditions to ensure appropriate service differentiation. We illustrate this problem with a study that we have presented in Reference [24], in the beginning of the thesis.

DiffServ complexity lies in the dimensioning of the network and in the configuration of the routers [10]. Typically, the enforcement of a specific service level relies on the appropriate configuration of both edge routers (admission control, marking, policing and shaping, *etc*) and core routers (scheduling, queue management, resource reservations, *etc*) [157]. Network operators need tools to determine appropriate configuration parameters for core-network routers, knowing the service level agreements that they have contracted with every individual customer at the edge of their network.

The objective of the present chapter is to propose a simple method to compute the routing and the bandwidth allocation on the links of a DiffServ-TE network to satisfy a given set of SLAs. In addition, the chapter presents several results that enable us to evaluate the quality of the routing and bandwidth allocation returned by our model. Our model does not directly address the network configuration issues. It proposes a routing and bandwidth allocation configuration in the considered network and assumes that the network operator can parameter its routers to reach this configuration. The evaluation parameters included in our model enable network operators to verify that their network has a sufficient capacity to fulfill the SLAs that they have sold. Moreover, these parameters allow the identification of unused and of congested links, and thus, help operators to forecast network evolutions.

The remainder of the present chapter is structured as follows. Section 6.2 presents DiffServ mechanisms and the treatment associated with the different classes of services. In Section 6.3, we introduce our bandwidth allocation algorithm, and the simulation environment in which we implement our model. We have studied many performance criteria to evaluate jointly the routing and the bandwidth allocation proposed by our model for an operator's network. We present the simulation results at the end of the chapter, in Section 6.4.

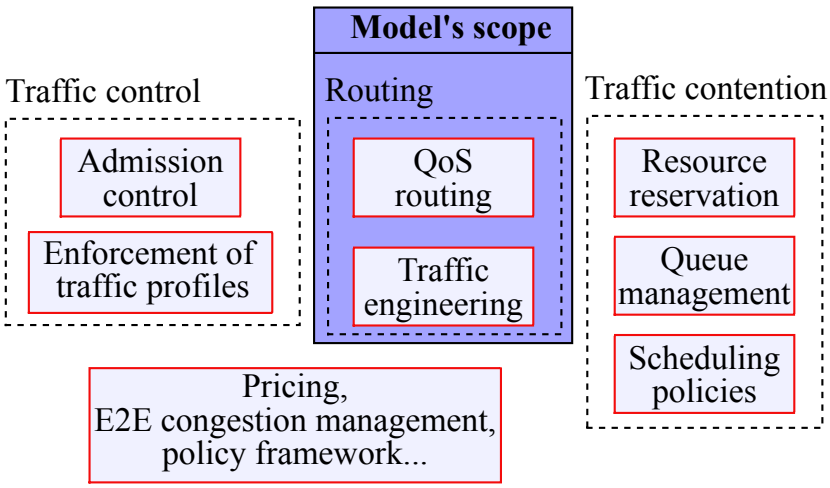

Figure 6.1: Building blocks for QoS management in DiffServ traffic-engineered networks

6.2 Traffic Engineering for DiffServ-Aware Networks

6.2.1 Building Blocks for QoS Management

QoS frameworks involve many mechanisms, which operate a various levels, to provide the desired service levels. Soldatos *et alii* [157] provide an interesting classification of these mechanisms into building blocks for traffic management at packet, flow, and network level. Figure 6.1 represents the main building blocks that operate in DiffServ traffic-engineered networks. We describe them in the following paragraphs.

We first describe the "traffic contention" block in Figure 6.1. In packet switched networks, the flows contend for the network resources: the incoming packets rate at a router can exceed the outgoing capacity of the router. Therefore, routers use buffers (queues) to store temporarily the packets that they cannot serve immediately, in case of burst arrivals. The routers use queue management functions and scheduling policies to share the available bandwidth among the admitted traffic flows and to differentiate the per-hop behaviors for every CoS. The configuration of these functions can be triggered through resource reservation techniques such as ones described in Section 4.4.3.

Figure 6.1 includes a traffic control block that contains the admission control an traffic profile enforcement function. Connection admission control (CAC) is a procedure that either accepts or rejects incoming flows, depending on the current network usage, to avoid congestion. CAC accepts a flow only if the network has sufficient resources to provide the required QoS level for the new flow and to maintain the QoS agreed for previously admitted flows. CAC decisions depend on the user subscription and on its rights to use the network resources. SLAs and SLSs specify a traffic profile that the user must respect to receive the agreed level of service. Traffic profiles can be enforced by the operator, at the edge of its network, thanks to various operations called *policing* (dropping excess traffic), *shaping* (delaying some packets in case of burst arrivals), and *marking* (tagging excess traffic so that, in case of congestion, core network routers drop this traffic in priority).

CAC operates at the edge of the network, to enforce traffic profiles and to avoid congestion. Additional mechanisms are required in edge and core network routers to use the network resources efficiently (*e.g.,* balance the traffic load on the network links). The admitted traffic

must be routed along paths with appropriate performance levels and must be shared among the available paths to avoid congesting bottleneck links. The routing block in Figure 6.1 provides suitable functions to do these tasks.

To complete the presentation of the QoS mechanisms, we have to mention the effect of pricing on the traffic patterns, the importance of end-to-end congestion management techniques (*e.g.,* TCP configuration [5], ECN, PCN [34]) to adapt the behavior of traffic sources to the network condition, and the role of policy management frameworks to reach a consistent network configuration.

The objective of our model is to propose an appropriate routing and bandwidth allocation configuration to support a set of service level agreements. Thus, the model focuses on the routing and traffic engineering building blocks. However, to evaluate the model's performance on a concrete example, we need to model the mechanisms of other building blocks, as explained in subsequent sections.

6.2.2 DiffServ Per-Hop Behaviors

The main function of traffic differentiation is to guarantee quality of service for designated flows during congestion phases, *i.e.* when the network is overloaded. DiffServ defines three behaviors: expedited forwarding (EF) [85], assured forwarding (AF) [57] and best-effort (BE) [132]. EF specifies a class of service for flows that require to be protected from losses and delays. AF provides four classes of service for elastic flows. BE is the class describing the actual functioning of the Internet, *i.e.* without any guarantee. DiffServ implements a specific per-hop behavior (PHB) for each class of service.

The EF PHB: The dimensioning of the network and the enforcement of the traffic profiles are the essential mechanisms to ensure the efficiency of the quality of service differentiation. The strong guarantees associated with the EF class of service rely on bandwidth over-provisioning: operators perform admission control and allocate only a small amount of the reserved bandwidth to transport EF flows. As a result, the queues used for EF packets are likely to be almost empty most of the time. Hence, packet losses and queuing delays are small.

The AF PHB: Four classes of service, namely AF1, AF2, AF3, and AF4, constitute the AF behavior and are associated with specific network configurations (queue management, scheduling policy, *etc*). AF traffic is well suited for elastic flows since it does not give any guarantee on the packet transit delays in the network. A characteristic of AF flows is the possibility given by the operator to transport more traffic than specified in the SLA, as long as the additional traffic does not disturb the network. In order to reach this objective, each class implements three drop-precedence levels. In the present study, we consider only a two drop-precedence level paradigm: this means that we distinguish *in-profile* traffic from *out-of-profile* traffic. During congestion periods or when traffic load increases, out-of-profile packets are dropped to improve the network condition.

The BE PHB: This PHB describes the usual behavior of the Internet: packets are sent without any guarantee in terms of delay or losses. During congestion periods, BE traffic is affected before AF or EF traffic.

Our model represents a realistic network environment: it includes all the standardized DiffServ classes of service. This feature is interesting to evaluate various parameters that typically cannot be studied with previous modelings (*e.g.,* [86, 151]), which considered only some of the DiffServ classes of service. For example, our model enables us to investigate the

interactions of AF out-of-profile traffic with BE, to evaluate the capacity of a network to support additionnal traffic.

6.3 Presentation of the Model

6.3.1 Objectives and Assumptions

The contribution of the present chapter is to propose a simple routing and bandwidth allocation model for DiffServ-TE network, considering all standardized classes of service. This model is valuable because (1) network operators typically use home-made tools or commercial software, based on non-publicly available models, to compute optimal routing and bandwidth allocations and (2) the available models typically do not include all the standardized classes of service.

In practice, the purpose of our model is to determine the amount of traffic that must be allocated on every link for every demand of every class of service: this is a routing and bandwidth allocation problem. However, to evaluate our model with credible hypotheses, we need to generate traffic demands for the studied classes of service and to model admission control in addition to routing and bandwidth allocation operations. The following section describes our modeling of admission control, to get realistic demands as input for our routing and bandwidth allocation algorithm.

In the scope of our routing and bandwidth allocation model, we assume that the considered network operator knows the amount of traffic of every class of traffic that its network must support. This information is commonly named a *traffic matrix* (TM). The assumption that a traffic matrix is available is commonly done by network operators when they perform offline traffic engineering and dimensioning operations. Determining appropriate estimations of the traffic matrices is a difficult task. It would be too long to describe the available traffic matrix estimation methods in the scope of this chapter; therefore, we refer the interested readers to [129, 107] for a good introduction to related subjects.

In the present study, we suppose that the network operator can parameterize scheduling and queue management in intermediate nodes in accordance with the bandwidth allocation proposed by our model. As explained in the chapter's introduction, this task is not straightforward but is out of the scope of the present chapter. This assumption enables us to consider that the network reaches the correct dimensioning defined by the network administrator. Hence, we can consider bandwidth allocation volumes without detailing how the bandwidth is allocated, in practice, in the network equipments.

6.3.2 Admission Control

To test our routing and bandwidth allocation model on a concrete example, we need to define a scenario that includes a network topology, the capacity of the links, and the expected traffic profiles. The traffic profiles depend strongly on the SLAs sold by the network operator to its customers and on the admission control policies of the operator. Therefore, we generate random traffic demands and include in our model the simple admission control policy described below.

We consider a network in which we classify network nodes into *transit* and *stub* nodes. Stub nodes are the final origins and destinations of the demands. We denote the total number of nodes in the network as N_{n}, and the number of stub nodes as N_{sn}. We assume that a traffic

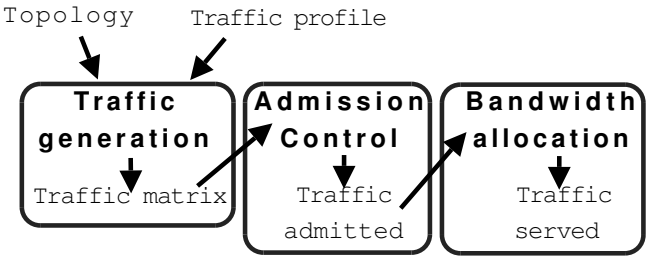

Figure 6.2: Operations simulated to evaluate our bandwidth allocation model

matrix is known for this network and for every class of service: it gives stub-node pairs (origin, destination) and a traffic volume for every node pair. Figure 6.2 depicts the operations of our model: the admission control process takes the traffic matrices as input and returns the admitted flows.

In our admission control model, we have tried to divide the available network resources *fairly* among the flows of a given class of service type. A commonly used definition of the fairness is named *max-min fairness* and is defined below [143].

Definition (Max-min Fairness)**.** A feasible allocation of a resource is "max-min fair" if and only if an increase of any resource allocation within the domain of feasible allocations must be at the cost of a decrease of some already smaller resource allocation.

The admission-control mechanism implemented in the model provides a fast two-iteration approximation of the max-min fairness. The operator specifies the bandwidth that each class of service c can use to inject traffic into the network. In our implementation of the bandwidht allocation model, this bandwidth is a share S_c of the minimum outgoing capacity $\min_{\text{out}}$ among the stub nodes. In the first iteration of the admission-control process, all demands d that originate from the same node i and belong to the same class of service c receive the same bandwidth share, which is presented in Equation (6.1). In this equation, S_c represents the bandwidth allocated in the router for the considered class of service c and B_d is the requested bandwidth for the demand d. Stated differently, all demands of a class receive the same bandwidth share. The link resources might not be fully used after the first allocation step, because some demands can be lower than the allocated bandwidth share. Therefore, during the second iteration, the remaining allocable bandwidth B_r for the considered class of service is shared among the demands that have not been completely granted yet (6.2). This method approximates the max-min fairness, because the smallest demands are satisfied first.

$$\min\left(B_d, \frac{\min_{\text{out}} S_c + 0}{N_{\text{sn}}}\right) \tag{6.1}$$

$$\min\left(B_d, \frac{\min_{\text{out}} S_c + B_r}{N_{\text{sn}}}\right) \tag{6.2}$$

As represented in Figure 6.3, the admission control policy is strict for the EF class: only a specific amount of EF traffic can be admitted into the network and the excess traffic is dropped. In addition, admitted flows marked with the EF tag are highly protected thanks to an over-provisioning margin: in the model, we apply a ratio of 10 Megabytes reserved for 1 Megabyte requested. The admission policy is less strict for AF traffic: excess traffic is

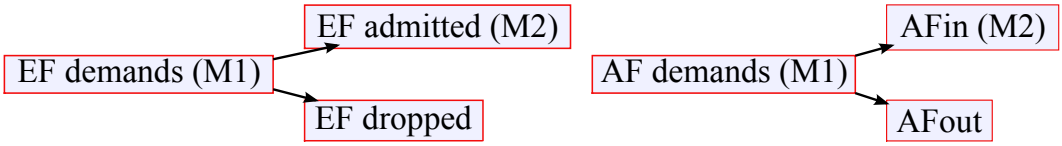

Figure 6.3: Admission control operations for EF and AF to obtain an admitted traffic matrix M2 given a random traffic matrix M1

admitted and remarked. The admission controller sets the drop precedence level to IN for the in-profile traffic or OUT for the out-profile traffic.

6.3.3 Bandwidth Allocation Model

Overview of the Bandwidth Allocation Operations

Our model provides a simple way to allocate bandwidth for given demands with an off-line optimization to guarantee quality of service. The bandwidth allocation is performed sequentially and attributes bandwidth according to the priorities of the flows.

1. EF flows are served first, according to the DiffServ philosophy. EF traffic is highly protected, and thus, if the network dimensioning is correct, the bandwidth requested by EF demands and accepted by the admission control is allocated.

2. The residual link capacities after bandwidth allocation for EF traffic are considered for serving AF in-profile traffic, in a second phase. The residual link capacities being smaller than the initial link capacities, capacity constraints can forbid the usage of certain paths. Thus, the path diversity can be smaller for AF flows than for EF flows and when the network is congested the paths chosen for AF flows are less good than the ones selected for EF traffic.

3. The third phase is the allocation for BE demands.

4. We consider that out-of-profile AF traffic is treated as *lower than best-effort*. Therefore, it is served last and only if the network provides sufficient resources.

Optimal bandwidth allocation for each class of service is computed using a linear program. This leads to solve several related multi-commodity flow problems [138]:

- a maximum concurrent flow problem for AF and BE classes of service, with a minimum guaranteed service for AF and without penalizing BE,
- a maximum multi-commodity flow for out-of-profile traffic from AF classes.

The idea of performing the bandwidth allocation hierarchically was proposed in [130], where Mitra *et alii* propose a traffic-engineering model for QoS and BE flows with a route-based formulation. We think that it is a good idea because it enables differentiating the performance of the classes of service in a simple manner.

Our model has an interesting property compared to previous work: it provides a global solution that addresses the flow allocation problem for all the traffic classes. On the contrary,

most papers on DiffServ network dimensioning concentrate on particular classes of services and do not consider all the standardized traffic classes. For example, Reference [86] addresses on-line EF flow acceptance. In addition, [151] focuses on the effect of EF flows on BE traffic, without considering AF and does not observe the effect of AF out-of-profile flows over best-effort traffic.

Linear Program for Bandwidth Allocation

The network is represented as a directed graph $G = (V, E)$, where V is the set of network nodes and E is the set of edges that represent the network links. Each edge e is characterized by a capacity c_e and we denote as C1 the matrix of all the initial edge capacities. The traffic matrix M1 provides the set of initial resource demands for a given class of service. The application of the admission control function on M1 provides a second traffic matrix M2 that depicts the admitted demands for the considered class of service, as depicted in Figure 6.3. The matrix M2 gives the set of commodities that we consider in our bandwidth allocation problem. In addition, it associates to each commodity a source $s(d)$, a sink $t(d)$ and a demand d.[1] We denote the set of the demands for the considered class of service as $\mathcal{D}$.

Given a traffic matrix M2 that represents the flows admitted into the network for a specific CoS, the bandwidth allocation problem is the following: *which amount of bandwidth should be allocated on which link for which demand, in order to maximize the flow admitted*. We solve successively this optimization problem for every CoS. We formulate the problem mathematically, with the following notations. The flow admitted into the network for a demand d is denoted as F_d. The objective function f of our optimization problem is the total traffic injected in the network after admission control and successfully routed.

$$\text{Maximize } f = \sum_{d \in \mathcal{D}} F_d \tag{6.3}$$

Two formulations can be adopted to model the considered problem: *path-link* or *node-link* based [138]. The path-link formulation considers the flow routed on a path for a given demand. It requires the knowledge of the set of all possible paths for each demand d, which may be exponentially large. On the contrary, the node-link formulation leads to a polynomial problem. Therefore, we use the node-link formulation: the problem variables are the flow values $R(i, j, d)$ that correspond to every demand d on link (i, j), and the flows F_d.

All variables are positive and the admitted flow F_d related to a demand d cannot exceed the bandwidth B_d requested for d. In addition to these $N_\text{v} + N_\text{d}$ lower and upper bound constraints, the problem variables are subject to three additional types of constraints: (1) the definition of F_d, (2) the flow conservation law, and (3) the link capacity constraint. We represent the network nodes by integer indices from one to N_n and we denote as $\mathcal{N} = [1..N_\text{n}]$ the set of the indices that correspond to the nodes in V.

$$\forall d \in \mathcal{D}, F_d = \sum_{j=1}^{N_\text{n}} R(s(d), j, d) \tag{6.4}$$

$$\forall d \in \mathcal{D}, \forall k \in \mathcal{N}, \sum_{j=1}^{N_\text{n}} R(k, j, d) - \sum_{i=1}^{N_\text{n}} R(i, k, d) = \begin{cases} F_d & \text{,k= } s(d) \\ -F_d & \text{,k= } t(d) \\ 0 & \text{, else.} \end{cases} \tag{6.5}$$

[1] We use indifferently the notation d to refer to a demand or to its index.

$$\forall e \in E, e = (i,j), \sum_{d \in \mathcal{D}} R(i,j,d) \leq c_e \tag{6.6}$$

The complexity of linear programming problems is closely related to the number of variables and to the number of constraints of the problem. The problem formulation in the equations 6.3–6.6 leads to a total number of variables N_{v} that is related to the number of demands N_{d} and to the number N_{l} of links in the network, as depicted in 6.7. In the worst case, there is a demand for every source-destination pair and the network has a full-mesh topology: in this scenario, the number of variables is bounded by the expression 6.8.

$$N_{\mathrm{v}} \leq N_{\mathrm{d}}(1 + N_{\mathrm{l}}) \tag{6.7}$$

$$N_{\mathrm{v}} \leq {N_{\mathrm{sn}}}^2(1 + {N_{\mathrm{n}}}^2) \tag{6.8}$$

The number N_{C} of constraints considered in the bandwidth allocation problem depends on the number of demands and of network links. In the worst case, it is bounded by Equation (6.9).

$$N_{\mathrm{C}} \leq {N_{\mathrm{sn}}}^2(1 + {N_{\mathrm{n}}}^2) \tag{6.9}$$

6.3.4 Bandwidth Allocation and Routing for AF Classes

Our model shares the bandwidth among the AF in-profile traffic classes (AF1, AF2, AF3, AF4), proportionally to their SLA, during congestion periods. For example, if the amount of AF1 traffic admitted in the network is two times the amount of AF2, then the amount of AF1 traffic routed will be approximately two times the quantity of AF2 traffic routed. For this purpose, the routing and bandwidth allocation linear program presented in Section 6.3.3 is solved for the traffic matrix corresponding to the global amount of AF in-profile traffic for all the classes. We denote the solutions to this problem as $F_{d,\mathrm{AF}}$ and $R_{\mathrm{AF}}(i,j,d)$, the bandwidth share for AF traffic as B_{AF} and the bandwidth share for AF1 as B_{AF1}. Then, the amount of resources allocated globally for all AF traffic is split among the AF classes: the solution variables (*e.g.,* R_{AF1}) corresponding to each AF class (*e.g.,* AF1) are computed using the equations 6.10 and 6.4.

$$R_{\mathrm{AF1}}(i,j,d) = \frac{B_{\mathrm{AF1}}}{B_{\mathrm{AF}}} R_{\mathrm{AF}}(i,j,d) \tag{6.10}$$

6.4 Results of Numerical Simulations

6.4.1 Simulator Overview

We have implemented our model with Matlab and with a linear programming solver named lp_solve, to study the dimensioning of an example topology. The purpose of this study is to show on a simple example that our bandwidth allocation model enables differentiating the network performance for the classes of service. We consider a fictitious ten-node topology (see Figure 6.4) with link capacity matrix C_1. In Figure 6.4 all links are made of two directed links with the indicated capacity (*e.g.,* 155Mb/s). Stub nodes (1–6) are the final sources and terminations of the demands, and data flows can transit through intermediate nodes (7–10).

The simulator implements the operations (traffic matrix generation, admission control, and bandwidth allocation) presented in Figure 6.2 according to the algorithm in Figure 6.5. We

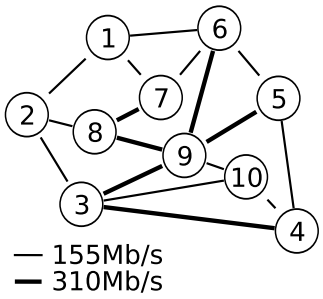

Figure 6.4: Simulation topology (10 nodes, 17 bidirectional links)

% Inputs processing and traffic generation
[C1, N]=Topology_generation(topology_type)
$M1_{EF}$, $M1_{AF1}$, $M1_{AF2}$, $M1_{AF3}$, $M1_{AF4}$, $M1_{BE}$ **% Original Traffic Matrices**
% Remarking AF out-of-profile traffic at edge node
[$M2_{AF1}$,$M2_{AF1OUT}$]=Admission_control($M1_{AF1}$,SLA_{AF1},N)
[$M2_{AF2}$,$M2_{AF2OUT}$]=Admission_control($M1_{AF2}$,SLA_{AF2},N)
[$M2_{AF3}$,$M2_{AF3OUT}$]=Admission_control($M1_{AF3}$,SLA_{AF3},N)
[$M2_{AF4}$,$M2_{AF4OUT}$]=Admission_control($M1_{AF4}$,SLA_{AF4},N)
$M2_{AFOUT}$=$M2_{AF1OUT}$+$M2_{AF2OUT}$+$M2_{AF3OUT}$+$M2_{AF4OUT}$
% Edge node EF admission control (dropping)
[$M2_{EF}$, dropped_ef]=Apply_SLA($M1_{EF}$, SLA_{EF}, N)
% Joint routing and bandwidth allocation
[obj_{EF},var_{EF},C2]=Bw_allocation(C1, N, $M2_{EF}$)
[obj_{AF},var_{AF},C3]=Bw_allocation(C2,N,$M2_{AF1}$+$M2_{AF2}$+$M2_{AF3}$+$M2_{AF4}$)
[obj_{BE},var_{BE},C4]=Bw_allocation(C3,N,$M1_{BE}$)
[obj_{AFOUT},var_{AFOUT},C5]=Bw_allocation(C4,N,$M2_{AFOUT}$)
[obj_{AF1}, var_{AF1}, obj_{AF2}, var_{AF2}, obj_{AF3}, var_{AF3}, obj_{AF4}, var_{AF4}] =share_allocated_bw(obj_{AF}, var_{AF})

Figure 6.5: Pseudo-code of the simulation framework

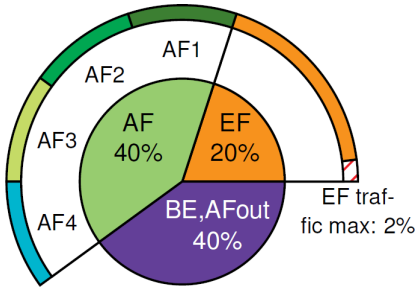

Figure 6.6: Bandwidth allocation for admission control

generate random resource demands for each class of service (EF, AF1 to AF4, BE) and store them in the traffic matrices $M1_{EF}, M1_{AF1}$ to $M1_{AF4}, M1_{BE}$. We use a lognormal probability density function to generate the resource demands, because this distribution reproduces the large variability of the demands in actual networks. The generated traffic is uniformly shared among the source-destination pairs, in average. The simulation is replicated many times with different random demands to derive statistically relevant estimates of the considered metrics. Confidence intervals are computed and are represented on the figures except if they are too small to be read.

As represented in Figure 6.6, for the admission control operations, we assume that about 20% of the bandwidth is reserved for EF, 10% for each class AF1 to AF4 and 40% for BE traffic. This traffic repartition is consistent with what was observed in a former project named VTHD and with the figures provided in [101]. An over-provisioning ratio of 10 is considered for the admitted EF traffic, to guarantee statistically low queuing delays. In the remainder of this chapter, we consider EF bandwidth demands, *i.e.* all subsequent traffic figures for EF represent 10 times the actual amount of EF traffic.

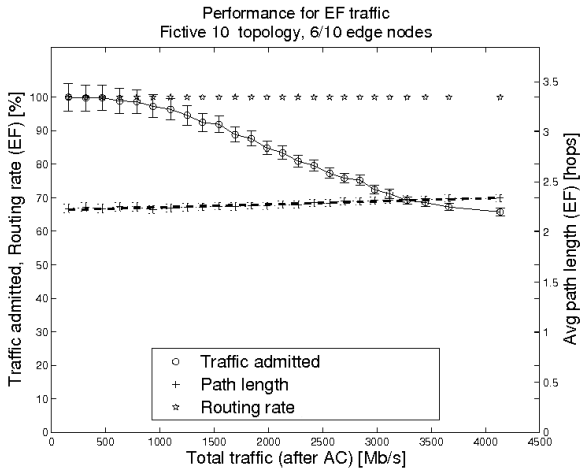

Figure 6.7: Performance for EF traffic

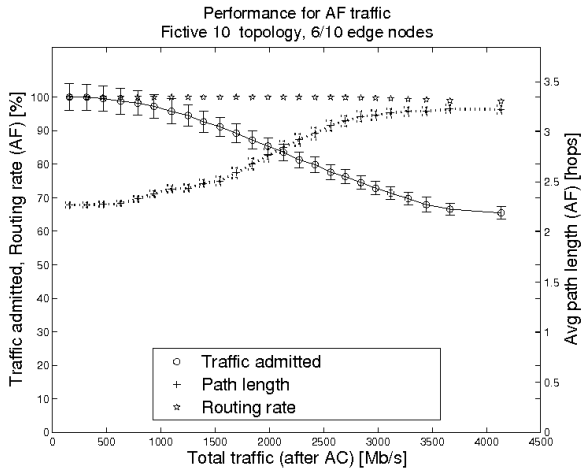

Figure 6.8: Performance for AF traffic

Admission control is performed on EF and AF classes and leads to admitted traffic matrices $M2_{EF}, M2_{AF1}$ to $M2_{AF4}$ and rejected (dropped_ef) or remarked traffic ($M2_{AFout}$). Routing and bandwidth allocation computations consider the admitted traffic matrices and are performed jointly, as presented in Section 6.2. They consider the traffic in the following order: EF, AF, BE, AF_{OUT}. After each bandwidth allocation stage (*e.g.,* after EF demands have been considered), network link capacities (*e.g.,* C1) are updated and subsequent computations are realized on the remaining link capacity matrix (*e.g.,* $C2$). The bandwidth allocated for AF traffic is shared among the AF classes as explained in Section 6.3.4. After bandwidth has been allocated for the BE traffic demands, the maximum AF_{OUT} flow that can be routed using the residual link capacities is computed.

6.4.2 Performance for Privileged Traffic

According to DiffServ QoS model, the admitted EF traffic is expected to experience low delay, low jitter, and low losses, while AF flows receive no delay guarantee but should experience low losses and better treatment than BE. The end-to-end delay has three main origins: propagation, queuing, and transmission delays. For EF traffic the queuing delays are typically small, thanks to the over-provisioning margin. Queuing and transmission delays are closely related to the number of times a packet is queued and transmitted, and therefore, closely related to path length. Thus, in our model, the average path length provides information about the end-to-end delay experienced by every class of service. We consider two additional metrics in the model to assess the performance experienced by EF and AF traffic: the percentage of traffic admitted after admission control and the percentage of this traffic that is successfully routed.

It can be seen in Figure 6.7 and Figure 6.8 that, in our model, EF and AF flows are slightly affected by the admission-control mechanism for traffic loads above 0.5 Gbps. In average, more than 90% of the privileged traffic is admitted for loads below 1.5 Gbps. Moreover, all AF and EF traffic is successfully routed. This means that the traffic that belongs to these classes and that is admitted in the network after admission control, experiences no loss. Finally, EF flows use shorter paths than AF, because they are routed first; for EF flows, the average path length is stable, while it increases for AF traffic. This indicates that EF traffic follows the shortest

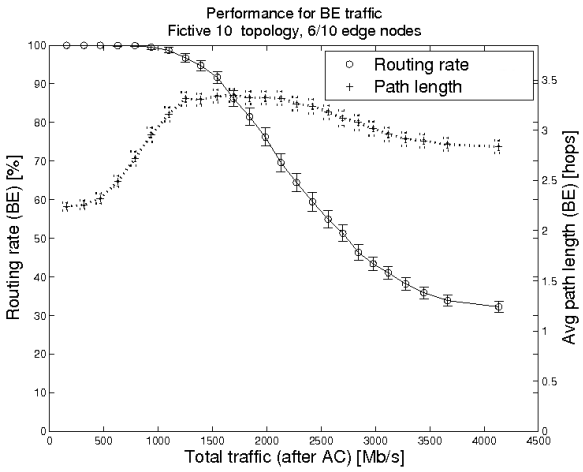

Figure 6.9: QoS-aware case: performance for BE traffic

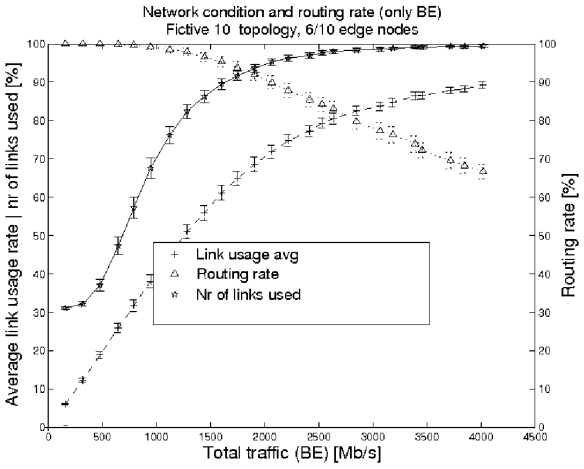

Figure 6.10: Network condition and routing rate for the QoS-agnostic case

paths; whereas, AF follows alternative paths. This property mimics the expected behavior of the EF and AF classes of service in the DiffServ model, which shows that our simple routing and bandwidth allocation scheme enables differentiating efficiently the network performance for the classes of service.

6.4.3 Performance for Best-Effort and for Out-of-Profile Traffic

We compare the performance experienced by BE traffic in two different scenarios. In the first, BE traffic is routed after EF and AF flows, while in the second there is no QoS traffic. In presence of QoS traffic (Figure 6.9), BE traffic experiences losses when the overall amount of traffic after admission control is larger than 1 Gbps. In average, more than 90% of BE is nevertheless routed for traffic loads below 1.5 Gbps. In average, the paths used by BE traffic (μ=3.0) are about half a hop longer (+0.7) than the ones used by EF traffic (μ=2.3). The performance is even slightly better when there is no privileged traffic (Figure 6.10), which was expected as EF and AF traffic are routed first and take advantage of the best paths. Indeed, the average path length decreases slightly to 2.8 hops and more traffic is routed in average. The main conclusion that can be drawn from these first curves is that our bandwidth allocation model is efficient to differentiate the network performance depending on the class of service.

Figure 6.10 depicts the evolution of three metrics that describe the network condition in presence of only BE traffic: the average link-usage rate, as well as its standard deviation and the number of links used. The average link-usage rate is quasi proportional to the total traffic load, up to a load of about 1.75 Gbps, and reaches a maximum value of about 80%. The number of used links shows that a stable proportion of 30% of the links, probably belonging to the best paths, is used for traffic loads up to about 0.3 Gbps. Above this threshold, more and more links are used until all links are employed. A similar threshold is to be noticed for the standard deviation of the link-usage rate, when the traffic load is about 1.25 Gbps. Below this limit, the traffic repartition is less and less balanced when the load increases, while for higher traffic loads it is more and more balanced. This matches the intuition that the links

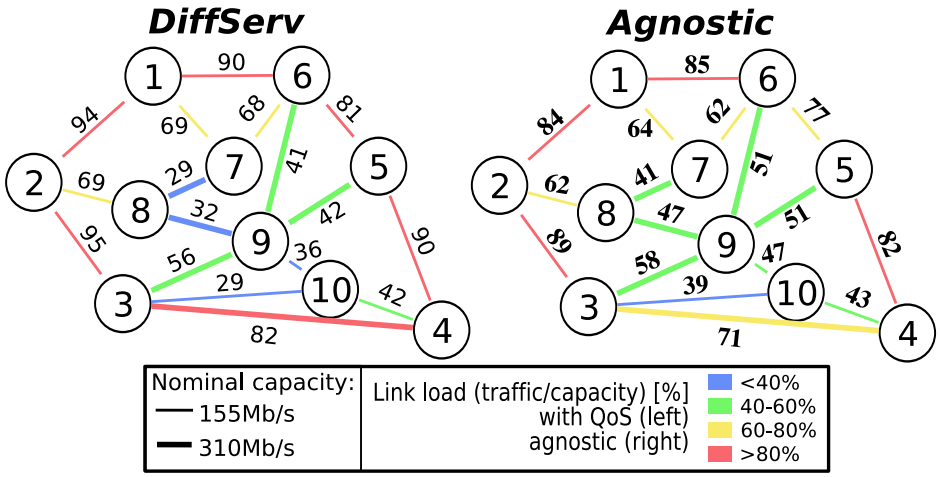

Figure 6.11: Link capacity after allocation

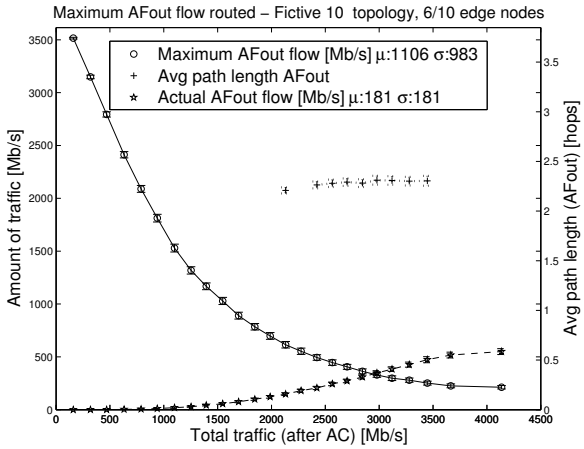

Figure 6.12: Maximum AF_{OUT} routed traffic

on the best paths will be used first, until they are filled, and then alternate paths with larger remaining capacity will be used.

Figure 6.11 presents the state of the network in the two simulated scenarios after routing BE traffic. The values presented are the remaining link capacities averaged on all simulation runs. In average, the links involving core nodes have more remaining capacity (μ=53% vs. μ=49%) than the ones between edge nodes (μ=11% vs. μ=19%). This confirms that the core of the studied network topology is probably over-dimensioned.

Figure 6.12 presents the maximum AF_{OUT} traffic flow that could be routed after all privileged and BE traffic, and compares it to the actual amount of remarked AF traffic. An additional metric shows the path lengths that would be experienced by AF_{OUT} traffic when the maximum AF_{OUT} flow is routed. The maximum AF_{OUT} flow takes huge values because it does not consider constraints such as the demand profile, *etc.* It provides an indication on how much additional traffic could be carried by the network.

6.5 Conclusion

We have presented a study of QoS management in DiffServ traffic-engineered networks. In particular, we have proposed a simple efficient model for offline traffic engineering (routing and bandwidth allocation) in DiffServ-MPLS networks. Our work focused on *offline* bandwidth allocation for every CoS in a DiffServ traffic-engineered network to guarantee *differentiated QoS* levels. We have assessed the performance of our model, through numerical simulations, with regard to a number of factors including the amount of traffic admitted and routed, the average path lengths, the link load repartition, *etc*. The simulations show that our model enables operators to differentiate the network performance for the classes of service and to use the network resources efficiently. Moreover, our model returns performance predictions for each class of service and evaluates the capacity of a network to admit additional traffic flows. The main conclusion that can be drawn from Chapter 6 is that it is complicated for network operators to configure their network to enforce the specific service guarantees that they contract with their customers. Dimensioning the network and provisioning an appropriate amount of resources to critical flows are difficult tasks. Therefore, our model provides a simple method to compute an efficient routing and bandwidth allocation configuration for all the classes of service and returns valuable information about the remaining capacity of the network after serving the expected traffic load.

In subsequent chapters, we complement this study by considering mechanisms for *online* resource provisioning to guarantee *strict QoS* levels at inter-domain level. More specifically, we study the problem of routing critical flows that traverse several domains on paths that provide QoS guarantees.

Key points of Chapter 6

- The chapter focuses on *offline* bandwidth allocation for every CoS in a DiffServ traffic-engineered network to guarantee *differentiated QoS* levels.
- We describe technologies to implement differentiated QoS based on traffic predictions and network dimensioning. Specifically, we propose a joint bandwidth allocation and routing framework for traffic-engineered DiffServ networks.
- As a bandwidth allocation framework, our model enables operators to use the network resources efficiently and to differentiate the network performance for the supported classes of service. Furthermore, our model is interesting to evaluate the capacity of a network to admit additional traffic flows.
- Dimensioning an operator's network and provisioning an appropriate amount of resources beforehand to critical flows are difficult tasks. Dynamic provisioning technologies represent an interesting alternative for implementing specific QoS levels. Hence, in subsequent chapters, we complement this study by considering mechanisms for *online* resource provisioning to guarantee *strict QoS* levels for inter-domain services.

CHAPTER 7

Resource Provisioning for Inter-Domain Traffic

7.1 Introduction

Recent advances in inter-domain TE, and in particular the definition of the PCE framework, open novel technical perspectives for the configuration of inter-domain paths with guaranteed performance. These technologies enable dynamic resource provisioning for critical flows at inter-domain level. For example, they can be used to allocate network resources for inter-domain virtual private networks (VPNs). The paths along which the resources are allocated depend on many constraints, such as the required QoS level and traffic-engineering objectives. In the present chapter, we define the mathematical problem that consists in finding appropriate paths to provision resources for performance-demanding services across domain boundaries

The placement of inter-domain flows on paths that offer specific performance guarantees is a constrained routing problem. When several constraints are considered, it is called the *multi-constrained path* (MCP) problem and is $\mathcal{NP}$-hard. We define this problem in Section 7.2 and describe the origin of its computational complexity in Section 7.3. Due to its important applications for TE and for QoS routing, the MCP problem has already been extensively studied. However, we explain in Section 7.4 that previous studies did not consider the problems introduced by the division of a network into domains. Thus, they are not applicable for solving the inter-domain problem. That is why we introduce the specificities of *inter-domain* MCP computation in Section 7.4.1 and our resolution approach in Section 7.5.

7.2 The Underlying Mathematical Problem

In the present thesis, we use the term MCPs to refer to paths subject to multiple constraints related to several metrics. The problem of computing MCPs is named the *MCP problem* and occurs when a system must determine a path, with guarantees on several performance criteria, between a given source and a given destination in a graph. For example, one of the purposes of automotive navigation systems is to find a route for which the travel time, the gas consumption, and the paid toll take reasonable values. Similarly, in telecommunications networks, the MCP problem arises when an ISP determines a path with QoS and TE performance guarantees.

Routing protocols usually advertise the value of a single metric for every link, and shortest-path algorithms use this value to compute the routes. However, to support a wide range of QoS requirements, routing must characterize the network with multiple metrics. For example, it is not straightforward to express reliability constraints with delay metrics.

In this paragraph, we develop a simple example that illustrates the fundamental role of the MCP problem in QoS aware routing and in TE. Consider two ISPs that provide a telephony service to customers in different countries and that inter-connect their voice over IP (VoIP) gateways for long-distance calls. The performance of the default IP routes might be insufficient to support VoIP traffic. Thus, the ISPs could impose constraints on specific routing metrics so that the VoIP traffic follows paths that provide a guaranteed propagation delay, a minimum available bandwidth, as well as a good reliability and a good availability. In addition, they could add a constraint on the hop-count to preserve network resources, as the hop-count captures the number of links over which resources are allocated [84]. This simple scenario highlights the fact that a network operator can be interested in imposing varied routing constraints on critical flows.

7.2.1 Classification of Path Constraints

Two main types of constraints are usually considered in path computation problems. The first one is called *optimum constraint*: it imposes to select a path for which the value of a parameter is the lowest. For example, an ISP can compute the shortest path in terms of the number of traversed links. We name *objective function* the expression that gives the value for this parameter (*e.g.,* the number of traversed links). The second type of constraints defines a *bound* on the acceptable value of a metric for the selected path. For instance, a service may require that the end-to-end propagation delay is below fifty milliseconds.

Note that it is equivalent to compute paths with the highest value of an objective function or with the lowest value of the opposite function. Similarly, a lower bound constraint on a metric is equivalent to an upper bound constraint on the opposite of the considered metric. Therefore, we can consider only upper-bound and lowest-value constraints, without loss of generality.

Bound constraints related to bottleneck metrics, such as a minimum acceptable bandwidth, are easy to treat: the path computation algorithm can ignore the edges whose value for the bottleneck metric is less than the constraint. This method is called *edge pruning* and its operations take polynomial time [183]. Therefore, multi-objective routing problems typically include constraints only on *additive* metrics[1], such as the propagation delay.

7.2.2 Mathematical Formulation of the Routing Problem

Notations and Vocabulary

Due to the number and the importance of its applications, the MCP problem is described in many papers under various names. For example, the MCP problem with two constraints is often called the restricted shortest path (RSP) problem, or more specifically, the delay-constrained least-cost path, minimum-cost restricted-time path, and constrained shortest path. In the present section, we define *the general MCP problem*, without restrictions on the number of considered metrics.

[1]For a definition of additive metrics, please refer to Section 4.2.3

Notation	Meaning
$\mathbb{N}$	Set of the natural integers
$\mathbb{R}$	Set of the real numbers
$G = (V, E)$	A graph
$G = (V, E, \vec{w})$	An edge-weighted graph
V	Set of nodes (vertices)
E	Set of links (edges)
$\|V\|$ or V	Number of nodes (vertices)^2
$\|E\|$ or E	Number of links (edges)
$\vec{w}$	The link weight function
l	A link
$\vec{w}(l)$	The weights of a link
$w_k(l)$	The k-th weight of a link
K	The number of link weights
$P_{s\to t}$	The set of the paths from s to t
$\mathbf{p}$	A path
$\vec{W}$	A set of constraints on the path weights
W_k	A constraint on the k-th weight
D	A domain
$\mathbf{S}$	A domain sequence

Table 7.1: Summary of frequently-used notations

In this document, we denote the set of the non-negative integers as $\mathbb{N}$, the set of the strictly positively-valued real numbers as $\mathbb{R}^{+*}$, and the set of the non-negative real numbers as $\mathbb{R}^+$. In addition, we represent the intervals of non-negative integer numbers as follows:

$$[a..b] \equiv \{i \in \mathbb{N} | a \leq i \leq b\}. \tag{7.1}$$

We consider a network represented by a directed graph $G = (V, E)$ where V is the set of vertices or *nodes* and E the set of edges or *links*. To model multiple QoS metrics, each edge l in E is associated with a vector $\vec{w}(l)$ of $K \in \mathbb{N}$ non-negative real-valued link weights $w_k(l) \in \mathbb{R}^+$, with k in $[1..K]$. The weights represent the value of every considered metric for the link l, thus, we use the terms *link weight* and *metric value* indifferently. We assume that at least one of the K weights associated with an edge l is different from zero.

We define a *path* as a finite sequence of adjacent edges. We denote the set of all paths in the considered graph G as P_G. We denote the set of paths from a node s to a node t as $P_{s\to t} \subset P_G$. Similarly, we denote the set of paths whose source is in the subset $D \subset V$ and whose destination is the node t in V as $P_{D\to t}$. Conversely, we refer to the set of paths whose source is the node s in V and whose destination is in the set $D \subset V$ as $P_{s\to D}$. We denote the concatenation of two paths $\mathbf{p}_1 \in P_{s\to a}$ and $\mathbf{p}_2 \in P_{a\to t}$ as $(\mathbf{p}_1 \cup \mathbf{p}_2) \in P_{s\to t}$. We consider two non-empty paths $\mathbf{p}$ and $\mathbf{s}$. We say that $\mathbf{s}$ is a *suffix* of $\mathbf{p}$ if there is a path $\mathbf{q}$ such that $\mathbf{p}$ is the concatenation of $\mathbf{q}$ and $\mathbf{s}$. We denote the sequence of edges obtained by removing a sub-path $\mathbf{s}$ from a path $\mathbf{p}$ as $\mathbf{p} \setminus \mathbf{s}$.

We name *cost function* any function $c : P_G \to \mathbb{R}^+$ that associates a non-negative real number to a path. In addition, we associate each path $\mathbf{p}$ in P_G with a vector $\vec{w}(\mathbf{p})$ of K non-negative real weights $w_k(\mathbf{p}) \in \mathbb{R}^+$ with k in $[1..K]$. The path weights represent the value

of *additive* metrics, thus, we define them by:

$$w_k(\mathbf{p}) \equiv \sum_{l \in \mathbf{p}} w_k(l),\ k \in [1..K]. \tag{7.2}$$

Definition

To define the MCP problem, we consider a *path computation request*. Such request specifies a source s and a destination t in V, as well as K constraints W_k in $\mathbb{R}^{+*}$ with k in $[1..K]$ on the requested path $\mathbf{p}$. The considered constraints represent maximum bounds W_k on every weight $w_k(\mathbf{p})$: to be acceptable a path $\mathbf{p} \in P_{s \to t}$ must satisfy $w_k(\mathbf{p}) \leq W_k$ for all k in $[1..K]$. We call *feasible path* any path that fulfills the constraints of the request; the MCP problem consists in determining a feasible path. For instance, the problem of finding a path whose end-to-end propagation delay is below fifty milliseconds and which traverses less than fifteen links is an MCP problem.

Problem (MCP)**.** *Given a source s and a destination t in an edge-weighted graph $G(V, E, \vec{w})$, an integer $K \in \mathbb{N}$ such that $K \geq 2$, and K bound constraints W_k with k in $[1..K]$, find a path $\boldsymbol{p}$ in $P_{s \to t}$ such that $w_k(\boldsymbol{p}) \leq W_k$, for all k in $[1..K]$.*

The inter-domain MCP problem is a generalization of the MCP problem: the former specifies that the source and destination nodes are in different domains, whereas the latter does not consider the division of a network into domains.

We introduce a few new definitions and notations to formalize the inter-domain problem. We consider a network represented by an edge-weighted graph $G = (V, E, \vec{w})$ and we define a partition of the set of nodes V. We call *domain* any element of this partition. We say that a link $l = (u, v)$ in E is *intra-domain* if there is a domain $D \subset V$ such that $u \in D$ and $v \in D$. Moreover, we say that a link $l = (u, v)$ in E is an *inter-domain* link if there is a domain $D \subset V$ such that $u \in D$ and $v \notin D$. We say that two domains are *neighbors* if an inter-domain link connects them. We call *entry-BN* of a domain D every node v in D for which there is a node u in $V \setminus D$ such that the inter-domain link (u, v) exists in E. We denote as $P^{\mathbf{S}}_{s \to t}$ the set of paths from a source s to a target t that traverse a particular domain sequence $\mathbf{S}$.

Problem (InterMCP)**.** *Given a (finite) loop-free domain sequence $\boldsymbol{S} = (D_1, D_2, \ldots)$ that contains $|\boldsymbol{S}|$ domains, a source $s \in D_1$ and a target $t \in D_{|\boldsymbol{S}|}$, an integer $K \in \mathbb{N}$ such that $K \geq 2$, and K bound constraints $W_k \in \mathbb{R}^+$ with k in $[1..K]$, find a path $\boldsymbol{p}$ in $P^{\boldsymbol{S}}_{s \to t}$ such that $w_k(\boldsymbol{p}) \leq W_k$, for all k in $[1..K]$.*

7.3 Tractability of Constrained-Path Computations

7.3.1 The Intra-Domain MCP Problem is $\mathcal{NP}$-Complete

Determining MCPs requires hard computations: Wang and Crowcroft have proved that the MCP problem with two or more additive metrics is $\mathcal{NP}$-complete [180]. To establish the $\mathcal{NP}$-completeness of the MCP problem, they proceed by induction. They exhibit two polynomial transformations: first, from the well-known $\mathcal{NP}$-complete PARTITION problem [82] to the MCP problem with two constraints, and then, from the MCP problem with n constraints to the MCP problem with $n + 1$ constraints. With this proof and related lemmas, Wang and

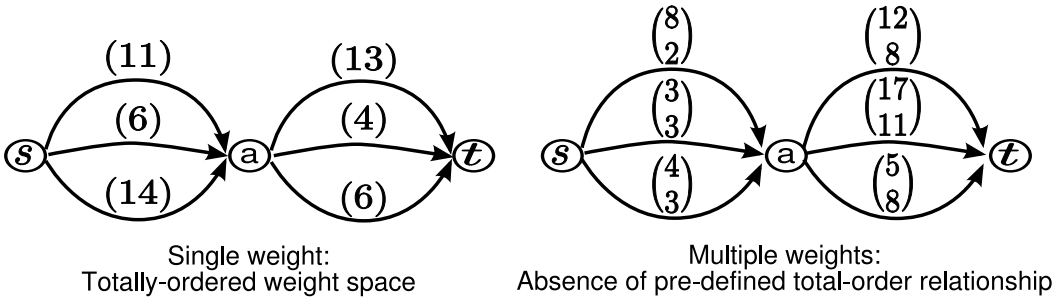

Figure 7.1: Absence of total order relationship for multi-dimensional weight spaces and consequences for the path computation algorithms

Crowcroft demonstrate that any combination of two metrics among delay, delay jitter, cost, and loss probability, leads to an $\mathcal{NP}$-complete MCP problem [180]. The $\mathcal{NP}$-completeness implies that some instances of the MCP problem cannot be solved in practice because the time complexity of the computation is prohibitive [82, 185].

7.3.2 The Inter-Domain MCP Problem is $\mathcal{NP}$-Complete

We investigate the computational complexity of the inter-domain MCP problem. We know that the problem MCP is $\mathcal{NP}$-complete. In addition, we show that the MCP problem is a special case of the problem InterMCP. Thus, the problem InterMCP is $\mathcal{NP}$-complete [82].

Theorem. *The problem InterMCP is $\mathcal{NP}$-complete.*

Proof. We consider an instance of the problem MCP with a source s, a target t, and an integer number $K \geq 2$ of bound constraints W_k with k in $[1..K]$ in a graph $G(V, E)$. The problem InterMCP with the same source, target, and constraints, and with the loop-free domain sequence $\mathbf{S} = (V)$ is equivalent[3] to the previous instance of the problem MCP. □

As the problem InterMCP is $\mathcal{NP}$-complete, some of its instances cannot be solved exactly in polynomial time by a deterministic Turing machine.[4] Nevertheless, many instances of the inter-domain MCP problem are solvable in polynomial time, as explained in Section 7.3.4.

7.3.3 Origin of the Large Complexity

In this section, we explain that the absence of a *total order relationship* [36] on the set of the path weights is one of the causes for the $\mathcal{NP}$-completeness of the MCP problem. To clarify our explanations, we compare the MCP problem to the problem with a single constraint.

The problem of computing a path subject to a single bound constraint on an additive metric is solvable in polynomial time with Dijkstra's or Bellman-Ford algorithm [72]. These algorithms maintain a single *shortest* path from the considered source node s to every intermediate node a. We consider the example in Figure 7.1: in this figure, there are several candidate paths between s and a, with path weights 11, 6, and 14. It is clear that the intermediate path with weight 6 leads to better end-to-end paths between s and t than the two

[3] Both problems have exactly the same solutions.
[4] If $\mathcal{NP} \neq \mathcal{P}$ [82].

alternative paths, because its weight is the lowest. Thus, the path computation algorithms need to memorize only this intermediate path. Usually, determining the path that the algorithm must memorize is straightforward because the paths weights take their values in $(\mathbb{R}, \leq)$, which is a *totally ordered set* [36].

By contrast, the MCP problem does not define a method to compare the weight vectors of two paths. Given two paths $\mathbf{p}_1$ and $\mathbf{p}_2$ between two nodes s and a, it is not always possible to determine if the weights $\vec{w}(\mathbf{p}_1)$ are better than the weights $\vec{w}(\mathbf{p}_2)$. We illustrate this problem on the example in Figure 7.1. There are three possible intermediate paths $\mathbf{p}_1$, $\mathbf{p}_2$, and $\mathbf{p}_3$ between two nodes s and a, with $\vec{w}(\mathbf{p}_1) = (4,3)^T$, $\vec{w}(\mathbf{p}_2) = (8,2)^T$, and $\vec{w}(\mathbf{p}_3) = (3,3)^T$. All the weights of $\mathbf{p}_3$ are lower than the ones of $\mathbf{p}_1$, so it is clear that the best end-to-end s-t-path does not include $\mathbf{p}_1$. However, $\mathbf{p}_3$ is better than $\mathbf{p}_2$ for the weight w_1 but worse than $\mathbf{p}_2$ for w_2. Therefore, it is hard to predict which of these paths shall lead to the best end-to-end solutions from s to t and we must consider both paths during MCP computation operations. This simple example shows that to solve certain instances of the MCP problem, an algorithm must consider many sub-paths to intermediate nodes and must combine them to the possible links toward the target, which drastically affects the computational complexity.

Kuipers and Van Mieghem [111] assert that the $\mathcal{NP}$-complete behavior of the MCP problem emerges only in particular network configurations that occur rarely in practice. In Reference [112], they investigate the conditions that affect the complexity of the MCP problem. In particular, they consider the number of paths to intermediate nodes that must be considered to solve the MCP problem exactly. They show that this number depends on the network topology, the granularity of the link weights, the correlation of these weights, and the constraints. In the next section, we detail MCP problem instances that are important in practice and that can be solved in polynomial time.

7.3.4 Problem Instances Solvable in Polynomial Time

Influence of the Metrics on the Complexity

The MCP problem can be simplified when the considered link metrics are bound by inter-dependency relationships. In particular, Ma and Steenkiste [120] show that for a broad class of scheduling algorithms the problem of finding a path subject to constraints on bandwidth, delay, jitter, and buffer space is solvable in polynomial time, because delay, jitter, and buffer space become functions of the bandwidth. Orda [134] describes the metric relationships for networks that use rate-based schedulers. In addition, he exploits the typical hierarchical structure of large-scale networks to derive an efficient algorithm for the minimum-cost bounded-delay problem. The work of Ma, Steenkiste, and Orda enables operators to solve the routing problems that involve delay and jitter constraints in a simple manner in networks that use rate-based scheduling. They first translate the constraints into a bandwidth requirement, and then they solve the bandwidth-constrained problem with usual methods (*e.g.,* link-pruning algorithm). Nevertheless, in the general case, some metrics, such as the propagation delay, cannot be formulated as a function of bandwidth. Thus, these methods do not solve the general MCP problem.

In addition to the inter-dependency of the link metrics, the granularity of the metric values affects the complexity of the MCP problem. Yuan [188] and Chen *et alii* [46] explain that when all considered metrics except one take bounded integer values, the MCP problem is solvable in polynomial time. In particular, Chen and Nahrstedt propose extended versions of the Bellman-Ford and the Dijkstra's algorithm that can solve this problem [46]. The MCP

problem with limited weight granularity is important in practice, because routing protocols usually dedicate a fixed-sized field for link-metric values (*e.g.,* 16 bits), and thus, link-weight values have a limited granularity.

Influence of the Constraints on the Complexity

Several researchers, such as Kuipers and Van Mieghem [111, 112], have highlighted the influence of the request constraints on the complexity of the MCP problem. To synthesize, the MCP instances with constraints W_1 and W_2 such that $w_1(\mathbf{p}_1) \leq W_1 < w_1(\mathbf{p}_2)$ and $w_2(\mathbf{p}_2) \leq W_2 < w_2(\mathbf{p}_1)$, where $\mathbf{p}_k$ is the end-to-end path that minimizes w_k, are usually difficult to solve. However, if the constraints are outside these intervals, we can solve the problem with a polynomial complexity.

We explain the origin of these inequalities on an example with two constraints. Given two nodes s and t in V, consider a path $\mathbf{p}_1 \in P_{s\to t}$ that verifies $w_1(\mathbf{p}_1) = 7$ and that minimizes the weight w_1. Moreover, consider a path $\mathbf{p}_2 \in P_{s\to t}$ that verifies $w_2(\mathbf{p}_2) = 16$ and that minimizes the weight w_2. If the request constraint W_1 on the weight w_1 verifies $W_1 < w_1(\mathbf{p}_1)$, for example $W_1 = 5$,[5] then, the considered MCP problem is *over-constrained*. In concrete terms, it is possible to compute $\mathbf{p}_1$ and $\mathbf{p}_2$ in polynomial time and to use their weights to determine that there is no solution to the MCP problem. Conversely, if the request constraints verify ($W_1 \geq w_1(\mathbf{p}_1)$ and $W_2 \geq w_2(\mathbf{p}_1)$),[6] for example $W_1 = W_2 = 20$ and $w_2(\mathbf{p}_1) = 17$, then, the considered MCP problem is *under-constrained*. In this case, $\mathbf{p}_1$ or $\mathbf{p}_2$ is an evident solution that we can compute in polynomial time.

7.4 The Need for Novel Solutions

7.4.1 Specific Challenges of the Inter-Domain MCP Problem

In the present section, we describe specific challenges of inter-domain path computation that forbid ISPs to apply existing MCP algorithms for inter-domain routing.

The inter-domain routing problem is subject to a number of challenges that make it especially complicated. For example, the domains usually have a wide autonomy (*e.g.,* BGP autonomous systems). Every ISP controls its routing according to specific management policies. Due to the competitive relationships among the ISPs, it would not be acceptable for any ISP to delegate the control over its routing to another operator. Thus, centralized computations do not seem relevant for determining inter-domain paths, because they imply that a single point would control the routing of several operators. To preserve the routing autonomy of the network operators, the computation of inter-domain paths must typically be distributed among the domains.

In addition to the autonomy requirements, security and scalability limitations deprive path computation entities from valuable information about the internal state of other domains. The absence of trust among the involved domains introduces visibility limitations for every domain. For instance, link-state routing, which provides every router with a complete view of the state of the network, is inapplicable at inter-domain level.

[5] or the second constraints verifies $W_2 < w_2(\mathbf{p}_2)$, for example $W_2 = 12$

[6] or ($W_1 \geq w_1(\mathbf{p}_2)$ and $W_2 \geq w_2(\mathbf{p}_2)$)

Relevant work of the domain, such as [175], takes a hypothesis to simplify path computations, to improve routing protocol scalability, and to cope with domain visibility limitations. It assumes that the path computation entities know a domain-sequence, for every acceptable path computation request and before computing the router-level paths. This assumption separates intra- and inter-domain routing operations and simplifies the path computation operations [187].

7.4.2 Inapplicability of Usual MCP Algorithms

The most well-known existing MCP algorithms do not consider the specificities of inter-domain routing. Therefore, they are not adapted for the computation of paths with performance guarantees in the context of inter-domain resource provisioning.

The main limitation of existing solutions is that they typically require centralized computation operations and link-state routing, which are not applicable for inter-domain routing, as explained in Section 7.4.1. For example, Chen and Nahrstedt [46] postulate that source routing is adopted and that every node maintains state information for the complete network. Andrew and Kusuma [6], as well as Yuan [188] assume centralized computations with link-state routing. Finally, SAMCRA [172] and H_MCOP [109] use centralized computations with variations of Dijkstra's or of Bellman-Ford's algorithm to perform *look-ahead*.

Recently, a few researchers have highlighted the limitations of link-state routing and of centralized computations for the MCP problem. In particular, Li and Garcia-Luna-Aceves [116, 117] propose a distributed approach for MCP selection. They describe MPOR, a heuristic that solves the MCP problem using distance vectors exchanged only among neighboring nodes. Their method is interesting and is the first distributed procedure for the MCP problem. They focus on IP-based connectionless routing: their approach relies on the distance-vector algorithm, according to which all the nodes play the same role. We think that a point in their approach is problematical: MPOR requires that every node runs a separate copy of MPOR, runs MPOR for every destination, and maintains the x shortest paths toward every destination [116]. Consequently, we consider that the memory requirements of MPOR are prohibitive for large networks and make it inapplicable for inter-domain scenarios.

Zhang *et alii* [190] propose a BGP QoS extension with multiple metrics. They describe the enhanced inter-domain information exchanges as well as the path selection procedure for inter-domain MCP computations. Their approach is interesting, however, it requires several significant modifications of BGP, which compromise its adoption by network operators. Moreover, the impact of the proposed BGP changes on BGP's convergence time is unclear. Finally, they do not explain how to differentiate the routing performance for every traffic class, whereas IP-based routing uses the same path for all the traffic.

Saad *et alii* [150] describe another interesting approach. They propose an algorithm for computing end-to-end paths crossing multiple wavelength division multiplexing (WDM) optical domains and subject to two constraints. Their solution extends Jaffe's algorithm [98] for inter-domain computations. Their idea is to compute MCPs in a hierarchical topology, considering a graph of inter-connected border routers. Thus, they assume that global state information is available in each border router. Their proposal suffers from the two following limitations.

1. They restrict their study to the two-constraint problem.

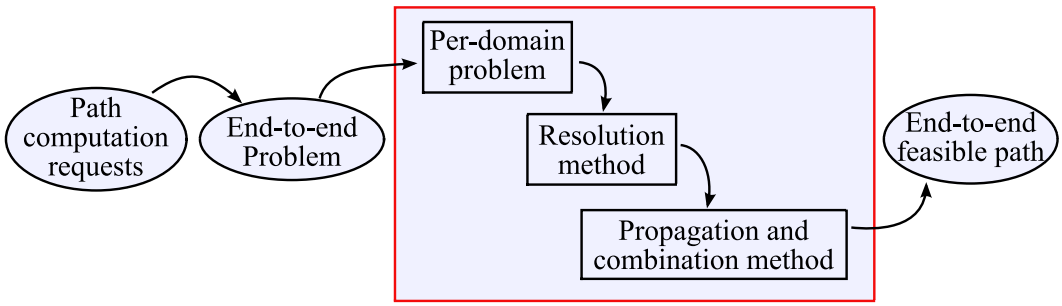

Figure 7.2: Proposed structure for distributed solutions to the inter-domain MCP problem

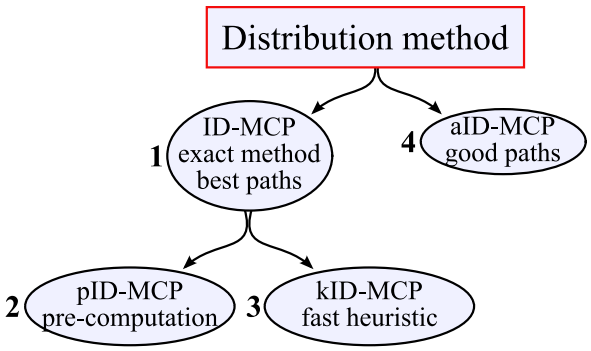

Figure 7.3: Summary of the proposed inter-domain path computation algorithms

2. They use the linear path length of Jaffe's algorithm, which is less efficient the non-linear path-length functions used by recently-proposed algorithms [109, 183].

7.5 Our Approach to the Inter-Domain MCP Problem

The preceding sections have explained the need for *distributed solutions* to the inter-domain MCP problem. To enable distributed computations, we propose three building blocks, as illustrated in Figure 7.2: (1) the formulation of a per-domain problem, (2) an algorithm that solves the per-domain problem, and (3) a method for propagating and for combining the computation results to determine the end-to-end path. The building blocks give birth to a variety of algorithms, depending on the considered per-domain formulation.

In the following sections, we describe several algorithms, which are all based on the aforementioned distribution method. Their characteristics are described in Figure 7.3: ID-MCP [22], which is presented in Chapter 9 is an exact solution. Therefore, it finds the best possible solutions to the inter-domain MCP problem. However, its complexity is prohibitive for particular problem instances. Consequently, we have designed a pre-computation approach, named pID-MCP, which enables computing per-domain representations offline and performing only simple operations online [21]. In Chapter 10, we have studied approximation methods (aID-MCP), which compute feasible paths in polynomial time and have a guaranteed accuracy. Finally, we have proposed a fast and efficient heuristic, called *k*ID-MCP [23]. We present this heuristic in Chapter 11.

7.6 Conclusion

In the present chapter, we have defined the problem of finding inter-domain paths that provide performance guarantees. The inter-domain MCP problem is the fundamental optimization problem that corresponds to the inter-domain path calculation. We have proved that this problem is $\mathcal{NP}$-Complete. In addition, our study has shown that most existing MCP algorithms cannot realistically be applied to compute inter-domain MCPs, because they require centralized computations and link-state routing. Therefore, in next chapters, we analyze the principles of existing algorithms, and we propose novel solutions that are better adapted for inter-domain routing.

Key points of Chapter 7

- The fundamental mathematical problem behind constrained routing is called the MCP problem and is computationally hard. Nevertheless, the MCP problem is tractable in specific situations (*e.g.,* links metrics with finite granularity). In all other situations, efficient heuristics and approximation algorithms can be used.
- Many solutions have been proposed, but they cannot be applied for inter-domain constrained routing because of security (confidentiality), autonomy, and scalability constraints.
- We propose a general approach to solve the inter-domain constrained routing problem. Our method features both *distributed computations* for solving per-domain problems and *propagation/combination operations* to derive end-to-end feasible paths from per-domain results.

CHAPTER 8

Analysis of Existing Multi-Constrained Routing Algorithms

8.1 Introduction

In this chapter, we describe existing solutions to the fundamental problem of QoS routing: the multi-constraint path (MCP) problem. This problem has important applications in telecommunications networks, and thus, has been already well investigated. We provide a synthesis of previous work and we exhibit the principles of the available exact and approximate algorithms. This study helps us to design the solutions to the inter-domain constrained routing problem that we present in subsequent chapters.

The remainder of the chapter is structured as follows. Section 8.2 presents the methods that allow transforming the MCP problem into a single-objective optimization problem. Then, we analyze the principles of two categories of MCP algorithms: exact brute-force solutions in Section 8.3 and fast algorithms in Section 8.4. Section 8.5 concludes the chapter with a synthesis of the most well-known MCP algorithms and of their computational complexity.

8.2 From Multi-Objective to Single-Objective Optimization

8.2.1 Least-Cost Path Problems

Multi-objective optimization problems are considered as complicated. Therefore, several techniques have been developed to translate them into mono-objective problems that lead to equivalent solutions. Similarly, it is preferable to solve a (usually simpler) least-cost path problem instead of the original MCP problem. The considered instance of the least-cost path problem is defined so that its solutions solve the original MCP problem too. This method allows path computation algorithms to consider a single objective function, which combines the constraints on additive metrics and which is called *path length* function or *cost* function[1].

Problem (Least-Cost Path)**.** *Given a source s and a target t in an edge-weighted graph*[2] *$G(V, E, \vec{w})$, an integer $K \in \mathbb{N}$ such that $K \geq 2$, K bound constraints W_k with k in $[1..K]$,*

[1] Some papers use the term *energy function*, for example [150].

[2] Section 7.2.2 defines the graph theory vocabulary and the notations used in the thesis.

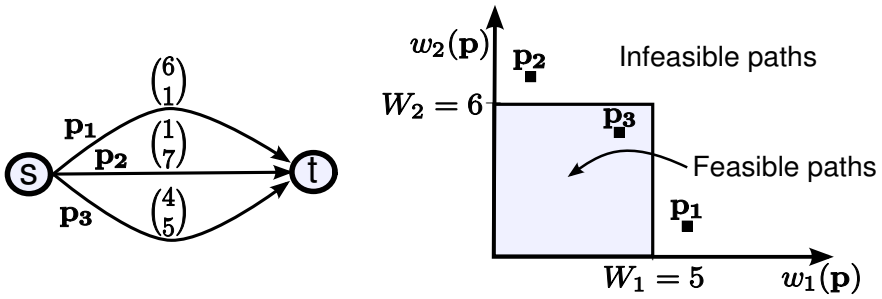

Figure 8.1: Example in which none of the least-cost paths for a linear path-length function is feasible, whereas the network contains a feasible path.

and a path-length function c, find a path $\boldsymbol{p}^*$ *in* $P_{s\to t}$ *such that for every path* $\boldsymbol{p}$ *in* $P_{s\to t}$, $c(\boldsymbol{p}^*) \leq c(\boldsymbol{p})$.

The choice of the considered path-length function for the least-cost problem is fundamental to guarantee that either the computed least-cost path is a feasible solution for the original MCP problem or the graph does not contain any feasible path for the original MCP problem.

Henceforth, we name *optimal* solutions or *shortest* paths the least-cost paths, between the source and the target of the considered path computation request, for the considered path-length function.

8.2.2 Linear Path-Length Functions

Linear path-length functions, defined in [6], obey to a strict triangle inequality:

$$c(\mathbf{p}_1 \cup \mathbf{p}_2) = c(\mathbf{p}_1) + c(\mathbf{p}_2). \tag{8.1}$$

For example, Equation (8.2) describes a commonly-used linear path-length function, where d_k in $\mathbb{R}^+$ are arbitrary multipliers and w_k are normalized path weights [6].

$$c : \mathbf{p} \to \sum_{k\in[1..K]} d_k \cdot w_k(\mathbf{p}) \tag{8.2}$$

Lemma 8.2.1 describes an interesting characteristic of linear expressions of the path length. Its proof is provided in Appendix B.

Lemma 8.2.1. Any shortest path $\mathbf{p}^*$ for the linear path length c is made up of shortest sub-paths.

Cormen *et alii* [50] name *"optimal sub-structure"* the fact that the solution of an optimization problem contains within it optimal solutions to sub-problems, as in Lemma 8.2.1. This property is fundamental for dynamic programming and for greedy algorithms; in particular, it allows the use of common shortest-path algorithms like the Dijkstra's or Bellman-Ford algorithm.

Jaffe [98] showed in 1984 that shortest path computations with linear path-length functions provide satisfying results for the MCP problem, especially if the link weights are positively correlated. More recently, Khadivi *et alii* [103] compared the performance of least-cost path

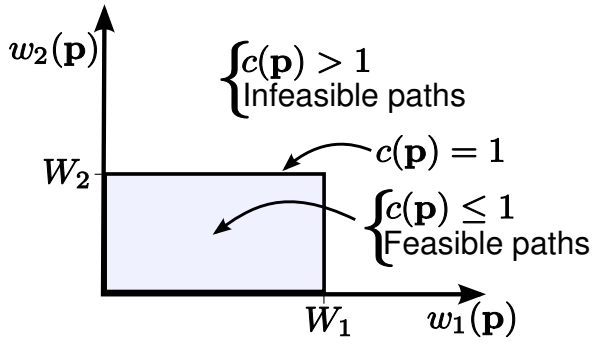

Figure 8.2: Values of the non-linear path length in Equation (8.3) and path feasibility

computations with various linear path lengths to solve the problem MCP and proposed novel linear path-length functions with improved performance. Despite interesting contributions on linear path lengths, this category of path-length functions suffers from a severe drawback: the least-cost path for a linear path-length function is not necessarily feasible, even if the network contains some feasible paths. Therefore, several recent studies, for instance [109, 183], have shown that certain *non-linear* path lengths provide better performance than any linear path-length function. Figure 8.1 illustrates this limitation of linear path lengths on a simple example. We consider the linear path length $c(\mathbf{p}) = w_1(\mathbf{p}) + w_2(\mathbf{p})$ and the constraints $\vec{W} = (5, 6)^T$. In our example, the least-cost path $\mathbf{p}^*$ has a cost $c(\mathbf{p}^*) = 6 + 1 = 7$, which is smaller than the costs $1 + 7 = 8$ and $4 + 5 = 9$ of the other paths. However, $w_2(\mathbf{p}^*) = 7$ is greater than $W_2 = 6$, thus, the least-cost path is infeasible, whereas the path with weights $(4, 5)^T$ is feasible.

8.2.3 Non-Linear Path-Length Functions

Many MCP algorithms use path-length functions, called *non-linear* path-length function, for which the equation (8.2.1) does not always hold. Korkmaz and Krunz [109] have shown through simulations that, in the general case, specific non-linear path-length functions outperform linear combinations of the path weights. Xue and Kami Makki [183] have later proved a similar result mathematically.

A well-known non-linear path-length function, described in [58, 169, 109], expresses the critical value $c(\mathbf{p})$ of the K considered additive metrics w_k for a path $\mathbf{p}$ subject to the constraints W_k for k in $[1..K]$. With these notations, the path-length function $c(\mathbf{p})$ follows the Equation (8.3).

$$c(\mathbf{p}) \equiv \max_{k \in [1..K]} \left(\frac{w_k(\mathbf{p})}{W_k} \right). \tag{8.3}$$

Intuitively, the expression for $c(\mathbf{p})$ in Equation (8.3) means that the path that is the furthest from violating the constraints is selected. Figure 8.2 represents the values of $c(\mathbf{p})$ for all the possible path-weight values: a path $\mathbf{p}$ is feasible if and only if $c(\mathbf{p}) \leq 1$.

Computing a least-cost path for a non-linear path-length function, such as the one in Equation (8.3), is difficult because the related least-cost path problem does not have an

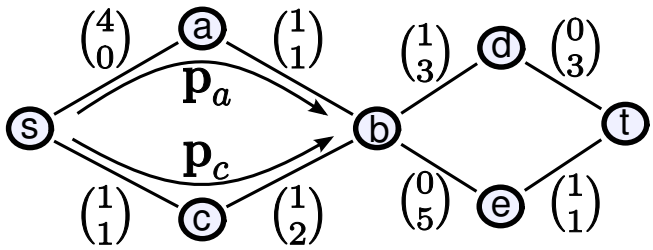

Figure 8.3: Absence of optimal sub-structure for the shortest path problems with a non-linear path-length function

optimal sub-structure[3]. For example, consider the network[4] in Figure 8.3 and a request for a path from s to t with the constraints $\vec{W} = (7, 8)^T$. There are two possible intermediate segments from s to b, namely $\mathbf{p}_a = (s, a)(a, b)$ and $\mathbf{p}_c = (s, c)(c, b)$. The segment $\mathbf{p}_c$ has the lowest value of the non-linear path-length function c defined in Equation (8.3). Its cost is $c(\mathbf{p}_c) = \max(\frac{2}{7}, \frac{3}{8}) = \frac{3}{8}$, whereas the cost of $\mathbf{p}_a$ is $c(\mathbf{p}_a) = \max(\frac{5}{7}, \frac{1}{8}) = \frac{5}{7}$. However, the least-cost paths $\mathbf{p}^* = (s, a)(a, b)(b, d)(d, t)$ and $\mathbf{p}^{*'} = (s, a)(a, b)(b, e)(e, t)$ from s to t, with cost $c(\mathbf{p}^*) = \max(\frac{6}{7}, \frac{7}{8}) = \frac{7}{8}$, uses the path segment $\mathbf{p}_a$, which does not have the lowest cost. The alternative paths $\mathbf{p} = (s, c)(c, b)(b, d)(d, t)$ and $\mathbf{p}' = (s, c)(c, b)(b, e)(e, t)$ that use the least-cost segment $\mathbf{p}_c$ are not feasible and have a larger cost $c(\mathbf{p}) = \max(\frac{3}{7}, \frac{9}{8}) \geq 1$.

Due to the absence of optimal sub-structure, it is not sufficient to consider a single shortest path segment to each intermediate node, as we do for a single *linear* path-length function. Therefore, usual shortest-path algorithms, which memorize a single shortest intermediate path per node, such as Dijkstra's algorithm, cannot solve the least-cost path problem associated with non-linear path lengths. As a result, several extensions of Dijkstra and Bellman-Ford algorithms have been described for the MCP problem. We describe these algorithms in the next sections.

8.3 Exact MCP Algorithms

Brute-force search is a general algorithm that consists in exhaustively enumerating all candidate solutions to a problem and checking whether each candidate satisfies the problem. The literature describes several brute-force algorithms for the MCP problem: usually, exact solutions[5] for the MCP problem consider all possible intermediate segments between a given source and a given target. Then, they combine these segments to form end-to-end paths. The problem of these approaches is that they often consider a prohibitive number of intermediate segments, and thus, their complexity is prohibitive.

The set of all candidate solutions to an optimization problem forms the *search-space* of the considered problem. Brute-force methods are often applied on a reduced search-space to restrict the enumeration to reasonable candidate solutions. In particular, exact MCP algorithms use four main search-space reduction methods, which have been listed by Van Mieghem

[3]The term optimal sub-structure is defined in Section 8.2.2 and in [50]

[4]To simplify the examples in the thesis, we typically consider only two link weights that take integer values, and the links are undirected.

[5]In this dissertation, we use the term *exact* to refer to the solutions that guarantee to find a feasible path if such path exists in the network.

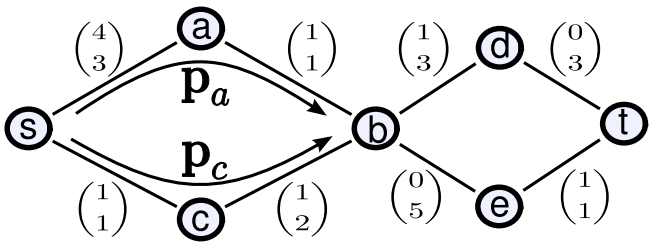

Figure 8.4: Dominated paths can be discarded

and Kuipers in [172]. They rely on simple lemmas that we describe in next paragraphs and prove in Appendix B.

The first search-space reduction method uses the fact that when several paths between the same nodes have the same weights, an algorithm can memorize only one of these paths without affecting the weights of the computed end-to-end solution. For example, in Figure 8.3, the paths $(b,d)(d,t)$ and $(b,e),(e,t)$ have the same weight vector $\vec{w} = (1,6)^T$. Thus, we can consider only the paths that use the segment $(b,d)(d,t)$, without losing the guarantee to find a path that is as far as possible from the request constraints.

Lemma 8.3.1 (Identical weights)**.** If two intermediate paths $\mathbf{p}_1 \in P_{a\to b}$ and $\mathbf{p}_2 \in P_{a\to b}$ have the same weights $\vec{w}(\mathbf{p}_1) = \vec{w}(\mathbf{p}_2)$, then an MCP algorithm can memorize only one of these intermediate paths without losing the guarantee to find an optimal end-to-end path.

Dominance is a widely-used concept, which expresses the idea that a solution is "better" than another, in the context of multi-objective optimization [49] and game theory [79]. Dominance is useful to reduce the search-space of MCP problems. A path $\mathbf{p}$ is *dominated* if there is a path $\mathbf{p'}$, *with the same source and target*, such that $w_k(\mathbf{p'}) \leq w_k(\mathbf{p})$ for all considered weights w_k, $k \in [1..K]$ and such that there is a k in $[1..k]$ for which $w_k(\mathbf{p'}) < w_k(\mathbf{p})$. For instance, consider the example in Figure 8.4. There are two paths $\mathbf{p}_a \in P_{a\to b}$ and $\mathbf{p}_c \in P_{a\to b}$ between the same nodes and such that $\vec{w}(\mathbf{p}_a) = (5,4)^T$ and $\vec{w}(\mathbf{p}_c) = (2,3)^T$; in this case, $\mathbf{p}_c$ dominates $\mathbf{p}_a$. If a path is not dominated by any path, we say that it is *non-dominated* or *Pareto optimal*. Given a source node s and a target node t, we call *Pareto frontier* the set of the non-dominated s-t-paths.

Lemma 8.3.2 (Dominated paths)**.** For any two nodes a and b in V, given two intermediate paths $\mathbf{p}_1 \in P_{a\to b}$ and $\mathbf{p}_2 \in P_{a\to b}$, if $\mathbf{p}_1$ dominates $\mathbf{p}_2$ then an MCP algorithm can memorize only $\mathbf{p}_1$ without losing the guarantee to find the optimal end-to-end path.

Lemma 8.3.2 implies that MCP algorithms can safely discard every path that contains a dominated segment, and thus, can memorize only non-dominated path segments, which significantly reduces the search-space for most instances of the MCP problem. For example, to solve the MCP problem exactly and efficiently, SAMCRA [169] and H_MCOP [109] consider only the candidate paths on the Pareto frontier.

We use Figure 8.5 to illustrate two additional methods for reducing the search-space. We study a request for a path from s to t subject to the constraints $\vec{W} = (7,15)^T$. As we consider non-negative additive link-weights, the paths that include a non-feasible segment are necessarily non-feasible. Therefore, MCP algorithms can safely exclude infeasible paths from the search-space. For instance, in Figure 8.5, the segment $\mathbf{s} = (a,d)(d,t)$ is infeasible: its weights $(8,9)^T$ infringe the constraint on the first metric. Therefore, any path that includes this segment is infeasible.

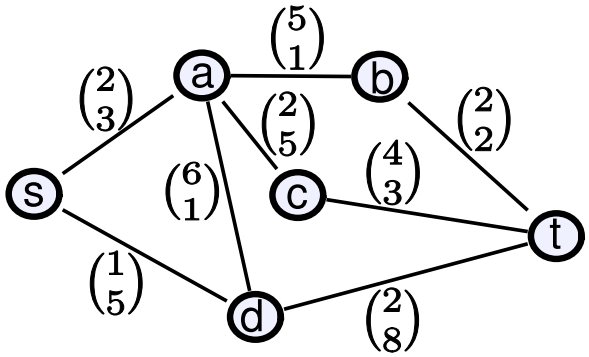

Figure 8.5: An instance of the MCP problem with two integer-valued metrics

Lemma 8.3.3 (Infeasible paths)**.** An MCP algorithm can memorize only feasible paths without losing the guarantee to find the optimal end-to-end path.

In Figure 8.5, consider the shortest path $(a, c)(c, t)$ from a to t for the weight w_1. This path provides the lower bounds $w^*_{1,a\to t} = 6$ on w_1 for any path between a and t. In addition, the shortest path from s to a for w_1 has a weight $w^*_{1,s\to a}$ equal to two. Therefore, any path $\mathbf{p}$ from s to t that traverses a has an end-to-end weight $w_1(\mathbf{p})$ that is larger than $w^*_{1,s\to a} + w^*_{1,a\to t} = 8$, and thus, infringes the constraints $\vec{W}$. Consequently, an MCP algorithm can safely discard any path going through a, which significantly simplifies the problem of finding an MCP from s tot t subject to $\vec{W} = (7, 15)^T$. Certain algorithms, such as H_MCOP [109] and SAMCRA [172], implement this idea. They compute a shortest path from every intermediate node to the target, for every single weight. Then, they use the weights of the computed shortest paths to compute lower bounds on the value of every metric for the end-to-end paths. This way, they avoid memorizing path segments that lead predictably to infeasible end-to-end paths. Van Mieghem *et alii* [172] name this approach *look-ahead*.

Lemma 8.3.4 (Predictably infeasible paths)**.** Given a constraint vector $\vec{W}$ and a shortest path segment $\mathbf{s}^* \in P_{a\to t}$ for a weight w_k. If a path $\mathbf{p} \in P_{s\to a}$ verifies $w_k(\mathbf{p}) > W_k - w_k(\mathbf{s}^*)$, then an MCP algorithm can discard $\mathbf{p}$ without losing the guarantee to find an optimal[6] end-to-end path.

8.4 Fast MCP Algorithms

Due to the computational intractability of the MCP problem, the time complexity of exact MCP algorithms grows exponentially in the worst-case. Thus, numerous approximate solutions, including provably good approximation algorithms and simple heuristics, have been proposed for the MCP problem. In this section, we present the underlying principles of the proposed approaches.

8.4.1 Use of Simple Shortest Path Algorithms

As explained in Section 8.2, a usual method to solve multi-criteria optimization problems, such as the MCP problem, consists in transforming them into mono-criterion optimization

[6]Here the term *optimal* means *the furthest below the constraints*.

problems. The idea behind this method is to enable the use of simple algorithms to solve the mono-criterion problem. Many popular methods for solving the MCP problem in an approximated manner apply this idea. The two main approaches determine a shortest path, considering (1) only one of the link weights or (2) a linear path cost function [183]. In this section, we describe several relevant examples of the two techniques. They offer a great advantage: they make it possible to use a simple and fast shortest-path algorithm, such as Dijkstra's algorithm, to determine approximate solutions to the MCP problem.

We first illustrate the approach that considers a single weight. An algorithm can compute a shortest path $\mathbf{p}_1^*$ for the first link weight w_1 in the hope that the value of the other metrics $w_2, \ldots, w_K$ for this path shall satisfy the constraints $w_k(\mathbf{p}_1^*) \leq W_k$ for k in $[2..K]$. If $w_1(\mathbf{p}_1^*)$ exceeds W_1, then for any path $\mathbf{p}$, we have $w_1(\mathbf{p}) \geq w_1(\mathbf{p}_1^*) > W_1$, by definition of a shortest path. Hence, the problem does not have any solution. To increase the probability that the shortest path computations considering a single weight provide a feasible solution to the MCP problem, an algorithm can compute shortest paths for all the metrics one by one, and then, assess if at least one of the computed paths is feasible [6].

The second approach applies shortest-path algorithms to optimize a path-length function. For example, Jaffe [98] describes an approximation algorithm that uses a linear path length to address the MCP problem with two constraints. His algorithm determines two positive multipliers d_1 and d_2, and optimizes the following path-length function with Dijkstra's shortest-path algorithm:

$$c : \mathbf{p} \rightarrow d_1 \cdot w_1(\mathbf{p}) + d_2 \cdot w_2(\mathbf{p}). \tag{8.4}$$

Andrew and Kusuma [6] extend Jaffe's algorithm to an arbitrary number of constraints. In addition, they show that the linear path length in Equation (8.5) provides a low blocking probability.

$$c(\mathbf{p}) = \frac{w_1}{W_1} + \frac{w_2}{W_2} \tag{8.5}$$

Jaffe's algorithm and related extensions suffer from the following important limitations, notwithstanding their interesting capability to use simple shortest-path algorithms. Xue and Kami Makki [183] show that the path-length function used by Jaffe's algorithm provides weaker guarantees on the performance of the computed MCPs than the path-length function in Equation (8.3). In addition, Van Mieghem and Kuipers note that the solutions to the least-cost path problem for the cost function in Equation (8.2) do not necessarily satisfy all constraints of the original MCP problem [172].

8.4.2 Quantization of the Metrics

One of the causes for the $\mathcal{NP}$-completeness of the MCP problem is that, in the general formulation of this problem, the path weights are real-valued, and thus, can take an infinite number of values. Therefore, many non-dominated paths to intermediate nodes and with different weights can exist. However, remember that according to Lemma 8.3.1, if several paths from the source node to the same intermediate node exist and have the same weights, then an MCP algorithm can memorize only one of the intermediate paths without losing the guarantee to find an optimal end-to-end path. This simplification can greatly reduce the complexity of the computations. Therefore, several papers quantize the link-weights so that they can take only discrete bounded values. This method permits bounding the number of intermediate paths that must be considered in the worst-case. In addition, the quantization translates the MCP problem into a simpler problem that can be solved in polynomial time [188] and whose solutions approximate the exact solutions of the MCP problem [185].

Chen and Nahrstedt [46] propose a heuristic algorithm based on metric quantization for the delay-cost-constrained routing problem. Their idea is to transform the second additive metric (cost) and the associated constraint to apply an extended Dijkstra's or an extended Bellman-Ford algorithm [50]. Thus, they scale and round the weights corresponding to the second metric so that the weights take integer values and that the related constraint becomes integer too. Yuan [188] generalizes this approach to the K-constraint problem. Yuan's limited granularity heuristic guarantees finding a feasible path if there exists a path between the considered source and target nodes, and whose weights w_k are all less than or equal to $(1-\epsilon)\cdot W_k$, where ϵ denotes a parameter of the heuristic. Song and Sahni [159] extend Yuan's algorithms [188] and describe novel approximation algorithms with improved performance. Finally, Xue *et alii* [185] describe the best-known *fully polynomial-time approximation schemes*[7] (FPTASs) based on the quantization of the weights w_k for k in $[2..K]$ for the MCP problem with K constraints. They propose the algorithm Fast-Dmcp that either finds a feasible path for the problem MCP or confirms that the network does not contain any s-t path with weights $\vec{w}$ such that $w_k \leq (1-\epsilon)\cdot W_k$ for all $k \in [2..K]$ and $w_1 \leq W_1$. According to Xue [185], the complexity of Fast-Dmcp for a network represented by a graph $G=(V,E)$ is in $\mathcal{O}\Big(|E|\Big(\frac{|V|}{\epsilon}\Big)^{K-1}\Big)$.

8.4.3 Bounding the Complexity of Brute-Force Search

Several heuristics for bounding the worst-case complexity of brute-force search mechanisms have been proposed in the literature. In this section, we detail the two main heuristics.

Tamcra [58, 59] and Yuan's limited-path heuristic [188] include in the path computation process only an arbitrarily limited number of paths to intermediate nodes. For example, a path computation algorithm that memorizes at most a single non-dominated path would consider only one path from s to d in Figure 8.5, even if there are two possible feasible non-dominated paths (s,d) and $(s,a)(a,d)$ with weights $(8,4)^T$ and $(1,5)^T$. The limitation of the number of considered intermediate paths is efficient to reduce the complexity of the MCP computation, but can lead to the exclusion of optimal paths. Ideally, the memorized intermediate path should be the most likely to lead the computation of end-to-end paths with good performance. In practice, the memorized path is often selected according to a path length function. For example, Tamcra uses the path-length function in Equation (8.3) and keeps the path with the lowest value for this function.

Many MCP heuristics base on the Bellman-Ford algorithm [50] because this algorithm explores the candidate paths from the source and *with an increasing number of hops*. This property can be used advantageously to reduce the worst-case complexity of MCP computations, by limiting the exploration to the paths that include a number of links lower than a predefined bound. This idea comes from the intuition that short paths are more likely to solve the MCP problem than longer paths.[8] For example, Xue *et alii* [185] use this technique in Fast-Dmcp, an efficient approximation algorithm for the MCP problem.

Algorithm	Category	Worst-Case Time Complexity	Reference
Jaffe's	Heuristic for MCP	$\mathcal{O}(V \log V + KE)$	[98]
TAMCRA	Heuristic for MCP	$\mathcal{O}(\alpha_{\max} V \log(\alpha_{\max} V) + \alpha_{\max}^2 KE)$	[58, 59, 110]
SAMCRA	Exact for MCP	$\mathcal{O}(\alpha V \log(\alpha V) + \alpha^2 KE)$	[172, 110]
H_MCOP	Heuristic for MCP	$\mathcal{O}(V \log V + KE)$	[109]
Yuan's	Approximation mechanisms for the MCP decision problem	$\mathcal{O}\left(EV\left(\frac{V}{\epsilon}\right)^{K-1}\right)$	[188]
FAST-DMCP	Approximation mechanisms for the MCP decision problem	$\mathcal{O}\left(E\left(\frac{V}{\epsilon}\right)^{K-1}\right)$	[185]

Table 8.1: Synthesis of relevant solutions to the intra-domain MCP problem

8.5 Synthesis of the Available MCP Algorithms

In Table 8.1, we provide a summary of relevant properties of the most well-known MCP algorithms. Many different problems are considered by the listed algorithms (*e.g.,* optimization version of the MCP problem, MCP problem with integer weights). Therefore, we classify all algorithms from the point of view of the solutions that they bring to the general MCP algorithm. We provide the worst-case time complexity of the algorithms: in the formulas, V and E are the number of nodes and of links in the considered network, K is the number of constraints. The notation ϵ describes an approximation factor; whereas, $\alpha_{\max}$ and α refer to specific parameters of TAMCRA and SAMCRA.

The main conclusions that can be drawn from Table 8.1, are the following. First, the worst-case complexity of exact MCP computation algorithms is non-polynomial[9] ($\mathcal{NP}$-completeness). Thus, *heuristics* and *approximation algorithms* must be studied; there is the following trade-off between these two categories of solutions. The complexity of the proposed heuristics is close to the complexity of Dijkstra's algorithm: $\mathcal{O}(|E| + |V| \log |V|)$ [13]. Approximation algorithms for the MCP problem have a larger complexity bound than the best heuristics. The advantage of approximation algorithms is that, by definition, they provide a solution that is quantifiably close to the exact solution.

8.6 Conclusion

Due to the important applications of the MCP problem for QoS routing and TE, many papers have proposed exact and approximation algorithms. We have highlighted the principles of these approaches. The essential idea is to convert a complex MCP problem into a simpler single-objective problem, the least-cost path problem. The tool to perform this conversion is a path-length function, which can be linear or not. We have provided a synthesis of exact

[7]This category of algorithms is defined in [50, p1021]. It represents elegant solutions to $\mathcal{NP}$-complete problems: fast and with guaranteed performance.

[8]However, note that there are straightforward examples for which this intuition does not hold.

[9]Because of the presence of α in the complexity bound

algorithms and exhibited the varied mechanisms that they use to reduce the search-space. In addition, we have explained the strategies adopted by fast MCP algorithms to reduce the problem complexity. Finally, we have given a short summary of relevant properties of existing solutions. The presented study of existing algorithms and of their principles enables us to design efficient solutions for the inter-domain MCP problem. We present these solutions in subsequent chapters.

Key points of Chapter 8

- Multi-objective problems are hard to solve: it is usually preferable to convert them to simpler single-objective problems. In the case of the MCP problem, path-length functions enable transforming the MCP problem into a simpler least-cost path problem.
- There are two categories of MCP solutions with different objectives: exact and fast methods. We have explained the fundamental ideas behind both types of techniques: these ideas can be re-used to solve the inter-domain MCP problem efficiently.

CHAPTER 9

A First Proposition: Finding the Best Paths

9.1 Introduction

We propose *inter-domain MCP* (ID-MCP), a novel distributed approach to compute inter-domain paths subject to an arbitrary number of constraints; ID-MCP solves the problem exactly. We have presented this approach in a paper at Eunice 2009 [22]. Our proposal relies on the three building blocks presented in Chapter 7.5: (1) the formulation of a per-domain problem, (2) an algorithm that solves the per-domain problem, and (3) a method for propagating and for combining the computation results to determine the end-to-end paths. The proposed algorithm is the first that guarantees to find the optimal inter-domain MCPs: the paths that ensure the best QoS performance with respect to the constraints, and thus, that provide the best resistance to changes in the QoS conditions.

We enhance ID-MCP to decrease the online computational load on the PCEs. The enhanced algorithm, called *pre-computation ID-MCP* (pID-MCP), splits the path computation operations into two stages. The first one involves the hardest computations and is performed offline. The second one consists in simple segment combination operations and is realized online. We have presented the enhanced algorithm in a book chapter [21].

The remainder of the present chapter is structured as follows. Section 9.2 details our distributed exact solution ID-MCP. In Section 9.3, we show that ID-MCP can be used in the PCE framework and in Section 9.4 we present the results of an extensive performance evaluation. Finally, Section 9.5 describes our pre-computation algorithm pID-MCP.

9.2 Proposition of an Online Distributed Exact Solution

Inter-domain path computations must be distributed among the traversed domains, so that all the domains control autonomously their internal paths. Therefore, ID-MCP defines a method to distribute end-to-end computation operations among the traversed domains. Moreover, it provides an algorithm to perform the per-domain calculations. The result of these per-domain computations is a set of paths that our algorithm will use to find a feasible end-to-end path. We use BRPC [176] to propagate the computation results and to enable the selection of optimal end-to-end paths in PCE-enabled network.

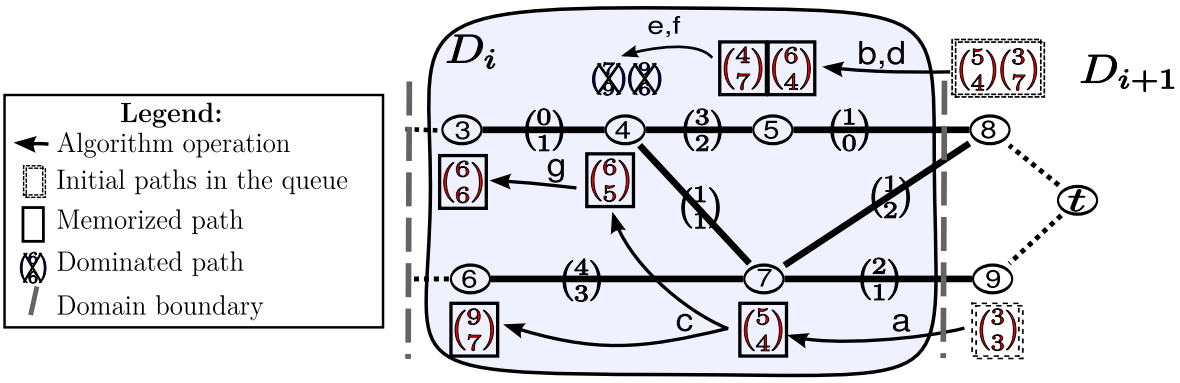

Figure 9.1: Operations of the proposed extended reverse Dijkstra's algorithm in a domain

Given a path computation request and the corresponding sequence of traversed domains, we compute all feasible non-dominated paths from the entry border-nodes (BNs) of every traversed domain to the target of the request. Then, we select the one that best fulfills the request constraints. This method is applicable to solve the problem InterMCP exactly if the network uses the BRPC procedure. It leads to the following per-domain problem for every traversed domain.

Problem (Per-Domain Problem)**.** *Given an instance of the problem InterMCP, a domain sequence* $\boldsymbol{S} = (D_1, D_2 \ldots)$*, a particular domain* D_i*,* $i \leq |\boldsymbol{S}|$ *in* $\boldsymbol{S}$ *and a set* $P^* \subset P_{D_{i+1} \to t}$ *of paths from the entry BNs of the next domain* D_{i+1} *to* t*, find all the non-dominated feasible paths from the entry BNs of* D_i *to* t *that have a suffix in* P^**.*

9.2.1 Solution to the Per-Domain Problem

The reverse Dijkstra's algorithm (RDA) [3, 72] computes shortest paths (considering a linear cost function) from several sources to a single target. For ID-MCP, we have designed a novel extended RDA that performs per-domain computations and memorizes all non-dominated feasible intermediate paths. The choice of extending Dijkstra's algorithm rather than the Bellman-Ford (BF) algorithm is based on several practical considerations. In particular, Van Mieghem *et alii* [171] have shown by simulations that MCP computation methods based on the BF algorithm require more computation time than methods, such as ours, that base on Dijkstra's algorithm.

Figure 9.2 gives the pseudo-code of ID-MCP. Moreover, Figure 9.1 depicts the operations of the proposed extended RDA in a specific domain that is denoted as D_i and for the constraints $w_1(\mathbf{p}) \leq 11$ and $w_2(\mathbf{p}) \leq 14$. The purpose of the operations is to find the non-dominated feasible paths from the entry BNs 3 and 6 of the domain D_i to the target $t \in D_{i+1}$ of a path computation request. To compute these paths, the algorithm uses a queue structure that contains the shortest paths from every intermediate node to t. This queue is initialized with paths from nodes 8 and 9, the entry-BNs of the neighboring downstream domain D_{i+1}, to the target t. In the example, the initial queue contains two different paths from 8 to t and a path from 9 to t. The calculation progresses from the right to the left of Figure 9.1.

The proposed extended RDA runs a loop. During each iteration of this loop, it picks among the paths of the queue a path $\mathbf{p}$ whose weights are the furthest from the constraints. In concrete terms, the algorithm selects the path is the one with the lowest value of the

1: **for all** domain i from D to 1 **do**
2: **if** $V_i = V_D$ **then**
3: queue ← {node:dest, predecessor:∅, w:0, status:not relaxed yet}
4: **else**
5: {queue, virtual_topology} ← concatenate(vspt$_{i+1}$, topology$_i$)
6: **end if**
7: **while** some elements of the queue have neither been relaxed nor been rejected yet **do**
8: element_min = extract_min(queue)
9: element_min.status ← relaxed
10: queue ← relax(element_min)
11: **end while**
12: vspt$_i$ ← extract_vspt(queue)
13: **end for**

Figure 9.2: Pseudo-code of ID-MCP

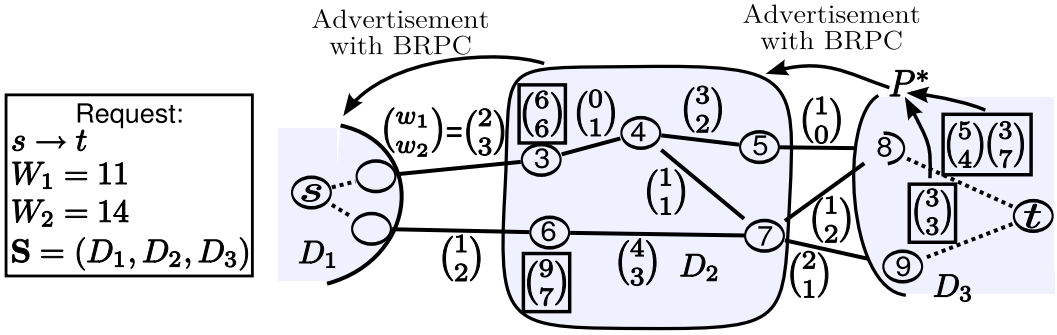

Figure 9.3: Advertisement of the computed paths with BRPC

path-length function $c(\mathbf{p})$ defined in Equation (8.3). In the example, the first selected path starts from node 9 and its weights are $(3,3)^T$. Then, the algorithm *relaxes* $\mathbf{p}$: it evaluates the weights of the paths from the neighboring nodes of the source of $\mathbf{p}$ (node 9) that have $\mathbf{p}$ as suffix. The algorithm adds the discovered paths to the queue if they are feasible and not dominated by any path in the queue. If a new path dominates one or more paths of the queue, then the procedure discards the dominated paths. The loop is repeated while the queue contains at least one element that has not been relaxed or discarded (operations b to g). Finally, the algorithm finds the feasible non-dominated paths from the entry BNs 3 and 6.

The correctness of the algorithm can be proved quite easily. During its operations, the algorithm does not discard any non-dominated feasible path. Thus, it guarantees to solve the per-domain problem defined in page 96. In addition, the termination is guaranteed: the algorithm relaxes one path during each iteration of the loop and the number of paths in a domain is finite.

9.2.2 Propagation of the Per-Domain Computation Results

Figure 9.3 continues the example in Figure 9.1 and illustrates the propagation of the per-domain computation results for a simple instance of the problem InterMCP[1]. The considered path computation request requires a path $\mathbf{p}$ from a node s in a domain D_1 to a node t in a domain D_3. Furthermore, the requested path $\mathbf{p}$ must satisfy the constraints $w_1(\mathbf{p}) \leq 11$ and $w_2(\mathbf{p}) \leq 14$. Finally, it must traverse the domain sequence $S = (D_1, D_2, D_3)$.

We use the BRPC procedure presented in Section 5.3.2 to forward the non-dominated feasible paths computed by every traversed domain to a PCE of the previous domain in the considered sequence of crossed domains [176]. In the example, to compute the requested path, the procedure determines the non-dominated feasible paths from the entry BNs (8 and 9) of the destination domain D_3 to t. Then, it forwards the set P^* of computed paths to the upstream domain D_2. The domain D_2 extends its local vision of the network topology with the paths in P^* and uses the information about these paths to compute the non-dominated feasible paths from its own entry-BNs (3 and 6) to t with our extended RDA. Then, D_2 advertises the paths that it has computed to the neighboring upstream domain (D_1). Finally, as D_1 is the source domain, it can compute an end-to-end feasible path.

9.2.3 Computational Complexity

We consider an instance of the problem InterMCP and define the following notations. We denote as α the maximum number of paths memorized by ID-MCP for a node to solve the considered problem[2]. K is the number of additive link metrics considered, D is the number of domains in the sequence considered, V and E are the maximum number of nodes and the maximum number of edges in a domain of the considered sequence.

Theorem (Worst-case time complexity)**.** *The worst-case time complexity of the algorithm ID-MCP is in* $\mathcal{O}(D\alpha^2 K(\alpha V + E + V^2))$.

Proof. The worst-case complexity of ID-MCP's operations inside a domain can be shown to follow the same expression as the one of SAMCRA which is in $\mathcal{O}(\alpha V \log(\alpha V) + \alpha^2 KE)$ according to [169], where α is the maximum number of paths memorized for a node in the queue, V is the number of nodes, K the number of additive link metrics considered and E the number of edges of the single domain considered. However, with ID-MCP the domains are extended by the contents of the VSPT. Thus, the number of nodes to consider is the number V_i of nodes of the domain plus the number of ingresses Ingr_{i+1} of the next domain V_{i+1} that are represented in the VSPT, plus the target. Moreover, the number of links to consider is the number E_i of edges of the domain, plus the number of inter-domain links between V_i and V_{i+1}, which is bounded by $V_i \cdot \mathrm{Ingr}_{i+1}$ and, in addition, the number of virtual edges extracted from the VSPT, which is bounded by $\alpha \cdot \mathrm{Ingr}_{i+1}$.

Hence, the worst-case complexity of the operations of ID-MCP inside a domain is in $\mathcal{O}(\alpha(V_i + \mathrm{Ingr}_{i+1} + 1) \cdot \log(\alpha(V_i + \mathrm{Ingr}_{i+1} + 1)) + \alpha^2 K(E_i + (\alpha + V_i) \cdot \mathrm{Ingr}_{i+1}))$. This expression can be simplified into $\mathcal{O}(\alpha V \cdot \log(\alpha V) + \alpha^2 K(\alpha V + E + V^2))$ by introducing the notations of the theorem and noting that $\mathrm{Ingr}_{i+1} \leq V$ and $V_i \leq V$. It can be further simplified into $\mathcal{O}(\alpha^2 K(\alpha V + E + V^2))$ by noticing that $\alpha V \log(\alpha V)$ is in $\mathcal{O}(\alpha^2 V^2)$. The operations are repeated in every domain along the considered sequence, thus, the complexity is multiplied

[1] The problem InterMCP is defined in page 78

[2] A bound for α is derived in Reference [169].

by the number D of domains crossed. In addition, we must take into account the operations for:

- the aggregation of the VSPT with the graph of the domain (at most $\alpha \cdot \text{Ingr}_{i+1}$ edges are added),
- the initialization of the queue (at most $\alpha \cdot \text{Ingr}_{i+1} + 1$ elements are added),
- and for the extraction of the final VSPT (at most $\alpha \cdot V_i$ elements of the queue are considered).

However, all these terms are in $\mathcal{O}(\alpha V)$, which leads to the complexity of the theorem. □

9.3 Integration of our Solution in the PCE Framework

9.3.1 Support of Requests with Multiple Metrics

The PCE protocol (PCEP), which is specified in Reference [175], can convey computation requests and replies that include several constraints. More precisely, the `METRIC` object allows a path computation client (PCC) to specify constraints on specific metrics. The `B` flag allows differentiating bound constraints (*e.g.,* the metric value must be lower than a specific threshold) from optimum constraints (*e.g.,* the metric value must be as low as possible). In addition, PCEP allows requests with bound constraints on various quantities, such as the IGP metric, the TE metric, and the hop-counts [175].

9.3.2 Support of the Per-Domain Algorithm

We need a way to specify in the PCEP request and reply messages that the PCEs must use our algorithm to compute the paths. Reference [113] defines an object to convey information on the objective function that the domains must use to select the shortest paths. We can define a specific objective function code to indicate that the domains should compute the non-dominated shortest paths with our algorithm.

In some cases, our algorithm requires that BRPC forwards several non-dominated feasible paths from the same entry BN to the previous domain. However, the original BRPC procedure uses a *virtual shortest-path tree* structure, that is defined in [176] and that contains typically a single path from every entry BN. Thus, our algorithm requires extending BRPC to allow the advertisement of several paths from the same BN. Note that this extension is purely conceptual: it does not put any additional requirements on PCEP.

9.3.3 Confidentiality Constraints

The BRPC procedure requires that path segments are transmitted to the upstream domain in a VSPT structure. However, supplying a path segment to a PCE of another domain may disclose confidential information about the internal topology. Therefore, we need a mechanism that enables the domains to exchange MCP segments without disclosing confidential details of their internal topology. For instance, we can use the method proposed by Bradford *et alii* in [33]. They have developed a mechanism to hide the contents of a path segment: they replace a confidential segment by a path-key that can be conveyed in the PCEP during the

computation operations. The path-key can be signaled in an ERO of RSVP-TE during the setup of the path [8].

9.4 Performance Evaluation

We evaluate the performance of ID-MCP by simulations. The purpose of our studies is to determine the quality of the computed paths and to analyze the complexity of the computations on practical examples.

9.4.1 Simulation Scenarios

By design, ID-MCP guarantees to find a path that satisfies the request constraints, if such a path exists. As ID-MCP computes all non-dominated feasible paths, it also enables selecting the path that provides the largest performance margin compared to the request constraints. This feature is important to maximize the chances that a computed path can be setup successfully, as the state of the network can change between the computation and the signaling of a path.

We evaluate the performance of our algorithm by simulations in topologies that represent extreme cases [111, 112]. More precisely, we consider the lattice domain topology in Figure C.1. We inter-connect the lattice domains to build a domain sequence along which we compute inter-domain MCPs. In LatticeFM (full-mesh, Figure C.2), every node of an intermediate domain is connected to every node in the next and in the previous domain of the sequence. As a result, the path diversity is extremely large. By contrast, in LatticeSL (single link, Figure C.3), only the bottom-right node of every domain is connected to the top-left node of the next-domain. We generate the values of two random uniformly distributed link-weights for every simulation run.

We consider the following path computation requests. The source is the top-left node of the first domain and the target is the bottom-right node of the last domain. We compare the outputs of ID-MCP for two different sets of constraints. First, we define *strict constraints* so that, in average, there is a feasible path for less than 70% of the simulated requests. Second, we define *loose constraints* so that there is a feasible path for every simulated request.

9.4.2 Performance Metrics

We evaluate the following performance metrics. The cost (C) is the lowest value of the function c defined below among the computed paths.

$$c(\mathbf{p}) = \max_{k \in [1..K]} \left(\frac{w_k(\mathbf{p})}{W_k} \right) \tag{9.1}$$

We define an additional function c' as:

$$c'(\mathbf{p}) = \mu_{k \in [1..K]} \left(\frac{w_k(\mathbf{p})}{W_k} \right). \tag{9.2}$$

In this expression, μ denotes the arithmetic mean operator. We call multi-dimensional cost (MC) the value $c'(\mathbf{p})$ for the end-to-end path $\mathbf{p}$ that has the lowest value of c' among the computed paths. MC helps evaluating the quality of the returned paths considering all metrics, whereas C indicates their performance margin with respect to the most restrictive metric. In

	C [%]		MC [%]		α		NP	
Constraints	*SL*	*FM*	*SL*	*FM*	*SL*	*FM*	*SL*	*FM*
Loose	19.2	13.9	18.7	11.4	10	7	7	3
Strict	89.3	72	86.5	60.1	8	2	5	1

Table 9.1: Performance of the exact algorithm ID-MCP with loose and strict constraints

accordance with the complexity formula for ID-MCP in Theorem 9.2.3, we derive an estimation of the time complexity from the measurement of the maximum number α of paths memorized in the computation queue for a single node. Finally, we denote the number of feasible non-dominated paths returned by ID-MCP as NP. This quantity provides an indication of the available path diversity.

9.4.3 Simulation Results

Table 9.1 presents the results of the simulations. We present additional simulation results in Appendix A and in a research report [19]. For loose constraints, the performance of the best paths returned by ID-MCP is much better than requested (C and MC are below 20%). This indicates that ID-MCP finds paths that will remain feasible even in case of variations in the network state. In addition, ID-MCP manages to find several feasible non-dominated paths (NP is greater than 3). By design, ID-MCP guarantees to find feasible paths even if the optimal performance of the network is close to the constraints. We can observe this behavior with strict constraints (C and MC are greater than 60%).

The complexity of the computations remains reasonable for the simulated scenario (α is less than 10). However, it would be prohibitive on larger topologies. There is a trade-off between the complexity of the calculations and the performance of the computed paths. In particular, our algorithm provides the theoretical basis for heuristics with a reduced complexity. We present such heuristics in subsequent chapters. Furthermore, in the next section, we present a pre-computation approach that enables the PCEs to perform only simple operations online.

9.5 Enhancement of ID-MCP with Pre-Computation

In [21], we have studied an enhancement of our exact approach to separate the computation operations in two phases. During the first phase, some intermediate results are pre-computed autonomously by every domain. During the second one, the pre-computed results are combined to obtain feasible end-to-end paths for specific path computation requests. The interest of this approach is to simplify the operations that are repeated for every request, and thus, to decrease the routers computational burden in case of frequent requests [192, 53, 54]. Recently, we have extended our study of pre-computation and we have compared the performance of various pre-computation algorithms, in [77, 78].

9.5.1 Per-Domain Formulation for Autonomous Computations

In our pre-computation approach, we postulate that the considered source and destination are two border nodes and that the considered domains agree on a set of constraints vectors for which they compute the path performance. Every constraint vector $\vec{W}$ represents a specific

class of service. We consider a network represented by an edge-weighted graph $G(V, E, \vec{w})$ and we assume that every destination network $N \subset V$ is represented by a single border node $t \in N$. With these assumptions, we define the following per-domain problem.

Problem (Autonomous Per-Domain Computations)**.** *Given a network represented by an edge-weighted graph $G(V, E, \vec{w})$, a specific domain $D \subset V$, a constraint vector $\vec{W}$, a target $t \in V$, find the non-dominated feasible paths from the entry BNs of $D \subset V$ to the entry BNs of any neighboring domain $D' \subset V$ of D.*

The per-domain problem enables every domain to compute an aggregate representation of its state. This representation describes the best possible performance that the domain can provide to neighboring domains for their transit traffic.

9.5.2 Solution to the Per-Domain Problem

To solve the aforementioned per-domain problem, we apply the following modifications on ID-MCP's per-domain algorithm.

- We change the initialization of the path computation queue. In the online case, the initial queue elements represent the paths advertised by the downstream domain and their weights. By contrast, in the autonomous case, the weights of the initial queue elements are equal to zero, to represent an empty path from each entry BN to itself.
- The computed MCPs have different targets too. In the online case, the algorithm always computes MCPs that end at the target t of the request. However, in the autonomous case, we are interested in MCPs that end at the entry BNs of the neighboring domains.
- Finally, the autonomous approach applies non-dominance comparisons for each target (each downstream entry BN) independently. For example, a path with weights $(4, 4)^T$ and target ingress A, does not dominate a path with weights $(5, 5)^T$ and target ingress B, attached to the same node.

Figure 9.4 depicts the operations for the resolution of the per-domain problem inside a particular domain with our extended RDA. These computation operations progress from the right to the left of the figure and provide the non-dominated path segments from each ingress of a domain, for example D_2, to the ingresses of the neighboring domains.

The initial queue elements represent empty paths with zero weights from each entry BN of the neighboring downstream domains toward the current domain. This enables independent operations inside every domain. For instance, in Figure 9.4, the computation starts with two elements representing empty paths from BN 8 and BN 9. The extended RDA relaxes the initial empty paths from the ingresses of the downstream neighboring domains to discover the non-dominated feasible paths from the neighboring nodes for a specific constraint vector. During this operation, the algorithm compares the memorized paths for each target ingress independently. For example, the path with the weights $(3, 3)^T$ to the target ingress 9, does not dominate the path with the weights $(4, 3)^T$ to the target ingress 8. The operations of the extended RDA end when all the memorized paths have been relaxed or discarded: at this moment, the algorithm has found all the non-dominated feasible paths from the ingresses of the domain to the ingresses of the neighboring downstream domains.

Theorem (Worst-case time complexity)**.** *We consider a domain that uses the autonomous approach to compute the paths from its entry BNs to the entry BNs of M neighboring domains.*

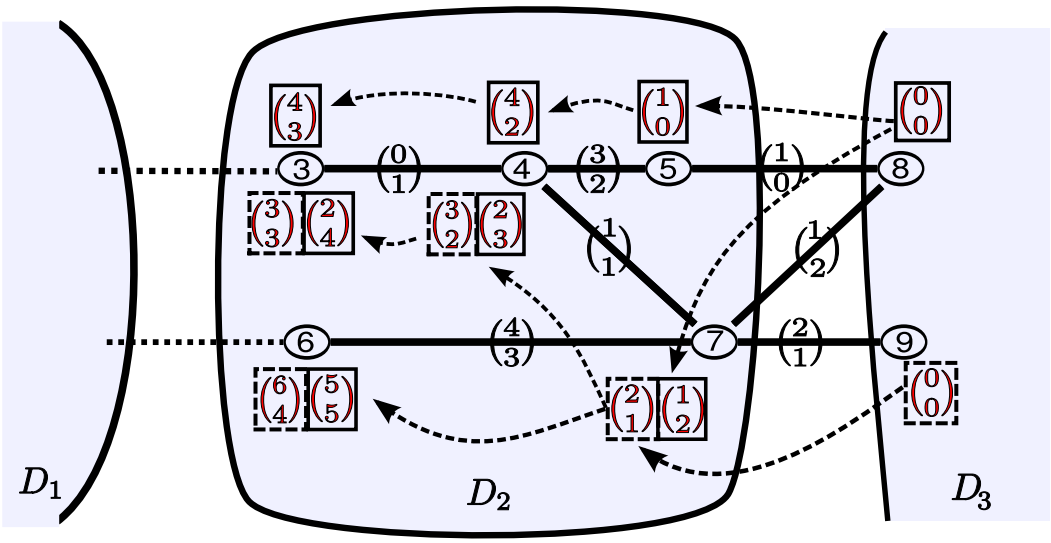

Figure 9.4: Operations of the extended RDA in a domain with the autonomous approach for the class of service with the constraints $W_1 = 11$ and $W_2 = 14$

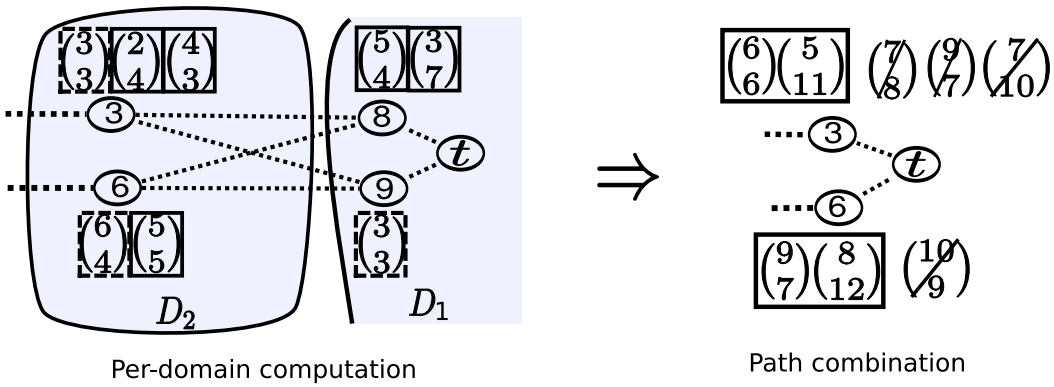

Figure 9.5: Propagation and combination of the per-domain results with the autonomous approach

Then, the worst-case complexity of the offline operations inside this domain is in $\mathcal{O}(\alpha^2 K(E + MN^2))$.

Proof. The proof is closely similar to the one of theorem 9.2.3, only the initial stage changes [19]. □

The offline operations of pID-MCP have a quite large time complexity because they solve an $\mathcal{NP}$-complete problem exactly. Therefore, approximation solutions and heuristics must be studied to perform faster offline operations.

9.5.3 Propagation and Combination of the Per-Domain Results

Figure 9.5 illustrates the operations for combining the results of the per-domain computations. When the domain D_2 receives the paths computed by a downstream neighboring domain D_3

for a specific request, it merges the results of its per-domain computations and the received paths. For instance, it combines the path with weights $(3,3)^T$ from node 3 to node 9 with the path from node 9 to the request's target t, which leads to a path with cost $(6,6)^T$ from node 3 to t. During the combination operations, the procedure discards the dominated paths. For example, the path with weights $(7,8)^T$ is dominated by the path $(6,6)^T$ from the same node.

Theorem (Worst-case time complexity)**.** *The worst-case complexity of the online operations inside every domain, for a specific path computation request, is in* $\mathcal{O}(\alpha^4 {N_{BN}}^3)$.

Proof. The combination operations rely on three phases: (1) adding the weights of the paths from the entry BNs of the considered domain and from the entry BNs of the downstream domain, (2) discarding the infeasible paths, and (3) comparing all the end-to-end paths to discard the dominated ones. The most computationally intensive operation comes from the path comparison operations. There are at most αN_{BN} paths from the entry BNs of the considered domain to every BN of the downstream domains. In addition, there are at most α paths from every entry BN of the downstream domain to the request's target t. The combination of these paths outputs at most $\alpha^2 N_{\mathrm{BN}}$ paths from the entry BNs of the considered domain to t. We must compare all these paths to one another to discard the dominated paths. The worst-case time complexity of this operation is in $\mathcal{O}(\alpha^4 {N_{\mathrm{BN}}}^2)$. Finally, the algorithm combines and compares the paths through every entry BN of the downstream domain, which leads to an overall worst-case time complexity in $\mathcal{O}(\alpha^4 {N_{\mathrm{BN}}}^3)$. □

For some configurations, the complexity of pID-MCP's online operations can be larger than the one of ID-MCP, which is provided in Theorem 9.5.2. In such scenarios, pre-computation is not interesting. However, the value of α is usually small (below 10). In addition, heuristic solutions can arbitrarily limit the value of α (see page 92). Consequently, the critical point that determines if the pre-computation decreases the online complexity is the number of entry border nodes. If this number is small (*e.g.,* $N_{\mathrm{BN}} < N^{\frac{2}{3}}$) compared to the number of nodes in the network, then, pre-computation can improve the computation speed.

9.6 Conclusion

In this chapter, we have studied the inter-domain MCP problem. The applications of this problem are more and more important with the recent advances in inter-domain traffic engineering. We have introduced ID-MCP, the first distributed exact solution to the considered problem for PCE networks. ID-MCP guarantees to find a feasible path if such a path exists in the network. Moreover, ID-MCP allows computing optimal end-to-end paths. These paths are the furthest from the constraints, thus, they ensure the best resistance to future variations of the QoS conditions. ID-MCP is based on distributed per-domain computations that are compatible with the PCE framework. Finally, we have described an enhancement of ID-MCP to support pre-computation. The enhanced algorithm, pID-MCP, performs the most complex operations offline and enables decreasing the online computational burden, in specific scenarios.

As the MCP problem is $\mathcal{NP}$-complete, the complexity of exact solutions is prohibitive for large topologies. Our solution provides a theoretical foundation from which we can derive heuristics with a reduced complexity. We present such heuristics in subsequent chapters.

Key points of Chapter 9

- In connection-oriented frameworks, the state of a network can change between the computation and the configuration of a traffic-engineering path. Therefore, it is interesting to find paths that are likely to remain feasible until they are configured.
- We propose a solution that finds the best inter-domain paths subject to a set of constraints. This solution, named ID-MCP, fulfills the security and autonomy requirements of the domains. We integrate our computation method in the PCE framework.
- We derive an alternative algorithm (pID-MCP) from our exact approach: this novel algorithm supports pre-computation and reduces the per-request computational complexity in specific situations.
- The proposed exact approaches require intensive computations, because the problem is $\mathcal{NP}$-complete. Consequently, approximate solutions and heuristics are required.

CHAPTER 10

Approximation Algorithms: Good Paths, Faster

10.1 Introduction

For a given optimization problem, we call *approximation algorithms* the methods that provably provide a close-to-optimum solution. By contrast to heuristics, such as TAMCRA [59], approximation algorithms have often a large asymptotic complexity (see Section 8.5). However, they provide more accurate performance guarantees. Here, we describe existing approximation algorithms for the MCP problem and show that they can be adapted for the inter-domain MCP problem. We introduce an inter-domain MCP algorithm that follows our three-stage structure: (1) formulating a per-domain problem, (2) solving it, and (3) combining the per-domain results. This novel method relies on a per-domain algorithm that is faster than exact solutions: the algorithm solves the per-domain problem in polynomial time and returns paths that are quantifiably close to the best solutions for the considered per-domain problem.

The remainder of the chapter is structured as follows. Section 10.2 presents related work on approximation algorithms for the MCP problem. In Section 10.3, we adapt the most efficient approximation technique to the inter-domain problem. Finally, we analyze the performance of the novel algorithm and show that faster methods (heuristics) are required in our context of application.

10.2 Approximation Methods for the MCP Problem

10.2.1 Objective of the Approximation

An MCP algorithm is a $(1+\epsilon)$-approximation if it fulfills the two following conditions: (1) it guarantees that the value of a specific objective function c for the returned path is at most $(1+\epsilon)$ times the optimum value of this function among the feasible paths, and (2) its running time is bounded by a polynomial in the encoding size of the problem instance [185, 82]. Fully polynomial-time approximation schemes (FPTASs) are methods that provide a $(1+\epsilon)$-approximation algorithm, whose time complexity is bounded by a polynomial in the encoding size of the problem instance and in $\frac{1}{\epsilon}$, for any positive value of $\epsilon \in \mathbb{R}$.

In concrete terms, approximation algorithms provide a solution that is as close as desired to the *optimum* for a given problem. In the context of the MCP problem, various definitions of optimality have been proposed [109, 185]. The most common one involves the definition of a path-length function, as explained in Section 8.4, and the computation of a path that simultaneously minimizes this function and fulfills a set of constraints. For example, this approach can lead to the following problem definition, in which we use the notations in Table 7.1.

Problem (MCP (optimization version))**.** *Given an instance of the MCP problem and a cost function* $c : P_{s\to t} \to \mathbb{R}$*, find a* feasible *path* $\boldsymbol{p}^* \in P_{s\to t}$ *such that any* feasible *path* $\boldsymbol{p} \in P_{s\to t}$ *verifies* $c(\boldsymbol{p}^*) \leq c(\boldsymbol{p})$*.*

10.2.2 Approximation Techniques

In Section 8.4.2, we have explained that quantizing the link metric values enables reducing significantly the complexity of the MCP problem. In fact, Yuan [188] shows that MCP problem instances with one real-valued and $K-1$ integer-valued link weights can be solved in polynomial time. Therefore, several researchers propose techniques to transform general MCP problem instances into specific ones with integer weights. For example, Fast-Dmcp [185] transforms the MCP decision problem into the Mcpp (Multi-Constrained Path with Positive rounding) problem that is defined below.

Problem (Mcpp$(G, s, t, K, \mathcal{D}, \mathcal{C}, \vec{w})$)**.** *We consider an MCP problem instance such that (1) for every link* $l \in E$*,* $w_1(l)$ *is a positive real number and* $w_k(l)$ *is a positive* integer *for* k *in* $[2..K]$*, and (2) the constraint* W_1 *takes a positive real value denoted as* $\mathcal{D}$*, whereas the constraints* $W_k,\ k \in [2..K]$ *are equal and verify* $W_k = \mathcal{C}$ *where* $\mathcal{C}$ *is a positive integer. Is there a feasible s-t path?*

The translation of an MCP problem into an Mcpp problem uses a parameter $\theta \in \mathbb{R}$, which determines the accuracy of the approximation. The translation consists in changing the original link weights $\vec{w}$ into:

$$\vec{w_\theta} = \left(w_1, \left\lfloor \frac{w_2 \cdot \theta}{W_2} \right\rfloor + 1, \ldots, \left\lfloor \frac{w_K \cdot \theta}{W_K} \right\rfloor + 1 \right)^T \tag{10.1}$$

and the path constraints $\vec{W}$ into:

$$\vec{W_\theta} = (W_1, \lfloor\theta\rfloor, \ldots, \lfloor\theta\rfloor)^T. \tag{10.2}$$

For instance, if $\theta = 10$, $\vec{w} = (3.11, 1.4, 2.23)^T$, and $\vec{W} = (11.01, 16.1, 15.02)^T$, then, the translation outputs the following values: $\vec{w_\theta} = (3.11, 1, 2)^T$ and $\vec{W_\theta} = (11.01, 10, 10)^T$. The interest of the translation is to reduce the weight granularity: this operation enables solving the Mcpp problem in polynomial time [188], instead of considering the original $\mathcal{NP}$-complete MCP problem.

The Mcpp problem and the original MCP problem are not equivalent. Their resolution can lead to different solutions. However, both problems are closely related: solving the Mcpp problem instead of the original MCP problem introduces a quantifiable performance loss that depends on the value of the translation parameter θ [119, 188, 185].

Several researchers [83, 119, 185] have described the same algorithm, which is depicted in Figure 10.1, to solve Mcpp problem instances in polynomial time. Xue *et alii* [185] name

1: # Initialization
2: **for all** $(c_2, \ldots, c_K) \in [0..\mathcal{C}]^{K-1}$ **do**
3: $d(t; c_2, \ldots, c_K) = 0$
4: $d(v; c_2, \ldots, c_K) = \infty$ and $\pi(v; c_2, \ldots, c_K) = \emptyset$ for all v in V such that $v \neq t$
5: **end for**
6: # Computations
7: **for all** $(c_2, \ldots, c_K) \in [0..\mathcal{C}]^{K-1}$, in increasing lexicographic order **do**
8: **for all** link (u, v) in E **do**
9: $b_k \equiv c_k - w_k(u, v)$
10: **if** $b_k \geq 0$ for all $2 \leq k \leq K$ and $d(v; c_2, \ldots, c_K) > d(u, b_2, \ldots, b_K) + w_1(u, v)$ **then**
11: $d(v, c_2, \ldots, c_K) \leftarrow d(u, b_2, \ldots, b_K) + w_1(u, v)$
12: $\pi(v, c_2, \ldots, c_K) \leftarrow u$
13: **end if**
14: **end for**
15: **end for**
16: # Extraction of the results
17: **if** $d(t, \mathcal{C}, \ldots, \mathcal{C}) > \mathcal{D}$ **then**
18: MCPP is infeasible
19: **else**
20: find the smallest integer $c \leq \mathcal{C}$ such that $d(t, c, \ldots, c) \leq \mathcal{D}$
21: return a feasible $s - t$ path π such that $w_1(\pi) \leq \mathcal{D}$ and $w_k(\pi) \leq c$ for $2 \leq k \leq K$
22: **end if**

Figure 10.1: Pseudo-code of PSEUDOMCPP [185]

this algorithm PSEUDOMCPP. For every node u in V and for every integer constraint values $c_2, \ldots, c_K$ with $0 \leq c_k \leq \mathcal{C},\ k \in [2..K]$, PSEUDOMCPP finds the path $\mathbf{p}^*_{c_2,\ldots,c_K}$ from s to u that minimizes the weight w_1 and that fulfills the constraints $w_k(\mathbf{p}^*_{c_2,\ldots,c_K}) \leq c_k,\ k \in [2..K]$. In the algorithm, the quantity $d(u, c_2, \ldots, c_K)$ denotes the value of the first weight $w_1(\mathbf{p}^*_{c_2,\ldots,c_K})$ for this path. The function π returns the predecessor of every node on the shortest path in terms of w_1, for given constraint values $(c_2, \ldots, c_K)$. If the MCPP problem admits some solutions, then, PSEUDOMCPP returns a feasible path that solves MCPP$(G, s, t, K, \mathcal{D}, c, \vec{w})$, where $c \leq \mathcal{C}$ is the smallest non-negative integer such that MCPP$(G, s, t, K, \mathcal{D}, c, \vec{w})$ is feasible [185]. In concrete terms, this means that PSEUDOMCPP returns a feasible path (p) whose weights $w_k(\mathbf{p})$ minimize $\max_{k\in[2..K]} \left(\frac{w_k(\mathbf{p})}{W_k}\right)$.

Previous work (*e.g.,* [185]) uses the PSEUDOMCPP algorithm to return a single $s - t$ path for a given source-destination pair. However, the operations of PSEUDOMCPP at lines 1 to 15 provide much more information: the algorithm maintains the shortest feasible path in terms of w_1 from the considered source s to any intermediate node and for every value of the constraint vector $(W_1, c_2, \ldots, c_K)$ with $(c_2, \ldots, c_K) \in [0..\mathcal{C}]^{K-1}$. In the next section, we will show that we can take advantage of this feature in the scope of inter-domain MCP computations.

Reducing the granularity of the link weights is probably the most-common approximation technique for approaching optimal MCPs. Nevertheless, the literature describes a few additional methods. For example, Xue *et alii* [183, 184] introduce a simple K-approximation algorithm that uses shortest-path algorithms (*e.g.,* Dijkstra's algorithm) to compute a shortest path for a single link weight function $w'(l)$ that they define as $w'(l) = \max_{k\in[1..K]} \left(\frac{w_k}{W_k}\right)$ for every link $l \in E$. They show that this simple method provides an approximation al-

gorithm for an optimization version of the MCP problem: the value of the cost function $c : \mathbf{p} \rightarrow \max_{k\in[1..K]} \left(\frac{w_k(\mathbf{p})}{W_k}\right)$ for the computed path is at most K times the minimum value of c for a path between the considered source and target. We think that the approximation provided by their method is too coarse: for example, with $K = 3$, the K-approximation can return a feasible path with cost eleven whereas there is a feasible path with cost four.

PseudoMcpp requires operations with a polynomial worst-case time complexity in $\mathcal{O}(|E| \cdot \mathcal{C}^{K-1})$ [185]. This worst-case time complexity is prohibitive in some situations: it increases rapidly when the number of links, the number of considered constraints, and the constraint values (*i.e.* the approximation accuracy) rise. In conclusion, we think that the complexity of approximation schemes represents a significant drawback of these methods when the network must determine a constrained path rapidly. Kuipers [110] arrives to the same conclusion and asserts that the complexity of approximation algorithms for the MCP problem is necessarily high compared to the complexity of simple heuristics. Because of this limitation of approximation algorithms, we must study heuristic solutions that enable faster computations.

10.3 Approximation for the InterMCP problem

10.3.1 Per-Domain Formulation

We try to adapt to the InterMCP problem the approximation techniques that have been developed for the MCP problem. We must first define a per-domain problem and our approximation objective for the end-to-end problem. In Chapter 7 to Chapter 9, we have explained that, to guarantee optimal end-to-end computations, we must compute all the non-dominated feasible paths from the entry-border nodes of every intermediate domain to the request's target. To reduce the complexity of these computations, we round and scale the link weights and the constraints. This leads us to the per-domain formulation detailed in the next paragraph.

We consider an instance of the problem InterMCP, a domain sequence $\mathbf{S} = (D_1, D_2 \ldots)$, a particular domain $D_i,\ i \leq |\mathbf{S}|$ in $\mathbf{S}$ and a set $P^* \subset P_{D_{i+1}\rightarrow t}$ of paths from the entry BNs of the next domain D_{i+1} to t. We assume that:

1. for every link l and for every path $\mathbf{p}$ in P^*, $w_1(l)$ and $w_1(\mathbf{p})$ are positive real numbers, $w_k(l)$ and $w_k(\mathbf{p})$ are positive *integers* for k in $[2..K]$, and
2. the constraint W_1 takes a positive real value denoted as $\mathcal{D}$, whereas the constraints $W_k,\ k \in [2..K]$ are equal and verify $W_k = \mathcal{C}$ where $\mathcal{C}$ is a positive integer.

With these assumptions, we define the per-domain problem as follows.

Problem (Per-Domain Problem: IdMcpp$(D_i, \vec{w}, \mathcal{D}, \mathcal{C}, P^*)$)**.** *For every constraint value $(c_2, \ldots, c_K) \in [0..\mathcal{C}]^{K-1}$, and for every entry BNs u of D_i, consider the u-t paths $\boldsymbol{p}$ that have a suffix in P^* and that verify $w_k(\boldsymbol{p}) \leq c_k$ for $k \in [2..K]$. Among these paths, find the one that has the lowest value of $w_1(\boldsymbol{p})$.*

We will show that this per-domain formulation enables us to find a feasible inter-domain path that enforces the first constraint W_1 and that minimizes $\max_{k\in[2..K]} \left(\frac{w_k(\mathbf{p})}{W_k}\right)$.

10.3.2 Per-Domain Algorithm

Pseudo-Code

1: # Initialization
2: **if** $t \in D_i$ **then**
3: **for all** $(c_2, \ldots, c_K) \in [0..\mathcal{C}]^{K-1}$ **do**
4: $d(t; c_2, \ldots, c_K) = 0$
5: $d(v; c_2, \ldots, c_K) = \infty$ and $\sigma(v; c_2, \ldots, c_K) = \emptyset$ for all v in V_i such that $v \neq t$
6: **end for**
7: **else**
8: **for all** $(c_2, \ldots, c_K) \in [0..\mathcal{C}]^{K-1}$ **do**
9: Initialize $d(v; c_2, \ldots, c_K)$ and $\sigma(v; c_2, \ldots, c_K)$ with the values advertised by D_{i+1} for all v in B_{i+1}
10: $d(v; c_2, \ldots, c_K) = \infty$ and $\sigma(v; c_2, \ldots, c_K) = \emptyset$ for all v in V_i
11: **end for**
12: **end if**
13: # Computations
14: **for all** $(c_2, \ldots, c_K) \in [0..\mathcal{C}]^{K-1}$, in increasing lexicographic order **do**
15: **for all** link (u, v) in E_i^+ **do**
16: $b_k \equiv c_k - w_k(u, v)$
17: **if** $b_k \geq 0$ for all $2 \leq k \leq K$ and $d(u; c_2, \ldots, c_K) > d(v, b_2, \ldots, b_K) + w_1(u, v)$ **then**
18: $d(u, c_2, \ldots, c_K) \leftarrow d(v, b_2, \ldots, b_K) + w_1(u, v)$
19: $\sigma(u, c_2, \ldots, c_K) \leftarrow v$
20: **end if**
21: **end for**
22: **end for**
23: # Extraction of the results
24: **if** s in D_i **then**
25: **if** $d(s, \mathcal{C}, \ldots, \mathcal{C}) > \mathcal{D}$ **then**
26: MCPP is infeasible for the s-t source-destination pair
27: **else**
28: find the smallest integer $c \leq \mathcal{C}$ such that $d(s, c, \ldots, c) \leq \mathcal{D}$
29: return a feasible $s - t$ path σ such that $w_1(\sigma) \leq \mathcal{D}$ and $w_k(\sigma) \leq c$ for $2 \leq k \leq K$
30: **end if**
31: **else**
32: **for all** u in B_i **do**
33: **if** $d(u, \mathcal{C}, \ldots, \mathcal{C}) > \mathcal{D}$ **then**
34: MCPP is infeasible for the u-t source-destination pair
35: **else**
36: return $d(u; c_2, \ldots, c_K)$ and $\sigma(u; c_2, \ldots, c_K)$ for all $(c_2, \ldots, c_K) \in [0..\mathcal{C}]^{K-1}$
37: **end if**
38: **end for**
39: **end if**

Figure 10.2: Pseudo-code of INTERMCPP

We design a novel algorithm, named INTERMCPP, to solve the per-domain problem IDMCPP. Figure 10.2 provides the pseudo-code of INTERMCPP. We represent the considered domain as an edge-weighted graph $D_i(V_i, E_i, \vec{w}_i)$. We denote the set of entry border nodes of the downstream domain D_{i+1} as B_{i+1}, the set $V_i \cup B_{i+1}$ as V_i^+, the set of inter-domain links from D_i to the downstream domain D_{i+1} as $E_{i \to i+1}$, and the set $E_i \cup E_{i \to i+1}$ as E_i^+.

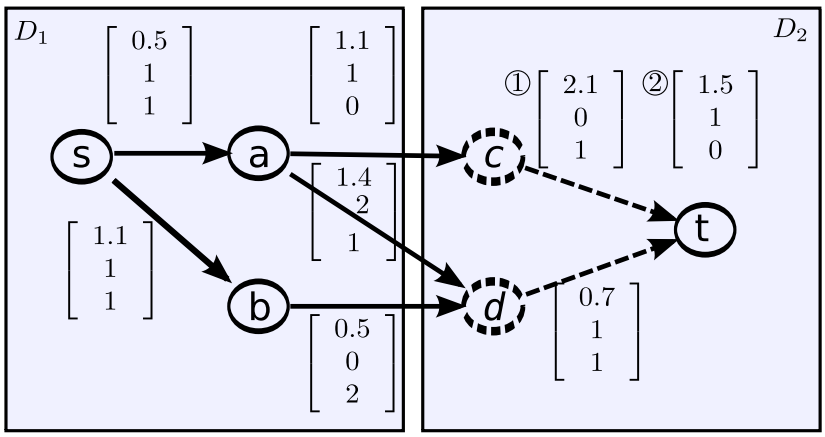

Figure 10.3: A scenario to illustrate the operations of InterMCPP

For the target domain, we consider that $E_i^+ = E$, $V_i^+ = V$, and $B_{i+1} = \emptyset$. We assume that the downstream domain D_{i+1} forwards the information $d(u; c_2, \ldots, c_K)$ and $\sigma(u; c_2, \ldots, c_K)$ for every entry border node $u \in B_{i+1}$ and for every $(c_2, \ldots, c_K) \in [0..\mathcal{C}]^{K-1}$. We do not study the structure that D_{i+1} uses to advertise this information.

InterMCPP brings significant enhancements compared to PseudoMCPP; nevertheless, it can be considered as an extension of PseudoMCPP. These two algorithms differ by the three following points. (1) *The initialization (line 1–12 in Figure 10.2):* InterMCPP needs to initialize the vector d to take into account the weights of the paths from the downstream domain. (2) *The extraction of the results (line 23–39):* InterMCPP must determine the non-dominated paths from the entry border nodes (or from the source node) to the request's target. By contrast, PseudoMCPP returns a single s-t path. (3) *The computations (line 13–22):* InterMCPP must find paths from several sources (the entry BNs) to a single target (the request's target) with the lowest possible complexity. However, PseudoMCPP computations provide information about the feasible paths from a single source s to every intermediate node u. It would be inefficient to run an algorithm such as PseudoMCPP one time from every entry border node; instead, we reverse the direction of the computations. This modification enables InterMCPP to compute the feasible paths from every intermediate node u to the target t in a single run. For every node u in V and for every integer constraint values $c_2, \ldots, c_K$ with $0 \leq c_k \leq \mathcal{C},\ k \in [2..K]$, InterMCPP finds the path $\mathbf{p}^*_{c_2,\ldots,c_K}$ from u to t that minimizes the weight w_1 and that fulfills the constraints $w_k(\mathbf{p}^*_{c_2,\ldots,c_K}) \leq c_k,\ k \in [2..K]$. In InterMCPP, $d(u; c_2, \ldots, c_K)$ represents the value of the first weight $w_1(\mathbf{p}^*_{c_2,\ldots,c_K})$ for this path. The function σ returns the *successor* of every node on the shortest path in terms of w_1, for given constraint values $(c_2, \ldots, c_K)$.

Example

The number of constraint value combinations at line 14 in the algorithm rises quickly with the value of the constraint $\mathcal{C}$ and of the number K of considered metrics. Therefore, we describe the operations of InterMCPP on a simple example with $\mathcal{C} = 3$, $K = 3$, and $\mathcal{D} = 3.5$. We consider the scenario in Figure 10.3 and we study the operations of InterMCPP in the domain D_1.

(c_2, c_3)	$d(c, c_2, c_3), \sigma(c, c_2, c_3)$	$d(d, c_2, c_3), \sigma(d, c_2, c_3)$
$(0,0)$	$\infty, \emptyset$	$\infty, \emptyset$
$(0,1)$	$2.1, t①$	$\infty, \emptyset$
$(0,2)$	$2.1, t①$	$\infty, \emptyset$
$(0,3)$	$2.1, t①$	$\infty, \emptyset$
$(1,0)$	$1.5, t②$	$\infty, \emptyset$
$(1,1)$	$1.5, t②$	$0.7, t$
$(1,2)$	$1.5, t②$	$0.7, t$
$(1,3)$	$1.5, t②$	$0.7, t$
$(2,0)$	$1.5, t②$	$\infty, \emptyset$
$(2,1)$	$1.5, t②$	$0.7, t$
$(2,2)$	$1.5, t②$	$0.7, t$
$(2,3)$	$1.5, t②$	$0.7, t$
$(3,0)$	$1.5, t②$	$\infty, \emptyset$
$(3,1)$	$1.5, t②$	$0.7, t$
$(3,2)$	$1.5, t②$	$0.7, t$
$(3,3)$	$1.5, t②$	$0.7, t$

Table 10.1: Data that the downstream domain D_2 transmits to D_1

In the figure, we use edges to represent the information (values of the functions d and σ) that the downstream domain D_2 has transmitted to D_1: in our example, there are two different paths, with different weights, from c to t. Table 10.1 describes the corresponding values that the downstream domain D_2 advertises for d and σ.

We run InterMcpp on our example scenario. We describe the computation operations in Table 10.2. Every column of the table presents the operations for a given edge. Every line of the table presents the operations for a given constraint value $(c_2, c_3) \in [0..3]^2$. In most situations, the conditions at line 17 of the algorithm are infringed: in this case, the table provides the infringement origin. When the conditions are fulfilled, the table provides the new values for the functions d and σ. At the end of the computations, InterMcpp returns a feasible s-t path σ such that $w_1(\sigma) \leq \mathcal{D}$ and $w_k(\sigma) \leq c$ for $2 \leq k \leq K$. In the example, c is equal to three with $d(s, 3, 3) = 3.1$, because $d(s, 2, 2) = 3.7 > \mathcal{D}$. InterMcpp examines the values of d and σ, and returns the end-to-end feasible path $s \to a \to c \to t②$, which corresponds to $s \to \sigma(s, 3, 3) \to \sigma(a, 2, 2) \to \sigma(c, 1, 2)$. During the path setup, the downstream domain D_2 will translate the virtual node $t②$ into the actual router-level path.

Analysis

Theorem. *The worst-case time complexity of* InterMcpp *for a domain* D_i *is in* $\mathcal{O}\left((|V_i| + |B_i| + |B_{i+1}| + |E_i^+|) \cdot (\mathcal{C}+1)^{K-1}\right)$.

Proof. The set $[0..\mathcal{C}]^{K-1}$ contains $(\mathcal{C}+1)^{K-1}$ elements. Therefore, the for loop at lines 3–6 in Figure 10.2 takes $\mathcal{O}\left(|V_i| \cdot (\mathcal{C}+1)^{K-1}\right)$ time. Moreover, the for loop at lines 8–11 takes $\mathcal{O}\left((|V_i| + |B_{i+1}|) \cdot (\mathcal{C}+1)^{K-1}\right)$ time. We set the convention that if $t \in V_i$, then, $|B_{i+1}| = 0$. With this convention, the initialization operations take $\mathcal{O}\left((|V_i| + |B_{i+1}|) \cdot (\mathcal{C}+1)^{K-1}\right)$ time for any domain of the considered sequence.

$\mathbf{c_2}, \mathbf{c_3}$	$\mathbf{s} \to \mathbf{a}$	$\mathbf{s} \to \mathbf{b}$	$\mathbf{a} \to \mathbf{c}$	$\mathbf{a} \to \mathbf{d}$	$\mathbf{b} \to \mathbf{d}$
$(0,0)$	$b_2 < 0$	$b_2 < 0$	$b_2 < 0$	$b_2 < 0$	$b_3 < 0$
$(0,1)$	$b_2 < 0$	$b_2 < 0$	$b_2 < 0$	$b_2 < 0$	$b_3 < 0$
$(0,2)$	$b_2 < 0$	$b_2 < 0$	$b_2 < 0$	$b_2 < 0$	$d(d, b_2, b_3) = \infty$
$(0,3)$	$b_2 < 0$	$b_2 < 0$	$b_2 < 0$	$b_2 < 0$	$d(d, b_2, b_3) = \infty$
$(1,0)$	$b_3 < 0$	$b_3 < 0$	$d(c, b_2, b_3) = \infty$	$b_2 < 0$	$b_3 < 0$
$(1,1)$	$d(a, b_2, b_3) = \infty$	$d(b, b_2, b_3) = \infty$	$d(a,1,1) = 3.2$, $\sigma(a,1,1) = c$	$b_2 < 0$	$b_3 < 0$
$(1,2)$	$d(a, b_2, b_3) = \infty$	$d(b, b_2, b_3) = \infty$	$d(a,1,2) = 3.2$, $\sigma(a,1,3) = c$	$b_2 < 0$	$b_3 < 0$
$(1,3)$	$d(a, b_2, b_3) = \infty$	$d(b, b_2, b_3) = \infty$	$d(a,1,3) = 3.2$, $\sigma(a,1,3) = c$	$b_2 < 0$	$d(b,1,3) = 1.2$, $\sigma(b,1,3) = d$
$(2,0)$	$b_3 < 0$	$b_3 < 0$	$d(a,2,0) = 2.6$, $\sigma(a,2,0) = c$	$b_3 < 0$	$b_3 < 0$
$(2,1)$	$d(a, b_2, b_3) = \infty$	$d(b, b_2, b_3) = \infty$	$d(a,2,1) = 2.6$, $\sigma(a,2,1) = c$	$d(d, b_2, b_3) = \infty$	$b_3 < 0$
$(2,2)$	$d(s,2,2) = 3.7$, $\sigma(s,2,2) = a$	$d(b, b_2, b_3) = \infty$	$d(a,2,2) = 2.6$, $\sigma(a,2,2) = c$	$d(d, b_2, b_3) = \infty$	$d(d, b_2, b_3) = \infty$
$(2,3)$	$d(s,2,3) = 3.7$, $\sigma(s,2,3) = a$	$d(b, b_2, b_3) = \infty$	$d(a,2,3) = 2.6$, $\sigma(a,2,3) = c$	$d(d, b_2, b_3) = \infty$	$d(b,2,3) = 1.2$, $\sigma(b,2,3) = d$
$(3,0)$	$b_3 < 0$	$b_3 < 0$	$d(a,3,0) = 2.6$, $\sigma(a,3,0) = c$	$b_3 < 0$	$b_3 < 0$
$(3,1)$	$d(s,3,1) = 3.1$, $\sigma(s,3,1) = a$	$d(b, b_2, b_3) = \infty$	$d(a,3,1) = 2.6$, $\sigma(a,3,1) = c$	$d(d, b_2, b_3) = \infty$	$b_3 < 0$
$(3,2)$	$d(s,3,2) = 3.1$, $\sigma(s,3,2) = a$	$d(b, b_2, b_3) = \infty$	$d(a,3,2) = 2.6$, $\sigma(a,3,2) = c$	$d(d, b_2, b_3) = \infty$	$d(d, b_2, b_3) = \infty$
$(3,3)$	$d(s,3,3) = 3.1$, $\sigma(s,3,3) = a$	$d(b, b_2, b_3) = \infty$	$d(a,3,3) = 2.6$, $\sigma(a,3,3) = c$	$d(a,3,3) = 2.1$, $\sigma(a,3,3) = d$	$d(b,3,3) = 1.2$, $\sigma(b,3,3) = d$

Table 10.2: Computation operations of InterMcpp in D_1

The for loops at lines 14–22 represent $|E_i^+| \cdot (\mathcal{C}+1)^{K-1}$ iterations of the operations at lines 16–20, which take constant time. Thus, the worst-case time complexity of the computations is in $\mathcal{O}\left(|E_i^+| \cdot (\mathcal{C}+1)^{K-1}\right)$.

In the source domain, the extraction of the results takes at most $\mathcal{O}(\mathcal{C})$ time to find the smallest c and $\mathcal{O}(n)$ time to extract the final path. In any other domain, the extraction of the results takes at most $\mathcal{O}\left(|B_i| \cdot (\mathcal{C}+1)^{K-1}\right)$ time to extract the values of d and σ for every entry BN and for every constraint value $c_2, \ldots, c_K$.

If we sum up the operations for the initialization, the computation, and the result extraction, we obtain a worst-case time complexity in $\mathcal{O}\left((|V_i| + |B_i| + |B_{i+1}| + |E_i^+|) \cdot (\mathcal{C}+1)^{K-1}\right)$. □

Theorem. *The algorithm* InterMcpp *solves the problem* IdMcpp.

Proof. The for loops at lines 14–22 of the algorithm compute, for every node $u \in V_i$ and for every constraint value $(c_2, \ldots, c_K) \in [0..\mathcal{C}]^{K-1}$, a u-t path $\mathbf{p}^*$ that fulfills the constraints $w_k(\mathbf{p}) \leq c_k$ for $k \in [2..K]$ and such that for every path $\mathbf{p}$ that fulfills these constraints

too, $w_1(\mathbf{p}^*) \leq w_1(\mathbf{p})$. In addition, the initialization phase at lines 2–12 guarantees that the computed path $\mathbf{p}^*$ has a suffix among the ones advertised by the downstream domain D_{i+1}. Therefore, InterMCPP solves the problem IdMCPP. □

10.3.3 End-to-End Approximation

General Algorithm

We name aIdMCP an algorithm in which the domains D_i from $D_{|S|}$ to D_1 translate the link weights, and then, run the algorithm InterMCPP. The algorithm takes a parameter $\theta \in \mathbb{R}$ as input and changes the original link weights $\vec{w}(l)$ for every link $l \in E_i^+$ into:

$$\vec{w_\theta} = \left(w_1, \left\lfloor \frac{w_2 \cdot \theta}{W_2} \right\rfloor + 1, \ldots, \left\lfloor \frac{w_K \cdot \theta}{W_K} \right\rfloor + 1\right)^T \tag{10.3}$$

and the path constraints $\vec{W}$ into:

$$\vec{W_\theta} = (W_1, \lfloor\theta\rfloor, \ldots, \lfloor\theta\rfloor)^T. \tag{10.4}$$

We explain the role of θ in the next section.

Approximation

We show that aIdMCP returns solutions that are as close as desired to the optimal for an optimization version of the InterMCP problem, in which we try to optimize the value of the function $c(\mathbf{p}) = \max_{k\in[2..K]} \left(\frac{w_k(\mathbf{p})}{W_k}\right)$. We can rephrase this sentence in a more intuitive manner: consider the problem of finding the inter-domain feasible path that minimizes $\max_{k\in[2..K]} \left(\frac{w_k(\mathbf{p})}{W_k}\right)$; the algorithm aIdMCP provides a solution that is as close as desired to the optimum of this problem.

We consider the original InterMCP problem and we denote as $\mathbf{p}^*$ the feasible s-t path that has the lowest value of the function $c(\mathbf{p}) = \max_{k\in[2..K]} \left(\frac{w_k(\mathbf{p})}{W_k}\right)$ among the feasible s-t paths for the InterMCP problem. Given a translation parameter θ, we denote as $\mathbf{p}^\theta$ the final path that the algorithm aIdMCP returns. The s-t path $\mathbf{p}^\theta$ verifies $w_1(\mathbf{p}^\theta) \leq \mathcal{D}$ and has the lowest value of the function $c_\theta(\mathbf{p}) = \max_{k\in[2..K]} \left(\frac{w_k^\theta(\mathbf{p})}{\mathcal{C}}\right)$ among the s-t paths $\mathbf{p}'$ that verify $w_1(\mathbf{p}') \leq \mathcal{D}$. We show that the distance between the performance of $\mathbf{p}^*$ and the performance of $\mathbf{p}^\theta$ is bounded and can be made as small as requested.

Lemma 10.3.1. $c(\mathbf{p}^\theta) - c(\mathbf{p}^*) \leq \frac{n-1}{\theta}$, where n denotes the total number of nodes in the considered network (*i.e.* in the graph that includes all the traversed domains).

Proof. We provide the proof of the lemma in Appendix B. □

The previous lemma proves that aIdMCP finds solutions that are as close as desired to the optimal solutions for the considered optimization version of the MCP problem, supposing that the value of θ is large enough. This is an interesting result, which enables the algorithm aIdMCP to guarantee that the computed MCPs for a given request have good performance.

We assume that we know a lower bound $\zeta > 0$ on the optimal value $c(\mathbf{p}^*)$. We consider a positive real number ϵ, and we define θ as $\theta = \frac{n-1}{\epsilon \cdot \zeta}$, where n is the total number of nodes in the considered network. With these assumptions, (B.8) indicates that:

$$\frac{c(\mathbf{p}^\theta)}{c(\mathbf{p}^*)} \leq \frac{|\mathbf{p}^*|}{\theta \cdot c(\mathbf{p}^*)} + 1. \tag{10.5}$$

We know that the length of any loop-free path $\mathbf{p}^*$ is bounded: $|\mathbf{p}^*| \leq n - 1$. In addition, we have defined ζ such that $\zeta \leq c(\mathbf{p}^*)$. Therefore, the inequality (10.5) leads to:

$$c(\mathbf{p}^\theta) \leq \frac{n-1}{\theta \cdot \zeta} + 1 \tag{10.6}$$

In (10.6), we can replace θ by its definition. Therefore, we obtain the final inequality below.

$$c(\mathbf{p}^\theta) \leq (1 + \epsilon) \cdot c(\mathbf{p}^*) \tag{10.7}$$

The inequalities 10.5 to 10.7 show that, to provide a $(1 + \epsilon)$-approximation algorithm *stricto sensu* for the considered optimization problem, it is sufficient to obtain a lower bound on the cost $c(\mathbf{p}^*)$ for the optimal path $\mathbf{p}^*$ and to know an upper bound on the total number of nodes in the network. We have not studied the methods to determine such bounds yet.

Worst-Case Time Complexity

We introduce the notations $V = \max_{i \in [1..D]}(V_i)$ and $E = \max_{i \in [1..D]}(E_i)$ to compute an asymptotic bound on the worst-case time complexity of aIDMCP.

Theorem. *The worst-case time complexity of a*IDMCP *is in* $\mathcal{O}\left(D \cdot E \cdot (\lfloor \theta \rfloor + 1)^{K-1}\right)$.

Proof. The weight translation operations take $\mathcal{O}\left(|E_{\text{tot}}| \cdot \mathcal{C}\right)$ time, where $|E_{\text{tot}}|$ is the total number of edges in the graph that includes all the traversed domains and their inter-connections. Every domain runs INTERMCPP, which takes $\mathcal{O}\left((|V_i| + |B_i| + |B_{i+1}| + |E_i^+|) \cdot (\mathcal{C} + 1)^{K-1}\right)$ time. The total time complexity of these operations for all the traversed domains is in $\mathcal{O}\left((3 \cdot |V_{\text{tot}}| + |E_{\text{tot}}|) \cdot (\lfloor \theta \rfloor + 1)^{K-1}\right)$, where $|V_{\text{tot}}|$ denotes the total number of nodes in the graph that includes all the traversed domains. With the notations E and V, the worst-case time complexity is in $\mathcal{O}\left(D \cdot (3 \cdot V + E) \cdot (\lfloor \theta \rfloor + 1)^{K-1}\right)$. We simplify this expression into $\mathcal{O}\left(D \cdot E \cdot (\lfloor \theta \rfloor + 1)^{K-1}\right)$, which proves the theorem. □

The algorithm aIDMCP ends in polynomial time. However, its complexity rises rapidly when the number K of constraints and the value of the parameter θ (the accuracy of the computations) increase. Therefore, we think that the cost of approximation schemes in terms of execution time hinders their application for inter-domain MCP computations.

10.4 Conclusion

A few approximation algorithms have been proposed for the MCP problem. The most common approximation technique consists in rounding and scaling the link weights to decrease their granularity. We have adapted this idea to solve the inter-domain routing problem more rapidly. We have described a per-domain problem that enables applying approximation techniques,

and an algorithm for solving this problem. Our inter-domain path computation method returns solutions that are quantifiably close to the optimum and its worst-case time complexity is polynomial, whereas the original problem is $\mathcal{NP}$-complete. Despite this significant complexity reduction compared to exact methods, we believe that the complexity of approximation mechanisms is prohibitive in practice. Therefore, in the next chapter, we study heuristic solutions, which do not bring provable performance guarantees but determine good solutions rapidly for most problem instances.

Key points of Chapter 10

- We have proposed aID-MCP, a solution to compute inter-domain MCPs, with good performance, in polynomial-time. This solution shows that it is possible to adapt MCP approximation techniques for efficient inter-domain computations.
- The worst-case time complexity of aID-MCP is polynomial, whereas the one of exact methods is significantly larger ($\mathcal{NP}$-complete problem). Nevertheless, the complexity of approximation methods is prohibitive in some scenarios; therefore, faster solutions (heuristics) must be studied.

CHAPTER 11

A Second Proposition: Finding Feasible Paths Rapidly

11.1 Introduction

Because of the complexity of the problem MCP, the complexity of exact and approximation methods for solving the inter-domain MCP problem becomes rapidly prohibitive when the size of the problem instances rises. Therefore, we derive a novel and efficient heuristic from our exact algorithms.

We present our heuristic, which is named kID-MCP, in Section 11.2. It can be used in various technological contexts. For QoS and TE, we integrate it in the PCE framework and describe the required protocol enhancements. The analytical evaluation described in Section 11.3 and the simulation study provided in Section 11.4 show that kID-MCP computes inter-domain MCPs with a reasonable complexity and without degrading the QoS guarantees.

11.2 Proposition of kID-MCP: a Fast Algorithm

In this section, we propose an efficient algorithm named kID-MCP that computes inter-domain paths subject to multiple constraints on independent additive metrics. We have previously presented this heuristic in a conference paper [23] and a research report [19].

11.2.1 Complexity Reduction Approach

We derive the proposed heuristic, kID-MCP, from the exact algorithm ID-MCP presented in Chapter 9. We know that the $\mathcal{NP}$-completeness of the MCP problem comes from the number of paths that must be considered for every intermediate node. Therefore, a method for bounding the complexity of MCP algorithms consists in arbitrary limiting the maximum number of paths that a path computation algorithm considers for every intermediate node.[1] We apply this method on ID-MCP to derive the proposed heuristic kID-MCP. In concrete terms, the considered per-domain problem and the method to propagate the per-domain computation results remain the same as for ID-MCP. However, we modify the algorithm

[1] Please refer to Section 7.3.3 and Section 8.4.3 for more details about these points.

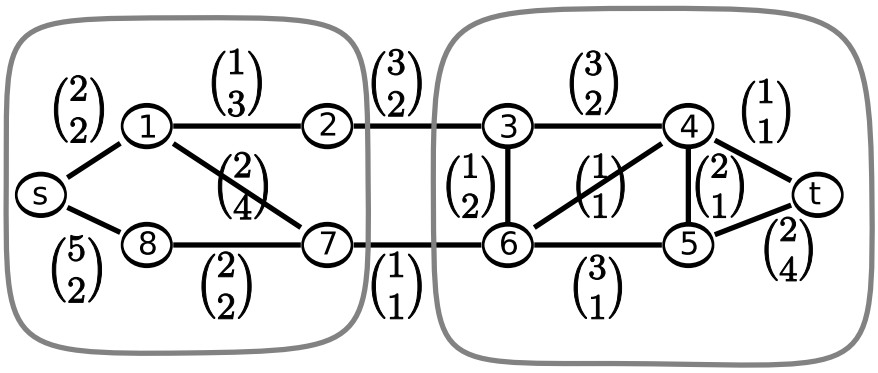

Figure 11.1: Example topology: we compute a path from s to t with the first and the second metric less than or equal to ten and to eight, respectively

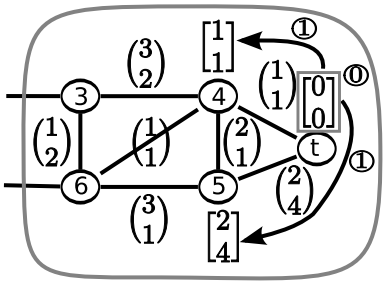

Figure 11.2: Initialization stage of the heuristic

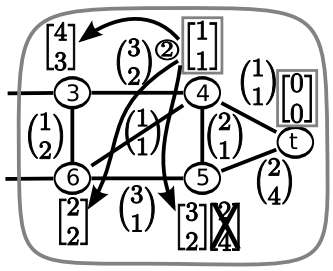

Figure 11.3: Relaxation of node 4 with the heuristic

that solves the per-domain problem: kID-MCP uses a heuristic with a bounded complexity, whereas ID-MCP uses a brute-force approach.

As kID-MCP memorizes only a subset of the non-dominated feasible paths, we need a criterion to select the memorized paths among the non-dominated feasible paths. We have observed that a greedy approach memorizing the paths that are the furthest from the constraints provides good results. In this chapter, we illustrate our approach with an algorithm that memorizes *at most one* MCP from every node to the target. Our studies show that this limitation provides good performance. Note nevertheless that extending kID-MCP to increase the limit and to memorize at most an arbitrary number $k > 1$ of paths for every intermediate node is straightforward. Such extension would augment the chances to find good solutions at the price of a higher computational complexity.

11.2.2 Example of kID-MCP Operations

We illustrate the operations of kID-MCP on a simple scenario before detailing the algorithm. Figure 11.1 depicts the example topology; we represent the values of the metrics for the links as follows: $\binom{a}{b}$, where a is the value of the first link metric and b the value of the second metric. In the example, kID-MCP computes a path between s and t with the following constraints: the value of the first and second metric for the end-to-end path must be less than or equal to ten and to eight, respectively.

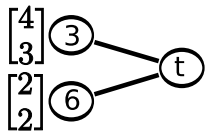

Figure 11.4: VSPT of the destination domain

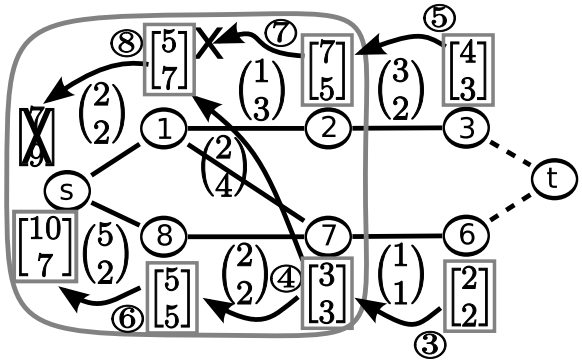

Figure 11.5: Operations of kID-MCP in the upstream domain

The path computation operations begin in the destination domain, as depicted in Figure 11.2. Their purpose is to calculate an MCP from every entry BN to the target t. During an initialization stage (stage ⓪), the node t is associated with the couple $\begin{bmatrix}0\\0\end{bmatrix}$, which represents the weights of an empty path from t to t. Then, the algorithm *relaxes* the initial path, during stage ①. This means that it computes the weights of the paths from neighboring nodes, 4 and 5, to the target t through the current node t. This operation discovers a path from node 4 to t with both metrics equal to one and another path from node 5 to t with the first metric equal to two and the second equal to four.

Then, the algorithm selects, among the discovered paths that have not been relaxed yet, the path with the lowest value of the path-length function $c(\mathbf{p})$ defined in Equation (8.3). In our example, the length of the path from node 4 is $\max(\frac{1}{10}, \frac{1}{8}) = \frac{1}{8}$, whereas the length of the path from node 5 is larger, with a value of $\max(\frac{2}{10}, \frac{4}{8}) = \frac{1}{2}$. Therefore, the algorithm relaxes the path from node 4, during stage ②, as depicted in Figure 11.3. During the relaxation operation, the algorithm discovers a path from node 5 to t through node 4 with a smaller length than the path that was already known:

$$\max\left(\frac{3}{10}, \frac{2}{8}\right) = \frac{3}{10} < \max\left(\frac{2}{10}, \frac{4}{8}\right) = \frac{1}{2}. \tag{11.1}$$

As a result, the algorithm memorizes the newly discovered path and discards the old path. The algorithm repeats the same operations as long as at least one discovered path has not been relaxed. In the example, this means that the algorithm relaxes paths from nodes 6, 5, and 3 successively. The relaxation of the path from node 6 leads to the discovery of two new paths, from node 3 and node 5. Both are immediately discarded, because their length is not smaller than the one of the paths from the same nodes that the algorithm already knows:

$$\begin{cases} \max\left(\frac{3}{10}, \frac{4}{8}\right) = \frac{1}{2} \geq \max\left(\frac{4}{10}, \frac{3}{8}\right) = \frac{2}{5} \\ \max\left(\frac{5}{10}, \frac{3}{8}\right) = \frac{1}{2} \geq \max\left(\frac{3}{10}, \frac{2}{8}\right) = \frac{3}{10}. \end{cases} \tag{11.2}$$

Similarly, the relaxation of the saved paths from node 5 and node 3 does not modify the paths memorized.

After completing the relaxation operations, the algorithm extracts a VSPT[2] and forwards it to the upstream domain, as represented in Figure 11.4. Then, the MCP computation

[2]Please refer to Section 5.3.2 for a presentation of the VSPT structure.

operations continue in the upstream domain, considering the virtual topology in Figure 11.5. The algorithm starts by selecting the path with the lowest path length $c(\mathbf{p})$ among the paths of the VSPT, and then, relaxes this path, during stage ③. Subsequently, the algorithm relaxes the discovered paths successively, during the stages ④ to ⑧. During these operations, the algorithm discovers a path from s to t through node 1 and immediately discards this path because the path weights infringe the constraints of the request. The value of the second metric, nine, is greater than the constraint eight. The procedure finally returns a single feasible end-to-end path: $s \rightarrow 8 \rightarrow 7 \rightarrow 6 \rightarrow -$, where $-$ denotes the nodes traversed in other domains. During the setup of this path, the traversed domains expand "$-$" into $4 \rightarrow t$, for example with the path-key mechanisms described in Section 9.3.3.

11.2.3 Formal Description of kID-MCP

Figure 11.6 provides a high-level description of kID-MCP. The operations of kID-MCP begin in the destination domain D and progress toward the source domain, as indicated in Figure 11.6, line 1. In each domain, the operations are similar to the ones of ID-MCP, but with an arbitrary limitation on the number of memorized paths and a selection criterion for the memorized paths.

The path computation operations rely on a queue whose elements describe MCPs from intermediate nodes to the target t of the considered request. Any queue element $\mathbf{q}$ contains the following information:

- the considered node ($\mathbf{q}$.node),
- the next-hop toward t ($\mathbf{q}$.predecessor),
- the values of the metrics for the path from $\mathbf{q}$.node to t ($\mathbf{q}.w$), and
- the state of this path ($\mathbf{q}$.mark).

The next-hop ($\mathbf{q}$.predecessor) of an element $\mathbf{q}$ is the node whose relaxation leads to the discovery of $\mathbf{q}$, hence, we call it the *predecessor* of $\mathbf{q}$. Any queue element is marked as "not relaxed yet" or "relaxed."

In the destination domain, kID-MCP initializes the queue with a zero-weight path to t (line 3). In any other domain, it initializes the queue with paths from the leaves of the received VSPT and to the target t (line 5). The operations of kID-MCP in each traversed domain rely on a loop, described in lines 6 to 9, that stops as soon as the queue does not contain any element marked as "not relaxed yet".[3] During each loop iteration, the path length of each queue element $\mathbf{q}$ is considered. The algorithm computes this path length from the metric values ($\mathbf{q}.w$). In addition, the algorithm selects and relaxes the element of the queue with the lowest path length among the queue elements that have not been relaxed yet. When the loop ends, the algorithm extracts a VSPT from the queue (line 10). In the source domain, the VSPT includes the elements of the queue that correspond to paths from the source node s. In any other domain, the VSPT includes the elements of the queue that correspond to paths from the entry BNs to the target t.

Figure 11.7 details the operations of the function that builds the domain virtual topologies and that initializes the path computation queue. This function determines the entry BN that

[3] or one relaxed path is associated with every entry BN

1: **for all** domain i from D to 1 **do**
2: **if** $i = D$ **then**
3: queue ← {node:t, predecessor:$\emptyset$, w:0, mark:not relaxed yet}
4: **else**
5: {queue, virtual_topology} ← concatenate(VSPT$_{i+1}$, topology$_i$, interdom_TE_links$_{i\to i+1}$)
6: **end if**
7: **while** the queue contains one or more elements marked as "not relaxed yet" **do**
8: element_min = extract_min(queue)
9: element_min.mark ← relaxed
10: queue ← relax(element_min, virtual_topology, queue)
11: **end while**
12: VSPT$_i$ ← extract_VSPT(queue)
13: **end for**

Figure 11.6: High-level description of kID-MCP

1: **for all** Leaf v of VSPT$_{i+1}$ **do**
2: virtual_topology ← connect_to_virtual_topology(v, topology$_i$, interdom_TE_links$_{i\to i+1}$, VSPT$_{i+1}$)
3: queue ← {node:v, predecessor:t, w:value of the metrics for $v \to t$ in VSPT$_{i+1}$, mark:not relaxed yet}
4: **end for**
5: **return** queue, virtual_topology

Figure 11.7: Description of "concatenate", the function building the virtual topology

corresponds to each leaf of the VSPT. Then, it extends the virtual topology of the considered domain to include this entry BN, as shown in line 3. The function also adds into the queue an element that describes the path of the VSPT from this entry BN to t (line 4). Finally, the function returns the extended topology and the initial queue.

Figure 11.8 details the operations of the relaxation function. This function visits every neighbor v of the relaxed element m, except its predecessor, as represented in lines 1 to 8. It evaluates the considered metrics for each newly discovered path $\mathbf{p}_v$ from v to t through m. This implies that the relaxation function adds the metric values for the link $v \to m$ and for the path from m to t, as shown in line 2. If there is already a path $\mathbf{q}$ from v to t in the queue and $\mathbf{q}$ has not been relaxed yet (line 3), then there are two possibilities: either the length of the new path $\mathbf{p}_v$ is strictly smaller (line 4), then $\mathbf{p}_v$ replaces the old path $\mathbf{q}$ (line 5), or, the length of the new path $\mathbf{p}_v$ is not strictly better, then the new path is discarded. If there is no path from v to t in the queue and the new path $\mathbf{p}_v$ is feasible (line 7), then the algorithm adds $\mathbf{p}_v$ to the queue (line 8), else, the algorithm discards $\mathbf{p}_v$. Finally, the function returns the queue.

The path memorization criterion, depicted in Figure 11.8 at line 4, takes only the length $c(\mathbf{p})$ of the paths into account. In concrete terms, according to Equation (8.3), this means that the memorized paths are selected depending on the metric whose value is the closest to the constraints. If the algorithm discovers several paths with the same length and from the same node successively, the first path is memorized and the others are discarded. More

```
1: for all neighbor v of m=element_min.node (except element_min.predecessor) in virtual_topology do
2:   w_v = w_{v→m}+ element_min.w
3:   if queue contains already a path q from the node v then
4:     if c(p_v) < c(q) and q.mark="not relaxed yet" then
5:       replace q by p_v in queue
6:     end if
7:   else
8:     if c(p_v) ≤ 1 then
9:       add {node:v, predecessor:m, w:w_v, mark:not relaxed yet} to queue
10:    end if
11:  end if
12: end for
13: return queue
```

Figure 11.8: Description of "relax", the relaxation function

sophisticated criteria can be implemented. However, the presentation of these criteria would complicate the description of the proposed algorithm.

11.3 Analytical Evaluation of kID-MCP

11.3.1 Termination and Correctness

In the present section, we prove that kID-MCP terminates and computes feasible inter-domain paths.

Theorem. *kID-MCP terminates.*

Proof. kID-MCP marks one more path as "relaxed" during each iteration of its main loop (Figure 11.6, line 8). As the number of paths in every domain is finite and as kID-MCP stops when all paths have been marked as "relaxed" or before, kID-MCP terminates. □

Theorem. *kID-MCP computes feasible inter-domain paths.*

Proof. The proof is immediate: infeasible paths are discarded in Figure 11.8 at lines 7–8. □

There is a main difference between ID-MCP, the exact solution presented in Chapter 9, and kID-MCP, the solution described in the present chapter. Because of its complexity reduction mechanisms, kID-MCP does not provide any guarantee to find a feasible path when such paths exist. In addition, kID-MCP does not guarantee to compute the path that is the furthest from the request constraints. On the contrary, exact solutions systematically find a feasible path when such paths exist and they can compute the path that is the furthest from the request constraints.

11.3.2 Scalability

In this section, we evaluate the computational complexity of the proposed algorithm. We denote the number of domains in the considered sequence $\mathbf{S}$ of traversed domains as $D = |\mathbf{S}|$ and the number of additive metrics as K. In addition, N_{BN} and N represent the maximum number of BNs and the maximum number of nodes in any domain of the sequence.

Theorem. *The worst-case complexity of kID-MCP is in* $\mathcal{O}(D \cdot K \cdot N^2)$.

Proof. We first compute the worst-case complexity of the operations of kID-MCP inside a domain. The main contribution to this worst-case complexity comes from the operations of the main loop (Figure 11.6, lines 6–9). During each iteration of the main loop, the algorithm marks one more queue element as "relaxed." In addition, the queue contains at most one single path from every node; that is, at most N elements. Thus, the algorithm repeats the operations of the loop at most $\mathcal{O}(N)$ times for each domain. The extraction from the queue, which contains N elements or less, of the element with the minimum length has a worst-case complexity in $\mathcal{O}(\log N)$, if the queue is implemented as a Fibonacci heap [76]. During the relaxation of an element, the algorithm visits at most $N-1$ neighboring nodes and for each of these neighboring nodes it computes at most one path length. The complexity of the computation of a path-length value depends on the number K of considered metrics; it is in $\mathcal{O}(K)$. Thus, the relaxation operations take $\mathcal{O}(K \cdot N)$. As a result, the contribution of the main loop to the worst-case complexity of kID-MCP is in $\mathcal{O}(N \cdot (\log N + K \cdot N)) = \mathcal{O}(K \cdot N^2)$.

The algorithm repeats the path computation operations in every domain along the sequence of traversed domains. Thus, the complexity is multiplied by the number D of domains crossed. In addition, we must consider the operations for the aggregation of the VSPT with the graph of the domain, for the initialization of the queue, and for the extraction of the final VSPT. Nevertheless, all these terms are in $\mathcal{O}(K \cdot N)$, which leads to the complexity of the theorem. □

The worst-case complexity of kID-MCP is quadratic[4] in N, which seems reasonable considering that the MCP problem is $\mathcal{NP}$-complete. The introduced amount of signaling traffic is another important aspect of the scalability of an inter-domain MCP algorithm. The proposed algorithm transmits at most one single path from every entry BN to the upstream domain. Therefore, the worst-case signaling overhead introduced by kID-MCP is in the same order as the one of the BRPC procedure presented in Reference [176].

11.4 Evaluation by Simulation

In Reference [19], we present a thorough evaluation of the performance of kID-MCP in numerous simulation scenarios and considering many performance criteria. For readability and brevity, we present here a summarized analysis with the most important results.

11.4.1 Simulation Scenarios

To limit the computational complexity, kID-MCP memorizes at most one single path from each node. In some cases, this limitation forbids finding a feasible path when there is one in the network. Consequently, we determine if kID-MCP finds a solution fulfilling the constraints

[4] K and D are usually small compared to N

when such feasible solutions exist in the network. In small topologies, a brute-force algorithm, such as ID-MCP, can determine if a solution exists and can calculate the optimal path.

We describe simulation results on four topologies. We name the first topology REAL8; it is described by Liljenstam *et alii* [118] and represents the network in the USA of major operators. This topology is too large to compute the optimal solutions exactly: we use it to present an example of application of kID-MCP in the inter-AS case. We consider two ASes A and B made up of approximately 1000 nodes and 3500 undirected links. They are inter-connected through eleven undirected links. We generate twenty requests with s in A and t in B. We assume that the traffic between these ASes follows the direct domain-sequence A-B. The requests include a constraint of 100 ms for the one-way speed-of-light delay. This value seems reasonable for voice traffic, for example. In addition, the requests specify that the paths must contain at most 30 hops. The number of hops represents a TE metric equal to one for every link, which is used by operators to minimize the amount of resources used by each request. These constraints are expected to be relatively high compared to the optimal performance of the network.

We compare the paths computed by kID-MCP to the optimal ones in the three following topologies. SYM-CORE is a topology used by Dasgupta *et alii* [56] to represent an inter-area case. In addition, we have inter-connected three[5] identical lattices, represented in Figure C.1, to create inter-domain topologies in which we test the performance of kID-MCP in extreme cases [111, 112]. In LATTICEFM (full-mesh), every node of an intermediate domain is connected to every node in the next and in the previous domain of the chain. In LATTICESL (single link), only the bottom-right node of every domain is connected to the top-left node of the next-domain.

In Section 7.3.4, we have explained that the "strictness" of the request constraints affects the complexity of the problem. In SYM-CORE, LATTICEFM, and LATTICESL, we generate the values of two random uniformly distributed link-metrics and we define request constraints so that, in average, there is a feasible path for less than 70% of the requests simulated. This enables us to observe some cases in which there is a feasible path but kID-MCP cannot find it. In SYM-CORE, we select a source s and a target t randomly in different areas and we compute an MCP connecting them. In lattice topologies, s is the top-left node of the first domain and t is the bottom-right node of the last domain.

11.4.2 Performance Criteria

In the evaluation of ID-MCP, we focused on the available path diversity and on the complexity of the operations. In the present chapter, we have different concerns: we must assess the quality of the paths that our fast heuristic provides. Therefore, we evaluate the following performance criteria. The *absolute success rate* (ASR) is the percentage of simulated requests for which kID-MCP finds a feasible path when one exists. The ASR cannot be computed in REAL8, because this topology is too large. Consequently, in REAL8, we consider the *success rate* (SR): the percentage of simulated requests for which an algorithm finds a feasible path. Finally, we consider the *cost* (C): the value of the path-length function $c(\mathbf{p})$ defined in Equation (8.3) for the best path computed by an algorithm. As this value is bounded by zero and one, we express it as a percentage.

[5] Bu and Towsley have shown that the average domain sequence length in the Internet is around three domains [35].

Topology	SR [%]	C [%]
Real8	100	12
Topology	**ASR [%]**	**C [%]**
Sym-Core	93	76 (optimal: 75)
LatticeFM	100	72 (optimal: 72)
LatticeSL	97	90 (optimal: 89)

Table 11.1: Performance of the heuristic kID-MCP in various topologies

The cost (C) of the optimal path is considered only for the requests for which kID-MCP finds a feasible path. We compute confidence intervals for all performance criteria. The simulations in Real8 represent an application example for a few requests, whereas the results in Sym-Core and in lattice topologies are statistically significant.

11.4.3 Simulation Results

Table 11.1 presents the results of the simulations. We present additional simulation results in Appendix A and in a research report [19]. For the scenario considering Real8, the topology of two large networks in the USA, kID-MCP can find a feasible path for every simulated request: SR=100% (so, in this case ASR=100%). The cost (C) of the computed MCPs is 12% in average. This indicates that the value of the considered metrics for the computed paths is far below the constraints: as expected the network would be able to support services with more stringent requirements.

In Sym-Core, kID-MCP finds a solution in 93% of the cases when there is a feasible path, which seems quite good considering that the algorithm complexity is reduced. We expected that the performance of the paths computed by kID-MCP would significantly differ from the optimal because the complexity of kID-MCP is tightly limited. However, in the simulations, the cost of the computed paths is extremely close to the optimal. The simulations on LatticeFM and LatticeSL confirm these results: kID-MCP usually manages to find a solution when there is a feasible path in the network (ASR $\geq$ 97%) and the cost of this solution is close to the one of the optimal path. This is an interesting outcome, which indicates that the complexity reduction mechanism adopted by kID-MCP, namely memorizing at most a single path from every intermediate node, is efficient. A possible explanation for this promising result is that, as the domains are connected through several entry-BNs, the path diversity remains sufficient even if the heuristics maintains a single path from every node.

Globally, this study shows that, in the simulated scenarios, kID-MCP computes acceptable solutions in most cases, as desired. Consequently, kID-MCP provides a fast and efficient solution to compute constrained inter-domain paths: it can be used in practical scenarios, for example to provision resources dynamically at inter-domain level.

11.5 Conclusion

In the present chapter, we have derived an efficient heuristic from the exact algorithm introduced in Chapter 9. We name this heuristic kID-MCP. We have integrated the proposed algorithm to solve the inter-domain MCP problem in the PCE architecture. In particular, kID-MCP is compatible with existing techniques developed for the PCE architecture to fulfill

the confidentiality requirements of the domains. In addition, a complexity analysis has shown that kID-MCP scales reasonably well. Finally, a simulation study on several topologies has concluded that, in most situations, kID-MCP finds a path fulfilling the constraints. These results indicate that algorithms with a reasonable complexity can be used for determining inter-domain MCPs without sacrificing route QoS. Thus, inter-domain MCP computations seem tractable in practice, which opens interesting perspectives for inter-domain QoS routing and traffic engineering.

In the present chapter we have studied heuristic solutions to compute inter-domain MCPs with a reasonable time complexity. As explained in previous chapters, the alternative approach consists in developing approximation algorithms to solve the same problem. Such algorithms have typically a larger complexity than heuristics. However, they provide guarantees (approximation bounds) both on the success of the path computation operations and on the quality of the computed paths. Due to the good performance of heuristic solutions, we consider that heuristics represent a better choice than approximation algorithms in most situations.

Key points of Chapter 11

- We propose kID-MCP, a method that finds inter-domain paths with performance guarantees rapidly. The worst-case time complexity of our algorithm is in the same order as the one of Dijkstra's algorithm.
- An extensive simulation study shows that our fast solution performs well in both realistic and worst-case scenarios: kID-MCP finds a feasible path in most situations and the performance of the computed path is close to optimal.
- Approximation algorithms provide guarantees (approximation bounds) both on the success of the path computation operations and on the quality of the computed paths. However, their complexity is significantly larger than the one of our heuristic. Consequently, we think that, in practice, our heuristic is the best approach for computing inter-domain MCP in a reasonable time.
- The performance trade-off between our fast heuristics and approximation algorithms requires further studies.

CHAPTER 12 Conclusion

In the present thesis, we have studied the problem of providing services with end-to-end performance guarantees. This problem involves important stakes for network operators and for service providers because value-added services (*e.g.,* VPN, IPTV) require end-to-end quality of service and because many ISPs would like to move telephony services onto their IP-based infrastructure.

Our work has illustrated the complexity of the QoS provisioning problem, which requires that many mechanisms, belonging to various networking layers, interact. We have shown that, nowadays, it is difficult to provide a specific level of QoS for critical services inside the network of a single operator. Providing end-to-end QoS guarantees for the traffic that traverses several routing domains, operated by various entities, is complicated too. Today, cooperation and information exchanges among the traversed domains are limited because every ISP tries to optimize its network performance and its revenue. We think that enhancing the inter-domain relationships to enable more efficient traffic management would be beneficial for all involved stakeholders [20]. Evolutions of the legacy organizational model of the Internet, in which networks are autonomous and disclose little information to adjacent networks, are already observable from the work realized in pre-standardization industry forums and in standardization organizations. Our work contributes to this evolution toward enabling QoS across the networks of several carriers.

Carriers' inter-connections that support QoS guarantees represent an enabler for various end-to-end services. We have investigated the constrained inter-domain routing problem, which has important applications such as dynamic resource provisioning at inter-domain level. We have shown that existing solutions to this problem are either restrictive or inapplicable in the considered context. Consequently, we have described novel propositions that enable network operators to route inter-domain traffic on inter-domain paths that fulfill a set of request constraints. More precisely, we have analyzed existing constrained routing solutions to exhibit their fundamental principles and their limitations for the inter-domain constrained path computation problem. This work helped us to introduce the first solution that enables distributing inter-domain constrained-routing operations and to compute the best inter-domain path for a given request in the PCE framework. We have enhanced this exact solution with pre-computation. Our algorithms provide optimal solutions (the paths that are the furthest from the request constraints). As the worst-case time complexity of exact solutions is necessarily large, we have studied approximation algorithms, which provide accurate performance guarantees and solve the InterMCP problem in polynomial time. Finally, we have introduced and evaluated a novel heuristic for solving the same problem rapidly: an extensive simulation study has shown that the time complexity and the performance of this heuristic are good

Algorithm	Properties	Worst-Case Time Complexity	Reference
ID-MCP	slow, exact	$\mathcal{O}(D\alpha^2K(\alpha V + E + V^2))$	[22], p. 98
pID-MCP	pre-computation, exact	$\mathcal{O}(\alpha^2K(E + MV^2))$ offline and $\mathcal{O}(\alpha^4{N_{\text{BN}}}^3)$ online, in every domain	[21], p. 102
aID-MCP	polynomial time, approximate	$\mathcal{O}\left(D \cdot E \cdot (\lfloor\theta\rfloor + 1)^{K-1}\right)$	p. 107
kID-MCP	fastest, heuristic	$\mathcal{O}(D \cdot K \cdot V^2)$	[23], p. 125

Table 12.1: Synthesis of our solutions to the inter-domain MCP problem

Table 12.1 synthesizes our proposed algorithms to solve the inter-domain multi-constrained routing problem. This table shows that the complexity of approximation algorithms for the inter-domain MCP problem is larger than the one of the fastest heuristics. Our studies show that the performance of our heuristic is good. Therefore, we think that, in most situations, our heuristic represents a better choice than approximation algorithms. To conclude, all the proposed solutions have their strengths and weaknesses: the selection of a specific algorithm depends essentially on the application context (*e.g.,* size of the considered network, reactivity requirements).

Our inter-domain traffic-management solutions enable determining the most appropriate paths for routing critical traffic aggregates dynamically. Thus, we are convinced that our contributions represent an important step toward more efficient inter-domain traffic management in traffic-engineered networks. Specifically, they contribute to making it easier for carriers to transition toward an all-IP world thanks to the provision of inter-domain QoS guarantees and thanks to inter-domain traffic engineering.

Our work has opened many perspectives for future work: for example, it has led to a successful collaborative project (Citric) and has triggered multiple research activities including a post-doc., two M.Sc. internships, and two future Ph.D. theses. Many additional applications of our work would be interesting to study. For instance, our algorithms could be adapted to compute constrained paths that use diverse inter-domain links. This would enable operators to balance the traffic load on inter-domain links. This issue is critical for network operators because peering links are known to represent a potential congestion point. Furthermore, our algorithms might prove useful in military networks, because they enable constrained routing in networks where the topology information is fragmented for security reasons.

Our work could be extended in various research directions. It would be interesting to analyze more profoundly the trade-offs between heuristic and approximate solutions to the MCP problem. An ideal solution would provide strong performance guarantees (like approximation algorithms) and have a low computational complexity (like heuristics). We currently integrate our approximation algorithm into a simulation framework and we hope to present evaluation results in a future paper. In addition, studying scenarios in which the traversed domains use different path computation algorithms (for example exact algorithms and heuristics) would be useful, for instance to enable incremental deployment and to adapt the routing accuracy to the computational load in the PCEs. Finally, from the architectural point of view, the trade-offs between BGP-tuning and PCE-based traffic engineering could be studied in a more detailed manner.

In the long term, further studies could investigate the following points. It would be interesting to examine the stability of the computed paths when the network state fluctuates. Such

study might require modeling the uncertainty (*e.g.,* state variations, information aggregation) on the metric values considered by the routing algorithm. Furthermore, future work should define strategies for the situations where a network is not able to provide a path that fulfills all requirements. A possibility would be to relax the request constraints for specific services. Last but not least, studying inter-domain QoS provisioning from the angle of game theory would be interesting. Specifically, it is necessary to provide incentives to carriers so that they provide the agreed level of QoS. In particular, reputation mechanisms represent a possible solution that should be considered.

APPENDIX A

Additional Simulation Results

In the present appendix, we provide additional simulation results, which enable us to show the performance of our algorithms in various scenarios. In the simulations, we include an algorithm named kpID-MCP that is similar to pID-MCP but that limits the maximum number of paths memorized for every node. Stated differently, kpID-MCP is a heuristic derived from the exact algorithm pID-MCP, just like kID-MCP is a heuristic derived from the exact algorithm ID-MCP.

A.1 Simulation Settings

A.1.1 Topologies

We have compared the performance of the algorithms presented in previous chapters through simulations. The performance of most routing algorithms is closely related to the properties (*e.g.,* topology, metrics) of the network to which they are applied. Here, intuitively, the size (number of domains, number of nodes in each domain) and the connectivity (domain degrees, node degrees) of the topologies considered have a strong effect on the performance of the routing algorithms compared. Thus, the topologies considered in the simulations have been carefully selected. A difficulty arising in this selection is that the PCE architecture is not expected to be deployed in the whole Internet but only in small sets of ASes, and the structure of these sets of ASes is not publicly available. Therefore, we have evaluated the performance of our algorithms on two types of topologies. First, we have used the following *lattice topologies* to assess the performances of the algorithms in extreme configurations. Lattice topologies represent an extreme case for QoS routing algorithms, as described in Reference [111]. These topologies are square grids in which the top-left node is the source and the bottom-right node is the destination of a request. We consider two topologies based on lattices.

- LatticeFM(N,D) (Full-Mesh) represents a worst-case for the complexity of the algorithms.[1] This topology is a chain of D identical domains; every domain is a grid made up of N nodes and $E = 2\sqrt{N}(\sqrt{N} - 1)$ undirected links. Every node of every domain is connected to every node of the previous domain, as well as to every node of the next domain in the chain.

[1] The absolute worst-case topology considering the number of paths is a topology in which any node is connected to any other node inside or outside its domain.

- LatticeSL(N,D) (Single Link) is similar to LatticeFM(N,D) except that only the top-left node of each domain is connected to the bottom-right node of the previous domain.

Second, we have used more realistic topologies to assess the applicability of our algorithms to real networks.

- A realistic backbone topology which appears in Reference [118] and is based on topologies measured by RocketFuel [160], essentially. It includes measurements of the networks of eight major Internet service providers in the USA. We refer to this topology as REAL(8).
- A fictitious inter-area topology named SYM-CORE, which was used in Reference [55].

A.1.2 Link Weights

Random weights have been added to the edges of the topologies considered, except for the topology REAL(8), which includes delay estimations. Most simulations have been realized with two weights ($K = 2$). We have not taken any assumption about the kind of additive weights considered: for example, the weights may be related to the number of hops ($w_l = 1,\ \forall l \in E$), to the link propagation delay, to the inverse of the capacity of the links, or to a measure of the reliability of the links. In addition, we have considered additive constraints only: for instance, we have not considered the effect of the capacity of the links, because bandwidth is a bottleneck metric that can easily be treated by edge pruning. We consider that every edge has enough bandwidth to serve the simulated requests. This assumption seems reasonable if the largest requests are rejected by an admission-control mechanism or if the network is over-dimensioned.

Link weights are usually advertised through a routing protocol that allocates a fixed number of bits for this information. Thus, link weights can be considered to be bounded by a function of the size in bits of the corresponding field of the routing messages. The bounds on the link weights have been chosen arbitrarily, without any loss of generality, as weights can be considered to be scaled. The number of possible values of the weights has an effect on the number of non-dominated paths and, thus, on the complexity, consequently, we have considered weights with a relatively fine granularity.

The work in Reference [170] suggests that, in the Internet, the link weights can be modeled using a uniform distribution. In addition, several important papers in the area adopt uniformly distributed weights (*e.g.,* [59, 109, 169]). Hence, we have generated uniformly-distributed link weights. We have assumed that the weights of inter-domain links follow the same distribution as the ones of intra-domain links. Constant weights (*e.g.,* number of hops) represent a special case of uniformly distributed weights.

The correlation of the weights $w_k,\ k = 1..K$ is known to have an effect on the complexity of MCP algorithms [111]. Thus, we have performed both simulations with independent link weights and simulations with positively or negatively-correlated link weights. When we consider correlated weights, we assume that the correlation of the weights is the same in all the traversed domains, so that the effect of correlation is visible on end-to-end path computations. The method that we use to generate correlated weights is inspired by Reference [109]: to generate K positively-correlated weights in the interval $]A, B]$, we partition the interval $]A, B]$ into $\left\{I_0 \equiv \left]A, A + \frac{B-A}{2}\right], I_1 \equiv \left]A + \frac{B-A}{2}, B\right]\right\}$. We generate a random weight w_1. If w_1 is in I_0 then we generate w_k, k= 2..K in the interval I_0, else we generate w_k, k= 2..K inside the interval I_1.

Weights	uniform distribution with $10 \leq w_k \leq 1023$, $k = 1..K$ or $k = 1..K-1$; correlated or not; delay estimations and number of hops for REAL(8)
Constraints	constant during a simulation, fixed depending on *a posteriori* observation of the rate of success of the computation and the cost of the paths computed

Table A.1: Parameters used for the generation of the weights and of the requests in the main simulations

A.1.3 Constraints

The value of the end-to-end constraints has an effect both on the complexity of ID-MCP and of pID-MCP and on their performance compared to kID-MCP ($k = 1$). For example, if $\mathbf{p}_1$ is a shortest path with respect to w_1 and $\mathbf{p}_2$ a shortest path with respect to w_2, then the constraints W_1 and W_2 should be selected in the interval $w_1(\mathbf{p}_1) < W_1 < w_1(\mathbf{p}_2)$ and $w_2(\mathbf{p}_2) < W_2 < w_2(\mathbf{p}_1)$. If the constraints are chosen outside these intervals, then, either there is no solution, which can be verified with a polynomial complexity, or, a shortest path with respect to a single metric is an evident solution, which can be computed with a polynomial complexity. Reference [111] investigates the effect of the choice of the constraints.

We know that the exact algorithms will find a solution if one exists. Consequently, we have chosen the constraints to obtain a high rate of success of the path computation procedure, thus, we focus on the cases where a solution exists. We have performed both simulations where the constraints are equally strict in average or where one constraint is stricter than the other. We consider the proportion of the requests for which the exact algorithms find a solution, as well as the value of the cost metric of SAMCRA (Equation 8.3) for the lowest-cost solutions computed, to determine *a posteriori* if the constraints are strict or loose. More precisely, loose means that exact algorithms find a solution for every request simulated and at least one solution has a low cost ($c(\mathbf{p}) \leq 0.2$). Strict means that in average the lowest-cost solutions found by exact algorithms have a high cost ($c(\mathbf{p}) \geq 0.8$) or a feasible solution exists for less than 60% of the requests simulated.

A.1.4 Selection of Domain Sequences

We focus on the problem of computing inter-domain paths along predefined domain sequences. In each simulation run, we have selected a source and a destination node in different domains. In the Internet, the domain sequence followed by traffic from a source to a destination domain is not necessarily the shortest, because this sequence depends on the policies of the domains. However, modeling the policies of each domain explicitly is difficult. Thus, we have rather tried to reproduce the average length of the AS-paths in the Internet. We have considered a domain sequence length equal to three, which matches the average domain sequence length in the Internet, according to [35]. In the realistic topology, we have computed a shortest domain-sequence between the source and the destination domain and considered this sequence for the node-level path computation between the source node and the destination node.

A.1.5 Performance Metrics

We denote as M(A) the average value of the metric M for the algorithm A for a batch of simulations. For example, α(pID-MCP) denotes the value of α measured for the algorithm pID-MCP in a set of simulations. Confidence intervals are computed for all performance metrics. When the number of requests simulated is not provided explicitly, this means that it is large enough to provide statistically significant results.

We define the absolute success rate (ASR) as the percentage of success of the algorithms to find a feasible path when a solution exists. The success rate (SR) is the percentage of success of the algorithms to find a feasible path for the requests considered. As ID-MCP and pID-MCP are exact, their ASR is 100%, whereas the ASR of kID-MCP, k=1 or kpID-MCP is not necessarily 100%. The number of paths returned by the algorithms is denoted as NP. Both exact algorithms return all end-to-end feasible non-dominated paths, thus, NP(ID-MCP)=NP(pID-MCP). With our implementation, the number of paths returned by kpID-MCP can exceed k, because the segments computed by a domain are combined with the segments in the previous VSPT. Our implementation of kID-MCP, k=1 returns a single path per request.

The cost (C) is the lowest value of the path-length function of SAMCRA among the paths computed, thus, it takes the same value for all exact algorithms. We define an additional path-length function $c'(\mathbf{p})$ as $\mu_{i=1..K}\left(\frac{w_i(\mathbf{p})}{W}\right)$, where μ denotes the arithmetic mean operator. We call multi-dimensional cost (MC) the value of c' for the end-to-end path that has the lowest value of c' among the computed paths. MC helps evaluating the quality of the returned paths considering all metrics, whereas C indicates their quality with respect to the most restrictive metric. The costs (C and MC) of the paths are taken into account only for the requests for which both heuristics and exact algorithms succeed to find a feasible path, so that the comparison of the algorithms is meaningful.

In accordance with the worst-case time complexity bounds presented in Chapter 9, we derive the relative time complexity of the algorithms from the measurement of the maximum number (α) of paths attached to a node and memorized in the computation queue. The value of α provides also an indication of the spatial complexity of the algorithms. By definition, α(kID-MCP, k=1) is equal to one.

We have simulated the algorithms ID-MCP, pID-MCP, kID-MCP with $k = 1$ and kpID-MCP ($k > 1$) on various network configurations to assess their performance and to illustrate their strengths and weaknesses. The scenarios allow us to outline some of the trade-offs between the proposed inter-domain multi-constraint path computation methods.

For more clarity, we define a reference simulation scenario to which all other scenarios are compared. The reference simulation scenario involves positively-correlated weights and its constraints are identical for all weights ($W_i = W_j,\ 1 \leq i \leq K,\ 1 \leq j \leq K$). Simulations with independent or negatively-correlated weights use the same constraints as in the reference scenario. We refer to the constraints as strict or loose constraints *with respect to the reference scenario.*

	SR [%]		C [%]		MC [%]		α		NP	
Lattice	*SL*	*FM*	*SL*	*FM*	*SL*	*FM*	*SL*	*FM*	*SL*	*FM*
ID-MCP	100	100	19.2	13.9	18.7	11.4	10	7	7	3
pID-MCP	100	100	19.2	13.9	18.7	11.4	5	52	7	3
kpID-MCP, k=3	100	100	19.3	26.8	18.8	23.8	3	3	6	1
kID-MCP, k=1	100	100	19.5	13.9	19	11.9	1	1	1	1

Table A.2: Results of simulations in the reference scenario (positive correlation) with loose constraints $((49100, 49100)^T$ for LatticeSL and $(3000, 3000)^T$ for LatticeFM)

A.2 Simulation Results

A.2.1 Effect of Inter-Domain Connectivity

We compare the performance of ID-MCP and pID-MCP on the topologies LatticeFM and LatticeSL. This comparison illustrates the effect of the inter-domain connectivity of the domains on the complexity of these algorithms.

On the Lattice topologies, the exact algorithms are penalized by a large path diversity that induces an increased time complexity. In particular, the topology LatticeFM(25,3) is designed to illustrate a drawback of the algorithms based on pID-MCP. In this topology, the number of inter-domain links is *extremely* large: each domain is connected to the next domain through 625 inter-domain links. As the algorithms based on pID-MCP perform non-dominance comparisons depending on the destination of each element, they memorize many paths if one domain is connected to the next domain through many ingress nodes. The algorithm kpID-MCP limits the number of paths memorized per node, which solves the problem about the complexity. However, the paths memorized must be selected carefully among the many paths available so that the end-to-end path is close to the optimum. The implemented version of kpID-MCP selects k shortest paths considering the length measure defined in Equation 8.3.

Table A.2 presents the results of the simulations for the topologies LatticeSL(25,3) and LatticeFM(25,3) with loose constraints in the reference scenario. These results illustrate the aforementioned drawback of the methods based on pID-MCP when the inter-domain connectivity is large. As expected, in the LatticeSL topology α(ID-MCP) is greater than α(ID-MCP), whereas in the LatticeFM topology α(ID-MCP) is much lower than α(ID-MCP). This underlines the need for limiting α in the algorithm pID-MCP, which leads to the heuristic kpID-MCP. In the LatticeSL topology kpID-MCP, k=3 provides better results (lower value of C and MC) than kID-MCP, k=1. However, in the LatticeFM topology kpID-MCP, k=3 provides worse results (lower value of C and MC) than kID-MCP, k=1. This problem is solved by allowing larger values of α in kpID-MCP.

Table A.3 presents the results of the simulations in the same scenario but this time with strict constraints. In this scenario, kpID-MCP, k=3 provides better results (lower value of C and MC, higher value of SR and NP) than kID-MCP, k=1 in the LatticeSL topology. In the LatticeFM topology, kpID-MCP, k=3 and kID-MCP, k=1 provide similar results.

	SR [%]		C [%]		MC [%]		α		NP	
Lattice	*SL*	*FM*	*SL*	*FM*	*SL*	*FM*	*SL*	*FM*	*SL*	*FM*
ID-MCP	66	56	89.3	72	86.5	60.1	8	2	5	1
pID-MCP	66	56	89.3	72	86.5	60.1	5	8	5	1
kpID-MCP, k=3	66	50	89.4	72	86.5	60.1	3	3	4	1
kID-MCP, k=1	64	56	89.8	72	87.9	60.1	1	1	1	1

Table A.3: Results of simulations in the reference scenario (positive correlation) with strict constraints $((9800, 9800)^T$ for LatticeSL and $(400, 400)^T$ for LatticeFM)

	SR [%]		C [%]		MC [%]		α		NP	
Lattice	*SL*	*FM*	*SL*	*FM*	*SL*	*FM*	*SL*	*FM*	*SL*	*FM*
ID-MCP	100	100	23.3	27.9	17.4	22.6	67	8	64	5
pID-MCP	100	100	23.3	27.9	17.4	22.6	14	57	64	5
kpID-MCP, k=3	100	100	23.3	33.2	17.4	26.1	3	3	11	2
kID-MCP, k=1	100	100	23.3	28	17.5	23.5	1	1	1	1

Table A.4: Results of simulations with negatively-correlated weights and asymmetric constraints $((147000, 37000)^T$ for LatticeSL and $(9000, 1000)^T$ for LatticeFM)

A.2.2 Effect of the Strictness of the Constraints

We investigate the effect of the non-feasibility check in the exact algorithms ID-MCP and pID-MCP by simulating either strict constraints or loose constraints in the topologies LatticeSL(25,3) and LatticeFM(25,3). ID-MCP considers non-zero initial weights in every domain, which allow it to discard several non-feasible paths when the constraints are strict, whereas pID-MCP considers initial weights equal to zero for the virtual nodes and usually detects less non-feasible paths. Thus, the strictness of the constraints has an effect on the complexity of exact algorithms.

We compare the results in Table A.2 and in Table A.3. In the LatticeFM topology: the values of α are several times smaller with strict constraints than with loose constraints (3.5 times smaller with ID-MCP and even 6.5 times smaller with pID-MCP). These results are explained by the large path diversity in the LatticeFM, which can be drastically reduced when strict constraints are used. With strict constraints, pID-MCP is less penalized by the large inter-domain connectivity of the LatticeFM topology. Typically, the success rate of the path computation procedure, the value of α, as well as the number of paths returned decrease and the cost of the paths computed increases, when the constraints become stricter.

A.2.3 Effect of Asymmetric Constraints

We investigate the performance of the heuristics in the topologies LatticeSL(25,3) and LatticeFM(25,3) when the constraints are asymmetric (W_1 is large and thus, easily fulfilled, whereas W_2 is more restrictive) and the link-weights are negatively-correlated. The heuristics memorize shortest paths with respect to the cost c and thus, select the paths depending on the most restrictive metric. Intuitively, this can lead to non-optimal values of the other metrics for the selected end-to-end path.

	SR [%]		C [%]		MC [%]		α		NP	
Lattice	*SL*	*FM*	*SL*	*FM*	*SL*	*FM*	*SL*	*FM*	*SL*	*FM*
ID-MCP	100	100	24.3	28.6	24	24.6	69	13	65	8
pID-MCP	100	100	24.3	28.6	24	24.6	15	74	65	8
kpID-MCP, k=3	100	100	24.9	48.1	24.4	41.3	3	3	11	2
kID-MCP, k=1	100	100	25.6	28.6	25.1	26.8	1	1	1	1

Table A.5: Results of simulations with negatively-correlated weights and loose constraints $((48100, 48100)^T$ for LatticeSL and $(3000, 3000)^T$ for LatticeFM)

Table A.4 presents the results of the simulations, which we compare to the results in Table A.5 for simulations with symmetric constraints. We have selected the constraints so that the cost C and the success rate SR take similar values in both tables. As expected, the difference between C and MC is larger with asymmetric constraints than with symmetric constraints for all algorithms. However, the difference between the value of MC for the heuristics and for the exact algorithms is smaller with asymmetric constraints than with symmetric constraints.

A.2.4 Effect of the Correlation of the Weights

We investigate the effect of the correlation of the weights on the performance of the algorithms considered. We keep the same constraints as in the reference scenario and simulate requests in the LatticeSL and LatticeFM topologies, but, this time, with negatively-correlated weights. These simulations are performed because the correlation of the weights is known to affect the number of non-dominated paths. When the weights are positively-correlated, a path with a low value for a metric is likely to take a low value for the other metric too: considering two paths $\mathbf{p}_1$ and $\mathbf{p}_2$, if $w_1(\mathbf{p}_1) < w_1(\mathbf{p}_2)$, then, it is likely that $w_2(\mathbf{p}_1) < w_2(\mathbf{p}_2)$. Thus, the number of non-dominated paths is usually low. However, when the weights are negatively-correlated, a path taking a low value for a weight is likely to take a large value for the other constraint. Thus, there are usually more non-dominated paths with negatively-correlated weights than with positively-correlated weights. A higher number of non-dominated paths has a negative effect on the complexity of the path computation algorithms, because more paths need to be memorized.

We compare the results in Table A.5 and in Table A.2. In the LatticeSL topology, the difference of cost between the results of the exact methods and the results of the heuristics is significantly larger with negatively-correlated weights than with positively-correlated weights. This increased difference certainly comes from the larger number of non-dominated paths with negatively-correlated weights. In addition, the value α and NP for exact methods are much larger with negatively-correlated weights than with positively-correlated weights. In fact, α(ID-MCP) is multiplied by seven and α(pID-MCP) by three, which increases the difference of complexity between these algorithms. Thus, pID-MCP is significantly faster than IP-MCP in this configuration. The difference on α has an effect on the cost of the paths returned by the heuristics too. First, the relative difference of cost between kpID-MCP, k=3 and kID-MCP, k=1 rises from about 1% with positively-correlated weights to 3% with negatively-correlated weights, for the requests considered. Second, the relative difference of cost between kID-MCP, k=1 and exact algorithms rises from about 2% with positively-correlated weights to 5% with negatively-correlated weights, for the requests considered.

	SR [%]	C [%]	MC [%]	α	NP
kpID-MCP, k=3	100	14	13	3	8
kID-MCP, k=1	100	10	10	1	1

Table A.6: Results of simulations on a realistic topology with a limit on α equal to three

With our implementation of pID-MCP and kpID-MCP, the segments computed in a domain are combined with the virtual segments of the preceding VSPT. Thus, in the initial queue considered by a domain, the number of paths attached to a virtual node may exceed the maximum allowed value of α. Thus, NP can exceed the number of maximum number of segments attached to the source node (*e.g.,* 3) multiplied by the maximum number of segments attached to the ingresses (*e.g.,* 3) multiplied by the number of ingresses (*e.g.,* 1). This explains why the number of paths (NP=11) returned by kpID-MCP, k=3 in LatticeSL with negatively-correlated weights and loose constraints exceeds the square of the maximum value of α allowed ($3^2 = 9$) multiplied by the number of ingresses (1) in the second domain crossed. An alternative implementation choice would be to enforce the limitation on α after the combination operations.

A constraint value appears usually much stricter with negatively-correlated weights than with positively-correlated weights. For instance, with the loose constraints of the reference scenario, C is approximately 25% larger with negatively-correlated weights than with positively-correlated weights in the LatticeSL topology. This difference is even larger in the LatticeFM topology: C is multiplied by approximately two when negatively-correlated weights are simulated. Moreover, with the strict constraints of the reference scenario no solution exists in the simulation runs with negatively-correlated weights.

A.2.5 Simulations on a Realistic Inter-Domain Topology

We simulate a scenario in the topology REAL(8), which represents the topology in the USA of some of the largest operators in the world [118]. These simulations are used as a demonstration of the applicability of our heuristics on a realistic example. The simulated requests are constrained by a maximum value of the end-to-end one-way speed-of-light propagation delay equal to 100 ms and a maximum number of hops equal to 30. The constraint of 100 ms is feasible according to the service level agreements advertised by Sprint for the USA [161] and seems reasonable for voice traffic, for example (see Reference [93]). The value of the constraint on the number of hops is set arbitrarily, so that this constraint is loose. The number of hops represents a traffic-engineering metric equal to one for every link, which used by operators to minimize the amount of resources taken by each request. The considered link-weights are relatively static[2] and thus, segments can be pre-computed for the considered class of service.

We consider a source-destination domain-pair and we generate twenty source-destination node-pairs. The domain pair considered in the simulations leads to a sequence of two domains. In average, the domains are made up of approximately 1000 nodes and 3500 undirected edges. They are connected through eleven undirected links. We compare the quality of the paths computed with kID-MCP, k=1 and kpID-MCP, k=3. In another simulation we compare the results provided by kID-MCP, k=1 and kpID-MCP, k=8.

[2]They can change if the topology is modified.

	SR [%]	C [%]	MC [%]	α	NP
kpID-MCP, k=8	100	14	13	8	49
kID-MCP, k=1	100	12	12	1	1

Table A.7: Results of simulations on a realistic topology with a limit on α equal to eight

	SR [%]		C [%]		MC [%]		α		NP	
Correlation	+	−	+	−	+	−	+	−	+	−
ID-MCP	60	41	69.2	76.2	64.8	70.3	5	3	2	1
pID-MCP	60	41	69.2	76.2	64.8	70.3	9	5	2	1
kpID-MCP, k=3	60	41	69.9	76.2	65.5	70.4	3	3	2	1
kID-MCP, k=1	58	41	69.6	76.5	65.2	70.4	1	1	1	1

Table A.8: Results of simulations with strict constraints in the SYM-CORE topology

Table A.6 and Table A.7 describe the results of the simulations. The slight difference between the cost of kID-MCP, k=1 in Table A.6 and in Table A.7 comes from the relatively low number of random requests simulated. This low number is not a problem because we are mainly interested into comparing both heuristics for a common set of requests. In both simulations, the paths computed by kpID-MCP have a slightly higher cost (C and MC) than the paths computed by kID-MCP, k=1. This comes probably from the relatively large inter-domain connectivity for the domain sequence considered. The difference of cost between the solutions computed by kpID-MCP and kID-MCP, k=1 decreases slightly when kpID-MCP considers a limit on α equal to eight instead of three. More precisely, for the requests considered, the difference of C between kpID-MCP and kID-MCP, k=1 is divided by two when kpID-MCP considers a limit on α equal to eight instead of three. A larger limit on α is required to approach the optimal solutions more accurately. In average, kpID-MCP, k=3 and kpID-MCP, k=8 compute eight and forty-nine feasible paths whereas kID-MCP, k=1 returns a single feasible path.

A.2.6 Inter-Area Scenario

We have simulated hundred random requests in the SYM-CORE topology with strict constraints. The results of these simulations are presented in Table A.8. They confirm the conclusions presented in previous chapters. The path diversity is smaller in this inter-area topology compared to larger inter-domain topologies. Thus, α and NP take low values. In addition, ASR is quite low whereas C is lower than 80%.

APPENDIX B

Proof of Lemmas

In this appendix, we prove some important lemmas that justify the principles of our algorithms. Some of these lemmas have been proved in the literature [172, 110]. We provide subsequent proofs because we think that they can help the readers to understand our algorithms more thoroughly.

Lemma (8.2.1)**.** Any shortest path $\mathbf{p}^*$ for the linear path length c is made up of shortest sub-paths.

Proof. Consider a non-shortest sub-path $\mathbf{s} \in P_{a\to b}$ of a shortest path $\mathbf{p}$ for the path-length function c. By definition, there exists another path $\mathbf{s}^* \in P_{a\to b}$ between the same nodes as $\mathbf{s}$ and such that $c(\mathbf{s}^*) < c(\mathbf{s})$. Consider the path obtained by replacing the sub-path $\mathbf{s}$ of $\mathbf{p}$ by the shortest path $\mathbf{s}^* \in P_{a\to b}$. According to Equation (8.1), we can write the cost of the two considered paths as follows: $c(\mathbf{p}) = c(\mathbf{p}\backslash\mathbf{s}) + c(\mathbf{s})$ and $c(\mathbf{p}^*) = c(\mathbf{p}\backslash\mathbf{s}) + c(\mathbf{s}^*)$. We have defined the path segments so that $c(\mathbf{s}^*) < c(\mathbf{s})$, thus $c(\mathbf{p}^*) < c(\mathbf{p})$ is verified, which contradicts the assumption that $\mathbf{p}$ is a shortest path for the path-length function c. □

The previous proof relies on a particular property of linear path-length function that is close to the concept of *isotonicity* defined by Sobrinho [156]. Sobrinho shows that *isotonicity* is both necessary and sufficient for a generalized Dijkstra's algorithm to produce optimal paths. We can adapt his definition of isotonicity for the considered problem: a function c is isotone if for all paths $\mathbf{p}_1$ and $\mathbf{p}_2$ in $P_{a\to b}$, the property $c(\mathbf{p}_1) \leq c(\mathbf{p}_2)$ implies both $c(\mathbf{p}_1 \cup \mathbf{p}_3) \leq c(\mathbf{p}_2 \cup \mathbf{p}_3)$ for all $\mathbf{p}_3$ in $P_{b\to V}$ and $c(\mathbf{p}_4 \cup \mathbf{p}_1) \leq c(\mathbf{p}_4 \cup \mathbf{p}_2)$ for all $\mathbf{p}_4$ in $P_{V\to a}$. The concept of isotonicity is important because it enables identifying the properties that a cost function must fulfill to enable solving the MCP problem with simple shortest-path algorithms.

Lemma (8.3.1)**.** [Identical weights] If two intermediate paths $\mathbf{p}_1 \in P_{a\to b}$ and $\mathbf{p}_2 \in P_{a\to b}$ have the same weights $\vec{w}(\mathbf{p}_1) = \vec{w}(\mathbf{p}_2)$, then an MCP algorithm can memorize only one of these intermediate paths without losing the guarantee to find the optimal end-to-end path.

Proof. Consider a path $\mathbf{p} \in P_{s\to t}$ that includes a sub-path $\mathbf{p}_1 \in P_{a\to b}$ going from a node a to another node b in V. We assume that the network offers an alternative intermediate path $\mathbf{p}_2 \in P_{a\to b}$ between the nodes a and b with the same weights as $\mathbf{p}_1$. Then the weight of the path $\mathbf{p}'$ that includes $\mathbf{p}_2$ instead of $\mathbf{p}_1$ verifies $\vec{w}(\mathbf{p}') = \vec{w}(\mathbf{p}) - \vec{w}(\mathbf{p}_1) + \vec{w}(\mathbf{p}_2) = \vec{w}(\mathbf{p})$. Therefore, both paths have the same weight vector, which implies that an MCP algorithm can memorize only one of the intermediate paths $\mathbf{p}_1, \mathbf{p}_2$ without losing the guarantee to find the end-to-end path with the lowest weights. □

Lemma (8.3.2)**.** [Dominated paths] For any two nodes a and b in V, given two intermediate paths $\mathbf{p}_1 \in P_{a\to b}$ and $\mathbf{p}_2 \in P_{a\to b}$, if $\mathbf{p}_1$ dominates $\mathbf{p}_2$ then an MCP algorithm can memorize only $\mathbf{p}_1$ without losing the guarantee to find the optimal end-to-end path.

Proof. If a segment $\mathbf{s}$ is dominated by another segment $\mathbf{s}^*$, then any path $\mathbf{p}$ using $\mathbf{s}$ is dominated by the path $\mathbf{p}^*$ obtained by replacing $\mathbf{s}$ by $\mathbf{s}^*$ in $\mathbf{p}$. Stated differently, $\mathbf{p}^*$ is a better answer than $\mathbf{p}$ to the MCP problem for any constraints. □

Lemma (8.3.3)**.** [Infeasible paths] An MCP algorithm can memorize only feasible paths without losing the guarantee to find the optimal end-to-end path.

Proof. If a segment $\mathbf{s}$ is infeasible, then any path $\mathbf{p}$ using $\mathbf{s}$ is infeasible because for all k in $[1..K]$, $w_k(\mathbf{p}) = w_k(\mathbf{p} \setminus \mathbf{s}) + w_k(\mathbf{s}) \geq w_k(\mathbf{s})$ □

Lemma (8.3.4)**.** [Predictably infeasible paths] We consider a constraint vector $\vec{W}$ and a shortest path segment $\mathbf{s}^* \in P_{a\to t}$ for a weight w_k. If a path $\mathbf{p} \in P_{s\to a}$ verifies $w_k(\mathbf{p}) > W_k - w_k(\mathbf{s}^*)$, then an MCP algorithm can discard $\mathbf{p}$ without losing the guarantee to find an optimal[1] end-to-end path.

Proof. We show that any path $\mathbf{p}'$ that includes the path $\mathbf{p}$ is infeasible. If $w_k(\mathbf{p}) \geq W_k - w_k(\mathbf{s}^*)$, then any path $\mathbf{p}' \in P_{s\to t}$ that includes $\mathbf{p}$ verifies $w_k(\mathbf{p}') = w_k(\mathbf{p}) + w_k(\mathbf{p}' \setminus \mathbf{p})$. As $\mathbf{s}^*$ is a shortest path for w_k from a to t, $\mathbf{p}'$ verifies $w_k(\mathbf{p}' \setminus \mathbf{p}) \geq w_k(\mathbf{s}^*)$. Consequently, we have $w_k(\mathbf{p}') \geq w_k(\mathbf{p}) + w_k(\mathbf{s}^*) > W_k$, and thus, $\mathbf{p}'$ is infeasible. □

Lemma (10.3.1)**.** $c(\mathbf{p}^\theta) - c(\mathbf{p}^*) \leq \frac{n-1}{\theta}$, where n denotes the total number of nodes in the considered network (*i.e.* in the graph that includes all the traversed domains).

Proof. The inequality (B.1) follows directly the definition of the paths $\mathbf{p}^\theta$ and $\mathbf{p}^*$.

$$c_\theta(\mathbf{p}^\theta) \leq c_\theta(\mathbf{p}^*) \tag{B.1}$$

In (B.1), we replace the function c_θ by its definition.

$$\max_{k\in[2..K]} \left(\frac{w_k^\theta(\mathbf{p}^\theta)}{\mathcal{C}} \right) \leq \max_{k\in[2..K]} \left(\frac{w_k^\theta(\mathbf{p}^*)}{\mathcal{C}} \right) \tag{B.2}$$

In (B.2), we replace the weights w_k^θ of the paths and $\mathcal{C}$ by their definition.

$$\max_{k\in[2..K]} \left(\frac{\sum_{l\in\mathbf{p}^\theta} (w_k^\theta(l))}{\lfloor\theta\rfloor} \right) \leq \max_{k\in[2..K]} \left(\frac{\sum_{l\in\mathbf{p}^*} (w_k^\theta(l))}{\lfloor\theta\rfloor} \right) \tag{B.3}$$

In (B.3), we replace the weights w_k^θ of the links by their definition.

$$\max_{k\in[2..K]} \left(\frac{\sum_{l\in\mathbf{p}^\theta} (\left\lfloor \frac{w_k(l)\cdot\theta}{W_k} \right\rfloor + 1)}{\lfloor\theta\rfloor} \right) \leq \max_{k\in[2..K]} \left(\frac{\sum_{l\in\mathbf{p}^*} (\left\lfloor \frac{w_k(l)\cdot\theta}{W_k} \right\rfloor + 1)}{\lfloor\theta\rfloor} \right) \tag{B.4}$$

For every edge l and for every weight w_k, the properties of the floor operator lead to the following inequalities.

$$\frac{w_k(l)\cdot\theta}{W_k} \leq \left\lfloor \frac{w_k(l)\cdot\theta}{W_k} \right\rfloor + 1 \leq \frac{w_k(l)\cdot\theta}{W_k} + 1 \tag{B.5}$$

[1]Here the term optimal means the furthest below the constraints.

We combine (B.4) and (B.5) to obtain the next inequalities.

$$\max_{k\in[2..K]}\left(\frac{\sum_{l\in\mathbf{p}^\theta}(\frac{w_k(l)\cdot\theta}{W_k})}{\lfloor\theta\rfloor}\right)\leq\max_{k\in[2..K]}\left(\frac{\sum_{l\in\mathbf{p}^*}(\frac{w_k(l)\cdot\theta}{W_k}+1)}{\lfloor\theta\rfloor}\right) \tag{B.6}$$

In (B.6), we recognize the definition of $w_k(\mathbf{p}^\theta)$ and $w_k(\mathbf{p}^*)$.

$$\frac{\theta}{\lfloor\theta\rfloor}\cdot\max_{k\in[2..K]}\left(\frac{w_k(\mathbf{p}^\theta)}{W_k}\right)\leq\frac{|\mathbf{p}^*|}{\lfloor\theta\rfloor}+\frac{\theta}{\lfloor\theta\rfloor}\cdot\max_{k\in[2..K]}\left(\frac{w_k(\mathbf{p}^*)}{W_k}\right) \tag{B.7}$$

In (B.7), we recognize the definition of $c(\mathbf{p}^\theta)$ and $c(\mathbf{p}^*)$ and we multiply the inequality by the positive quantity $\frac{\lfloor\theta\rfloor}{\theta}$.

$$c(\mathbf{p}^\theta)-c(\mathbf{p}^*)\leq\frac{|\mathbf{p}^*|}{\theta}. \tag{B.8}$$

As the length of any path is bounded by $n-1$, the inequality (B.8) proves the lemma. □

APPENDIX C

Simulated Topologies

The Sym-Core topology represents an inter-area routing scenario and is introduced in [56].

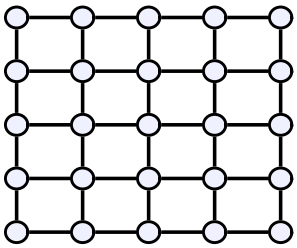

Figure C.1: A lattice domain topology with 25 nodes

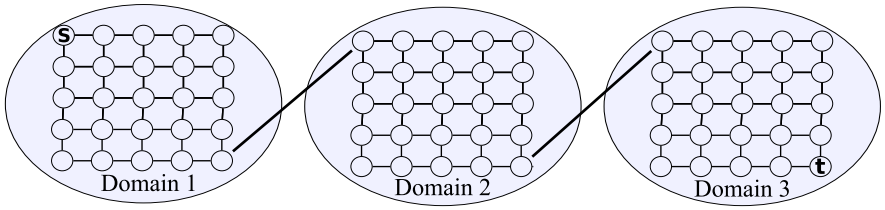

Figure C.2: The LatticeSL topology with three domains: lattices with single link inter-domain connections

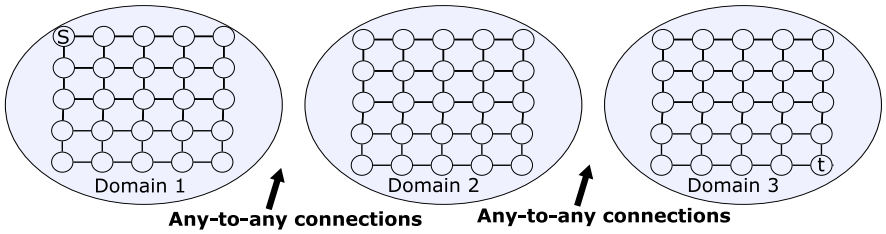

Figure C.3: The LatticeFM topology with three domains: any node is connected with undirected links to any node of the downstream domain

Lattice topologies (Figure C.1, C.2, and C.3) are known to represent hard cases for routing algorithms, because many paths exist between the source and the destination node, and all these paths traverse many links. Kuipers introduces lattices and their properties in [111, 112].

The REAL8 topology is a measurement-based representation of the networks of two major operators and of their inter-connection. We have taken it from [118]. REAL8 has two interesting properties: (1) it represents real large-scale networks, (2) it includes the values of several metrics (*e.g.,* speed of light propagation delay, link capacity).

Bibliography

[1] International IP Interconnection forum (i3 Forum), `http://www.i3forum.org`.

[2] 3GPP, GSM, AND ETSI. TS 24 229: Internet Protocol (IP) multimedia call control protocol based on Session Initiation Protocol (SIP) and Session Description Protocol (SDP) - version 7.2.0 Release 7. Tech. rep., ETSI, 2005.

[3] AHUJA, R., MAGNANTI, T., AND ORLIN, J. *Network Flows: Theory, Algorithms, and Applications*. Prentice-Hall, Inc., New Jersey, 1993.

[4] AKELLA, A., SESHAN, S., AND SHAIKH, A. An empirical evaluation of wide-area internet bottlenecks. In *IMC '03: Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement* (New York, NY, USA, 2003), ACM, pp. 101–114.

[5] ALLMAN, M., PAXSON, V., AND STEVENS, W. TCP Congestion Control. RFC 2581 (Proposed Standard), Apr. 1999. Updated by RFC 3390.

[6] ANDREW, L., AND ANANDA KUSUMA, A. Generalised analysis of a QoS-aware routing algorithm. In *IEEE Global Telecommunications Conference (GLOBECOM)* (1998), vol. 1, pp. 1–6.

[7] ASH, J., AND ROUX, J. L. Path Computation Element (PCE) Communication Protocol Generic Requirements. RFC 4657 (Informational), Sept. 2006.

[8] AWDUCHE, D., BERGER, L., GAN, D., LI, T., SRINIVASAN, V., AND SWALLOW, G. RSVP-TE: Extensions to RSVP for LSP Tunnels. RFC 3209 (Proposed Standard), Dec. 2001. Updated by RFCs 3936, 4420, 4874, 5151, 5420.

[9] AYYANGAR, A., KOMPELLA, K., VASSEUR, J., AND FARREL, A. Label Switched Path Stitching with Generalized Multiprotocol Label Switching Traffic Engineering (GMPLS TE). RFC 5150 (Proposed Standard), Feb. 2008.

[10] BABIARZ, J., CHAN, K., AND BAKER, F. Configuration Guidelines for DiffServ Service Classes. RFC 4594 (Informational), Aug. 2006.

[11] BAKER, F. Requirements for IP Version 4 Routers. RFC 1812 (Proposed Standard), June 1995. Updated by RFC 2644.

[12] BAKER, F., ITURRALDE, C., FAUCHEUR, F. L., AND DAVIE, B. Aggregation of RSVP for IPv4 and IPv6 Reservations. RFC 3175 (Proposed Standard), Sept. 2001. Updated by RFC 5350.

[13] Barbehenn, M. A Note on the Complexity of Dijkstra's Algorithm for Graphs with Weighted Vertices. *IEEE Transactions on Computers 47*, 2 (1998), 263.

[14] Barth, D., Mautor, T., and Villa Monteiro, D. Impact of alliances on end-to-end QoS satisfaction in an interdomain network. In *IEEE International Conference on Communications (ICC)* (2009).

[15] Bates, T., Chen, E., and Chandra, R. BGP Route Reflection: An Alternative to Full Mesh Internal BGP (IBGP). RFC 4456 (Draft Standard), Apr. 2006.

[16] Benmohamed, L., Liang, C., Naber, E., and Terzis, A. QoS Enhancements to BGP in Support of Multiple Classes of Service. draft-liang-bgp-qos-00.txt, work in progress, IETF, 2006.

[17] Bernet, Y., Ford, P., Yavatkar, R., Baker, F., Zhang, L., Speer, M., Braden, R., Davie, B., Wroclawski, J., and Felstaine, E. A Framework for Integrated Services Operation over Diffserv Networks. RFC 2998 (Informational), Nov. 2000.

[18] Bertrand, G. The IP Multimedia Subsystem in Next Generation Networks, 2007. White paper.

[19] Bertrand, G., Lahoud, S., Molnár, M., and Texier, G. Inter-Domain Path Computation with Multiple Constraints. Tech. Rep. 1902, IRISA, 2008. `http://hal.inria.fr/inria-00319401`.

[20] Bertrand, G., Lahoud, S., Molnár, M., and Texier, G. *Intelligent Quality of Service Technologies and Network Management: Models for Enhancing Communication.* IGI Global, Hershey, Pennsylvania, USA, 2010, ch. QoS Routing and Management in Backbone Networks.

[21] Bertrand, G., Lahoud, S., Molnár, M., and Texier, G. *Recent Advances in Providing QoS and Reliability in the Future Internet Backbone.* Nova Science, Hauppauge, NY, USA, 2010, ch. Inter-Domain Path Computation with Multiple QoS Constraints. Accepted for publication.

[22] Bertrand, G., Lahoud, S., Texier, G., and Molnár, M. A Distributed Exact Solution to Compute Inter-Domain Multi-Constrained Paths. In *EUNICE – The Internet of the Future* (September 2009), vol. 5733 of *Lecture Notes in Computer Sciences*, Springer.

[23] Bertrand, G., Lahoud, S., Texier, G., and Molnár, M. Computation of Multi-Constrained Paths in Multi-Domain MPLS-TE Networks. In *Fifth Conference on Next Generation Internet Networks (NGI), Aveiro, Portugal* (July 2009), IEEE.

[24] Bertrand, G., and Texier, G. DiffServ-Aware Flow Amission Control and Resource Allocation Modeling. In *EuroFGI Workshop on IP QoS and Traffic Control, Lisbon, Portugal* (2007).

[25] Bertrand, G., and Texier, G. Ad-hoc Recursive PCE Based Inter-domain Path Computation (ARPC) Methods. In *Fifth International Working Conference on Performance Modelling and Evaluation of Heterogeneous Networks (HET-NETs)* (February 2008).

[26] Bertrand, G., and Texier, G. *Mobility Management and Quality-of-Service for Heterogeneous Networks*. River Publishers, Gistrup, Denmark, ISBN: 978-87-92329-20-2, 2009, ch. Ad-Hoc Recursive PCE-Based Inter-Domain Path Computation (ARPC) Methods.

[27] Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., and Weiss, W. An Architecture for Differentiated Service. RFC 2475 (Informational), Dec. 1998. Updated by RFC 3260.

[28] Boucadair, M. QoS-Enhanced Border Gateway Protocol. draft-boucadair-qos-bgp-spec-01, work in progress, IETF, 2005.

[29] Boucadair, M., Levis, P., Griffin, D., Wang, N., Howarth, M., Pavlou, G., Mykoniati, E., Georgatsos, P., Quoitin, B., Rodriguez Sanchez, J., and Garcia-Osma, M. A Framework for End-to-End Service Differentiation: Network Planes and Parallel Internets. *IEEE Communications Magazine 45*, 9 (September 2007), 134–143.

[30] Braden, R. Requirements for Internet Hosts - Communication Layers. RFC 1122 (Standard), Oct. 1989. Updated by RFCs 1349, 4379.

[31] Braden, R., Clark, D., and Shenker, S. Integrated Services in the Internet Architecture: an Overview. RFC 1633 (Informational), June 1994.

[32] Braden, R., Zhang, L., Berson, S., Herzog, S., and Jamin, S. Resource ReSerVation Protocol (RSVP) – Version 1 Functional Specification. RFC 2205 (Proposed Standard), Sept. 1997. Updated by RFCs 2750, 3936, 4495.

[33] Bradford, R., Vasseur, J., and Farrel, A. Preserving Topology Confidentiality in Inter-Domain Path Computation Using a Path-Key-Based Mechanism. RFC 5520 (Proposed Standard), Apr. 2009.

[34] Briscoe, B., and Rudkin, S. Commercial Models for IP Quality of Service Interconnect. *BT Technology Journal 23*, 2 (Apr. 2005), 171–195.

[35] Bu, T., and Towsley, D. On distinguishing between Internet power law topology generators. In *Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)* (2002), vol. 2, pp. 638–647.

[36] Burris, S., and Sankappanavar, H. *A Course in Universal Algebra*. Springer, 1981.

[37] Calhoun, P., Loughney, J., Guttman, E., Zorn, G., and Arkko, J. Diameter Base Protocol. RFC 3588 (Proposed Standard), Sept. 2003.

[38] Camarillo, G., and Garcia-Martin, M. A. *The 3G IP Multimedia Subsystem - merging the internet and the cellular worlds*. John Wiley & sons, Ltd, 2004.

[39] Camarillo, G., Marshall, W., and Rosenberg, J. Integration of Resource Management and Session Initiation Protocol (SIP). RFC 3312 (Proposed Standard), Oct. 2002. Updated by RFCs 4032, 5027.

[40] Carugi, M., Hirschman, B., and Narita, A. Introduction to the ITU-T NGN focus group release 1: target environment, services, and capabilities. *IEEE Communications Magazine 43*, 10 (Oct. 2005), 42–48.

[41] Cauvin, A., Niger, P., Orozco, J., and Bertrand, G. An Optical Burst Switched network architecture with passive access. In *ICNS '07: Proceedings of the Third International Conference on Networking and Services* (2007), IEEE Computer Society, p. 97.

[42] Cerav-Erbas, S., Delcourt, O., Fortz, B., and Quoitin, B. The Interaction of IGP Weight Optimization with BGP. In *International Conference on Internet Surveillance and Protection (ICISP '06)* (Washington, DC, USA, 2006), IEEE Computer Society.

[43] Chandra, R., Traina, P., and Li, T. BGP Communities Attribute. RFC 1997 (Proposed Standard), Aug. 1996.

[44] Chao, H. J., and Guo, X. *Quality of service control in high-speed networks*. J. Wiley & sons, 2002.

[45] Chen, S., and Nahrstedt, K. An overview of quality of service routing for next-generation high-speed networks: problems and solutions. *IEEE Network 12*, 6 (Nov/Dec 1998), 64–79.

[46] Chen, S., and Nahrstedt, K. On finding multi-constrained paths. In *IEEE International Conference on Communications (ICC), Atlanta, GA, USA* (June 1998), vol. 2, pp. 874–879.

[47] Chiueh, T.-C., Neogi, A., and Stirpe, P. Performance analysis of an RSVP-capable router. In *Fourth IEEE Real-Time Technology and Applications Symposium* (Jun 1998), pp. 39–48.

[48] Cisco. Introduction to EIGRP. Document ID: 13669, `http://www.cisco.com`, August 2005.

[49] Collette, Y., and Siarry, P. *Multiobjective Optimization: Principles and Case Studies*. Springer Verlag, Berlin, 2004.

[50] Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. *Introduction to Algorithms, Second Edition*. The MIT Press, September 2001.

[51] Cuevas, A., Moreno, J. I., Vidales, P., and Einsiedler, H. The IMS Service Platform: A Solution for Next-Generation Network Operators to Be More than Bit Pipes. *IEEE Communications Magazine August* (2006), 75–81.

[52] Cuevas, M. Admission control and resource reservation for session-based applications in next generation networks. *BT Technology Journal 2*, 23 (April 2005), 130–145.

[53] Cui, Y., Wu, J., and Xu, K. Precomputation for intra-domain QoS routing. *Computer Networks 47*, 6 (2005), 923 – 937.

[54] Cui, Y., Xu, K., and Wu, J. Precomputation for multiconstrained QoS routing in high-speed networks. In *Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)* (March-April 2003), vol. 2, pp. 1414–1424.

[55] Dasgupta, S., de Oliveira, J., and Vasseur, J. Performance Analysis of Inter-Domain Path Computation Methodologies. draft-dasgupta-ccamp-path-comp-analysis-02, work in progress, IETF, July 2008.

[56] Dasgupta, S., de Oliveira, J., and Vasseur, J.-P. Path-Computation-Element-Based Architecture for Interdomain MPLS/GMPLS Traffic Engineering: Overview and Performance. *IEEE Network 21*, 4 (Jul.-Aug. 2007), 38–45.

[57] Davie, B., Charny, A., Bennet, J., Benson, K., Boudec, J. L., Courtney, W., Davari, S., Firoiu, V., and Stiliadis, D. An Expedited Forwarding PHB (Per-Hop Behavior). RFC 3246 (Proposed Standard), Mar. 2002.

[58] De Neve, H., and Van Mieghem, P. A multiple quality of service routing algorithm for PNNI. In *IEEE ATM Workshop Proceedings* (1998), pp. 324–328.

[59] De Neve, H., and Van Mieghem, P. TAMCRA: a tunable accuracy multiple constraints routing algorithm. *Computer Communications 23*, 7 (2000), 667–679.

[60] Duan, Z., Zhang, Z.-L., Thomas Hou, Y., and Gao, L. A Core Stateless Bandwidth Broker Architecture for Scalable Support of Guaranteed Services. *IEEE Transactions on Parallel and Distributed Systems 15*, 2 (2004), 167–182.

[61] Dugeon, O., Mingozzi, E., Stea, G., and Bisti, L. Provisioning QoS in interdomain traffic engineering. *Annals of Telecommunications 63* (December 2008), 545–557.

[62] Durham, D., Boyle, J., Cohen, R., Herzog, S., Rajan, R., and Sastry, A. The COPS (Common Open Policy Service) Protocol. RFC 2748 (Proposed Standard), Jan. 2000. Updated by RFC 4261.

[63] Estrin, D., Li, T., Rekhter, Y., Varadhan, K., and Zappala, D. Source Demand Routing: Packet Format and Forwarding Specification (Version 1). RFC 1940 (Informational), May 1996.

[64] Farrel, A., Ayyangar, A., and Vasseur, J. Inter-Domain MPLS and GMPLS Traffic Engineering – Resource Reservation Protocol-Traffic Engineering (RSVP-TE) Extensions. RFC 5151 (Proposed Standard), Feb. 2008.

[65] Farrel, A., Satyanarayana, A., Iwata, A., Fujita, N., and Ash, G. Crankback Signaling Extensions for MPLS and GMPLS RSVP-TE. RFC 4920 (Proposed Standard), July 2007.

[66] Farrel, A., Vasseur, J.-P., and Ash, J. A Path Computation Element (PCE)-Based Architecture. RFC 4655 (Informational), Aug. 2006.

[67] Farrel, A., Vasseur, J.-P., and Ayyangar, A. A Framework for Inter-Domain Multiprotocol Label Switching Traffic Engineering. RFC 4726 (Informational), Nov. 2006.

[68] Faucheur, F. L. Protocol Extensions for Support of Diffserv-aware MPLS Traffic Engineering. RFC 4124 (Proposed Standard), June 2005.

[69] Faucheur, F. L. Russian Dolls Bandwidth Constraints Model for Diffserv-aware MPLS Traffic Engineering. RFC 4127 (Experimental), June 2005.

[70] Faucheur, F. L., and Lai, W. Maximum Allocation Bandwidth Constraints Model for Diffserv-aware MPLS Traffic Engineering. RFC 4125 (Experimental), June 2005.

[71] Faucheur, F. L., Wu, L., Davie, B., Davari, S., Vaananen, P., Krishnan, R., Cheval, P., and Heinanen, J. Multi-Protocol Label Switching (MPLS) Support of Differentiated Services. RFC 3270 (Proposed Standard), May 2002. Updated by RFC 5462.

[72] Festa, P. *Handbook of Optimization in Telecommunications*. Springer US, 2006, ch. Shortest Path Algorithms, pp. 185–210.

[73] Fonte, A., Monteiro, E., Yannuzzi, M., Masip-Bruin, X., and Domingo-Pascual, J. A Framework for Cooperative Inter-Domain QoS Routing. In *EUNICE* (Secaucus, NJ, USA, 2005), Springer-Verlag New York, Inc., pp. 91–104.

[74] Fortz, B., and Thorup, M. Internet traffic engineering by optimizing OSPF weights. In *Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)* (2000), vol. 2, pp. 519–528.

[75] Fortz, B., and Thorup, M. Increasing Internet Capacity Using Local Search. *Computational Optimization and Applications 29*, 1 (Oct. 2004), 13–48.

[76] Fredman, M. L., and Tarjan, R. E. Fibonacci heaps and their uses in improved network optimization algorithms. *Journal of the ACM 34*, 3 (1987), 596–615.

[77] Frikha, A., Bertrand, G., and Lahoud, S. Pré-calcul de chemins inter-domaines soumis à plusieurs contraintes de qualité de service. Tech. Rep. 1935, IRISA, 2009. `http://hal.inria.fr/inria-00319401`,ISSN: 2102-6327.

[78] Frikha, A., Bertrand, G., Lahoud, S., Texier, G., and Bellabas, A. Inter-Domain QoS Routing Based on Autonomous Pre-Computation. *Annals of Telecommunications* (2010). Submitted.

[79] Fudenberg, D., and Tirole, J. *Game Theory*. MIT Press, October 1991.

[80] Gao, L. On inferring autonomous system relationships in the internet. *IEEE/ACM Transactions on Networking 9*, 6 (2001), 733–745.

[81] Garcia-Martin, M., Henrikson, E., and Mills, D. Private Header (P-Header) Extensions to the Session Initiation Protocol (SIP) for the 3rd-Generation Partnership Project (3GPP). RFC 3455 (Informational), Jan. 2003.

[82] Garey, M. R., and Johnson, D. S. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979.

[83] Goel, A., Ramakrishnan, K., Kataria, D., and Logothetis, D. Efficient computation of delay-sensitive routes from one source to all destinations. In *Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)* (2001), vol. 2, pp. 854–858 vol.2.

[84] Guerin, R., and Orda, A. Computing shortest paths for any number of hops. *IEEE/ACM Transactions on Networking 10*, 5 (Oct 2002), 613–620.

[85] Heinanen, J., Baker, F., Weiss, W., and Wroclawski, J. Assured Forwarding PHB Group. RFC 2597 (Proposed Standard), June 1999. Updated by RFC 3260.

[86] Hernandez-Orallo, E., and Vila-Carbo, J. Efficient QoS routing for differentiated services EF flows. In *10th IEEE Symposium on Computers and Communications (ISCC)* (June 2005), pp. 91–96.

[87] Hoffman, P., and Harris, S. The Tao of IETF - A Novice's Guide to the Internet Engineering Task Force. RFC 4677 (Informational), Sept. 2006.

[88] Howarth, M. P., Boucadair, M., Flegkas, P., Wang, N., Pavlou, G., Morand, P., Coadic, T., Griffin, D., Asgari, A., and Georgatsos, P. End-to-end quality of service provisioning through inter-provider traffic engineering. *Computer Communications 29*, 6 (Mar. 2006), 683–702.

[89] Huang, Y., Feamster, N., Lakhina, A., and Xu, J. J. Diagnosing network disruptions with network-wide analysis. *SIGMETRICS Performance Evaluation Review 35*, 1 (2007), 61–72.

[90] Huston, G. Commentary on Inter-Domain Routing in the Internet. RFC 3221 (Informational), Dec. 2001.

[91] Huston, G. The 32-bit AS Number Report. `http://www.potaroo.net/tools/asn32/`, March 2009.

[92] ITU-T. Information technology - Open Systems Interconnection - Basic reference model: The basic model. Recommendation X.200, ISO/IEC 7498-1, 1994.

[93] ITU-T. Recommendation G.114. `http://www.itu.int/rec/T-REC-G.114-200305-I`, May 2003.

[94] ITU-T. General overview of NGN. Recommendation Y.2001, Dec. 2004.

[95] ITU-T. General principles and general reference model for Next Generation Networks. Recommendation Y.2011, Oct. 2004.

[96] ITU-T. IMS for Next Generation Networks. Recommendation Y.2021, Sept. 2006.

[97] ITU-T. Resource and admission control functions in next generation networks. Recommendation Y.2111, 2006.

[98] Jaffe, J. M. Algorithms for finding paths with multiple constraints. *Networks 14* (1984), 95–116.

[99] Jamoussi, B., Andersson, L., Callon, R., Dantu, R., Wu, L., Doolan, P., Worster, T., Feldman, N., Fredette, A., Girish, M., Gray, E., Heinanen, J., Kilty, T., and Malis, A. Constraint-Based LSP Setup using LDP. RFC 3212 (Proposed Standard), Jan. 2002. Updated by RFC 3468.

[100] Johnson, D. M. QoS control versus generous dimensioning. *BT Technology Journal 23*, 2 (2005), 81–96.

[101] Joseph, V., and Chapman, B. *Deploying QoS for Cisco IP-Based and Next Generation Networks: The Definitive Guide*, isbn: 012374461x ed. Morgan Kaufmann Publishers, 2009.

[102] Katz, D., Kompella, K., and Yeung, D. Traffic Engineering (TE) Extensions to OSPF Version 2. RFC 3630 (Proposed Standard), Sept. 2003. Updated by RFC 4203.

[103] Khadivi, P., Samavi, S., and Todd, T. D. Multi-constraint QoS routing using a new single mixed metrics. *Journal of Network and Computer Applications 31*, 4 (2008), 656–676.

[104] Kim, C., Gerber, A., Lund, C., Pei, D., and Sen, S. Scalable VPN routing via relaying. *SIGMETRICS Performance Evaluation Review 36*, 1 (2008), 61–72.

[105] Knoll, T. BGP Extended Community Attribute for QoS Marking. draft-knoll-idr-qos-attribute-02, work in progress, IETF, 2009.

[106] Koch, B. F., and Hussmann, H. *Architectures for Quality of Service in the Internet.* Springer Berlin / Heidelberg, 2003, ch. Overview of the Project AQUILA (IST-1999-10077), p. 1084.

[107] Kodialam, M., Lakshman, T. V., Orlin, J. B., and Sengupta, S. Oblivious routing of highly variable traffic in service overlays and IP backbones. *IEEE/ACM Trans. Netw. 17*, 2 (2009), 459–472.

[108] Kompella, K., and Rekhter, Y. Label Switched Paths (LSP) Hierarchy with Generalized Multi-Protocol Label Switching (GMPLS) Traffic Engineering (TE). RFC 4206 (Proposed Standard), Oct. 2005.

[109] Korkmaz, T., and Krunz, M. Multi-constrained optimal path selection. In *Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)* (2001), vol. 2, pp. 834–843.

[110] Kuipers, F. *Quality of Service Routing in the Internet: Theory, Complexity and Algorithms.* PhD thesis, Delft University, 2004.

[111] Kuipers, F., and Van Mieghem, P. The impact of correlated link weights on QoS routing. In *Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)* (2003), vol. 2, pp. 1425–1434.

[112] Kuipers, F. A., and Van Mieghem, P. F. A. Conditions that impact the complexity of QoS routing. *IEEE/ACM Transactions on Networking 13*, 4 (2005), 717–730.

[113] Le Roux, J. L., Vasseur, J. P., and Lee, Y. Encoding of Objective Functions in the Path Computation Element Communication Protocol (PCEP). draft-ietf-pce-of, work in progress, IETF, 2008.

[114] Levis, P., and Boucadair, M. Considerations of Provider-to-Provider Agreements for Internet-Scale Quality of Service (QoS). RFC 5160 (Informational), Mar. 2008.

[115] Li, T., and Smit, H. IS-IS Extensions for Traffic Engineering. RFC 5305 (Proposed Standard), Oct. 2008. Updated by RFC 5307.

[116] Li, Z., and Garcia-Luna-Aceves, J. J. A distributed approach for multi-constrained path selection and routing optimization. In *QShine '06: Proceedings of the 3rd international conference on Quality of service in heterogeneous wired/wireless networks* (New York, NY, USA, 2006), ACM, p. 36.

[117] Li, Z., and Garcia-Luna-Aceves, J. J. Loop-free constrained path computation for hop-by-hop QoS routing. *Computer Networks 51*, 11 (August 2007), 3278–3293.

[118] Liljenstam, M., Liu, J., and Nicol, D. Development of an Internet backbone topology for large-scale network simulations. In *Winter Simulation Conference* (Dec. 2003), vol. 1, pp. 694–702.

[119] Lorenz, D. H., and Raz, D. A simple efficient approximation scheme for the restricted shortest path problem. *Operations Research Letters 28*, 5 (2001), 213 – 219.

[120] Ma, Q., and Steenkiste, P. Quality-of-service routing for traffic with performance guarantees. In *IFIP International Workshop on Quality of Service (IwQoS)* (1997), pp. 115–126.

[121] Mahajan, M., and Parashar, M. Managing QoS for Multimedia Applications in the Differentiated Services Environment. *Journal of Network and Systems Management 11*, 4 (2003), 469–498.

[122] Mahajan, R., Wetherall, D., and Anderson, T. Towards coordinated interdomain traffic engineering. In *SIGCOMM Workshop on Hot Topics in Networking (HotNets)* (New York, NY, USA, 2004), ACM.

[123] Malkin, G. RIP Version 2. RFC 2453 (Standard), Nov. 1998. Updated by RFC 4822.

[124] Marshall, W. Private Session Initiation Protocol (SIP) Extensions for Media Authorization. RFC 3313 (Informational), Jan. 2003.

[125] Masip-Bruin, X., Yannuzzi, M., Serral-Gracia, R., Domingo-Pascual, J., Enriquez-Gabeiras, J., Callejo, M. A., Diaz, M., Racaru, F., Stea, G., Mingozzi, E., Beben, A., Burakowski, W., Monteiro, E., and Cordeiro, L. The EuQoS system: a solution for QoS routing in heterogeneous networks [Quality of Service based Routing Algorithms for Heterogeneous Networks]. *IEEE Communications Magazine 45*, 2 (Feb. 2007), 96–103.

[126] Matos, A., Matos, F., Simoes, P., and Monteiro, E. A Framework for the establishment of inter-domain, on-demand VPNs. In *IEEE Network Operations and Management Symposium (NOMS)* (April 2008), pp. 232–239.

[127] McPherson, D., and Gill, V. BGP MULTI_EXIT_DISC (MED) Considerations. RFC 4451 (Informational), Mar. 2006.

[128] Medhi, D., and Ramasamy, K. *Network Routing: Algorithms, Protocols, and Architectures*. Morgan Kaufmann, 2007.

[129] Medina, A., Taft, N., Salamatian, K., Bhattacharyya, S., and Diot, C. Traffic matrix estimation: existing techniques and new directions. *SIGCOMM Comput. Commun. Rev. 32*, 4 (2002), 161–174.

[130] Mitra, D., and Ramakrishnan, K. A case study of multiservice, multipriority traffic engineering design for data networks. *Global Telecommunications Conference (GLOBECOM) 1B* (1999), 1077–1083.

[131] Moy, J. OSPF Version 2. RFC 2328 (Standard), Apr. 1998.

[132] Nichols, K., Blake, S., Baker, F., and Black, D. Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers. RFC 2474 (Proposed Standard), Dec. 1998. Updated by RFCs 3168, 3260.

[133] Nichols, K., Jacobson, V., and Zhang, L. A Two-bit Differentiated Services Architecture for the Internet. RFC 2638 (Informational), July 1999.

[134] Orda, A. Routing with end-to-end QoS guarantees in broadband networks. *IEEE/ACM Transactions on Networking 7*, 3 (1999), 365–374.

[135] Pan, P. *Scalable Resource Reservation Signaling in the Internet*. PhD thesis, Columbia University, 2002.

[136] Pan, P., Hahne, E., and Schulzrinne, H. BGRP: A Tree-Based Aggregation Protocol for Inter-domain Reservations. *Journal of Communications and Networks 2* (June 2000), 157–167.

[137] Pan, P., and Nolle, T. GIBSON: Global IP-Based Service-Oriented Network Architecture Overview and IMS User Case. IPSphere contribution, September 2006.

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

[139] Postel, J. Internet Protocol. RFC 791 (Standard), Sept. 1981. Updated by RFC 1349.

[140] Postel, J. Transmission Control Protocol. RFC 793 (Standard), Sept. 1981. Updated by RFCs 1122, 3168.

[141] Pouyllau, H., and Haar, S. Distributed Busacker-Gowen algorithm for end-to-end QoS pipe negotiation in X-domain networks. *Annales des Télécommunications 63*, 11-12 (2008), 621–630.

[142] Quoitin, B., Pelsser, C., Bonaventure, O., and Uhlig, S. A performance evaluation of BGP-based traffic engineering. *International journal of network management 15* (2005), 177–191.

[143] Radunović, B., and Boudec, J.-Y. L. A unified framework for max-min and min-max fairness with applications. *IEEE/ACM Trans. Netw. 15*, 5 (2007), 1073–1083.

[144] Rekhter, Y., Li, T., and Hares, S. A Border Gateway Protocol 4 (BGP-4). RFC 4271 (Draft Standard), Jan. 2006.

[145] Rosen, E., Viswanathan, A., and Callon, R. Multiprotocol Label Switching Architecture. RFC 3031 (Proposed Standard), Jan. 2001.

[146] Rosenberg, J., and Schulzrinne, H. An Offer/Answer Model with Session Description Protocol (SDP). RFC 3264 (Proposed Standard), June 2002.

[147] Rosenberg, J., and Schulzrinne, H. Reliability of Provisional Responses in Session Initiation Protocol (SIP). RFC 3262 (Proposed Standard), June 2002.

[148] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and Schooler, E. SIP: Session Initiation Protocol. RFC 3261 (Proposed Standard), June 2002. Updated by RFCs 3265, 3853, 4320, 4916, 5393.

[149] Roux, J.-L. L., Vasseur, J.-P., and Boyle, J. Requirements for Inter-Area MPLS Traffic Engineering. RFC 4105 (Informational), June 2005.

[150] Saad, T., Mouftah, H., and Nouroozifar, A. Constraint-based routing across multi-domain optical WDM networks. In *Canadian Conference on Electrical and Computer Engineering* (May 2004), vol. 4, pp. 2065–2068.

[151] Sang, A., Zhu, H., and Li, S.-Q. Weighted fairness guarantee for scalable Diff-Serv assured forwarding. In *IEEE International Conference on Communications (ICC)* (2001), vol. 8, pp. 2365–2369 vol.8.

[152] Schulzrinne, H., Casner, S., Frederick, R., and Jacobson, V. RTP: A Transport Protocol for Real-Time Applications. RFC 3550 (Standard), July 2003. Updated by RFC 5506.

[153] Scudder, J. G., and Appanna, C. Multisession BGP. draft-ietf-idr-bgp-multisession-03, work in progress, IETF, 2007.

[154] Seitz, N. ITU-T QoS standards for IP-based networks. *IEEE Communications Magazine 41*, 6 (June 2003), 82–89.

[155] Shenker, S., Partridge, C., and Guerin, R. Specification of Guaranteed Quality of Service. RFC 2212 (Proposed Standard), Sept. 1997.

[156] Sobrinho, J. a. L. Algebra and algorithms for QoS path computation and hop-by-hop routing in the internet. *IEEE/ACM Transactions on Networking 10*, 4 (2002), 541–550.

[157] Soldatos, J., Vayias, E., and Kormentzas, G. On the building blocks of quality of service in heterogeneous IP networks. *IEEE Communications Surveys and Tutorials 7*, 1 (First Quarter 2005), 69–88.

[158] Song, J., Chang, M. Y., Lee, S. S., and Joung, J. Overview of ITU-T NGN QoS Control. *IEEE Communications Magazine 45*, 9 (September 2007), 116–123.

[159] Song, M., and Sahni, S. Approximation Algorithms for Multiconstrained Quality-of-Service Routing. *IEEE Transactions on Computers 55*, 5 (2006), 603–617.

[160] Spring, N., Mahajan, R., Wetherall, D., and Anderson, T. Measuring ISP topologies with Rocketfuel. *IEEE/ACM Transactions on Networking 12*, 1 (Feb. 2004), 2–16.

[161] Sprint. Service Level Agreements. `http://www.sprintworldwide.com`. Last visited: 8th August 2008.

[162] Teixeira, R., and Rexford, J. Managing routing disruptions in Internet Service Provider networks. *IEEE Communications Magazine 44* (2006), 160–165.

[163] Thaler, D., and Hopps, C. Multipath Issues in Unicast and Multicast Next-Hop Selection. RFC 2991 (Informational), Nov. 2000.

[164] TISPAN. Next Generation Network (NGN); Quality of Service (QoS) Framework and Requirements. TS 185 001, 2005.

[165] TISPAN. IP Multimedia Subsystem (IMS); Functional architecture NGN IMS Architecture. ES 282 007, 2006.

[166] TISPAN. Resource and Admission Control Sub-System (RACS): Functional Architecture. ES 282 003, May 2008.

[167] TISPAN. Resource and Admission Control Sub-System (RACS): Functional Architecture. ES 282 003, September 2009. V3.4.1, ETSI Standard.

[168] Trotter, G. Terminology for Forwarding Information Base (FIB) based Router Performance. RFC 3222 (Informational), Dec. 2001.

[169] Van Mieghem, P., De Neve, H., and Kuipers, F. Hop-by-hop quality of service routing. *Computer Networks 37*, 3-4 (2001), 407–423.

[170] Van Mieghem, P., Hooghiemstra, G., and van der Hofstad, R. A scaling law for the hopcount in internet. Tech. rep., Delft University of Technology, 2000.

[171] Van Mieghem, P., Kuipers, F., Korkmaz, T., Krunz, M., Curado, M., Monteiro, E., Masip-Bruin, X., Solé-Pareta, J., and Sánchez-López, S. *COST263 final report*. No. 2856 in Lecture Notes in Computer Science. Springer, 2003, ch. Quality of Service Routing, pp. 80–117.

[172] Van Mieghem, P., and Kuipers, F. A. Concepts of exact QoS routing algorithms. *IEEE/ACM Transactions on Networking 12*, 5 (2004), 851–864.

[173] Vasseur, J., Ayyangar, A., and Zhang, R. A Per-Domain Path Computation Method for Establishing Inter-Domain Traffic Engineering (TE) Label Switched Paths (LSPs). RFC 5152 (Proposed Standard), Feb. 2008.

[174] Vasseur, J., Ikejiri, Y., and Zhang, R. Reoptimization of Multiprotocol Label Switching (MPLS) Traffic Engineering (TE) Loosely Routed Label Switched Path (LSP). RFC 4736 (Informational), Nov. 2006.

[175] Vasseur, J., and Roux, J. L. Path Computation Element (PCE) Communication Protocol (PCEP). RFC 5440 (Proposed Standard), Mar. 2009.

[176] Vasseur, J., Zhang, R., Bitar, N., and Roux, J. L. A Backward-Recursive PCE-Based Computation (BRPC) Procedure to Compute Shortest Constrained Inter-Domain Traffic Engineering Label Switched Paths. RFC 5441 (Proposed Standard), Apr. 2009.

[177] Wang, J. H., Chiu, D. M., Lui, J., and Chang, R. Inter-AS Inbound Traffic Engineering via ASPP. *IEEE Transactions on Network and Service Management 4*, 1 (June 2007), 62–70.

[178] Wang, N., Ho, K., Pavlou, G., and Howarth, M. An overview of routing optimization for Internet traffic engineering. *IEEE Communications Surveys and Tutorials 10*, 1 (First Quarter 2008), 36–56.

[179] Wang, Y., Wang, Z., and Zhang, L. Internet traffic engineering without full mesh overlaying. In *Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)* (2001), vol. 1, pp. 565–571 vol.1.

[180] Wang, Z., and Crowcroft, J. Quality-of-Service Routing for Supporting Multimedia Applications. *IEEE Journal on Selected Areas in Communications 14*, 7 (1996), 1228–1234.

[181] Wroclawski, J. Specification of the Controlled-Load Network Element Service. RFC 2211 (Proposed Standard), Sept. 1997.

[182] Wroclawski, J. The Use of RSVP with IETF Integrated Services. RFC 2210 (Proposed Standard), Sept. 1997.

[183] Xue, G., and Kami Makki, S. Multiconstrained QoS Routing: A Norm Approach. *IEEE Transactions on Computers 56*, 6 (2007), 859–863.

[184] Xue, G., Sen, A., Zhang, W., Tang, J., and Thulasiraman, K. Finding a path subject to many additive QoS constraints. *IEEE/ACM Transactions on Networking 15*, 1 (2007), 201–211.

[185] Xue, G., Zhang, W., Tang, J., and Thulasiraman, K. Polynomial time approximation algorithms for multi-constrained QoS routing. *IEEE/ACM Transactions on Networking 16*, 3 (2008), 656–669.

[186] Yannuzzi, M., Fonte, A., Masip-Bruin, X., Monteiro, E., Sánchez-López, S., Curado, M., and Domingo-Pascual, J. *Quality of Service in the Emerging Networking Panorama*. Springer Berlin / Heidelberg, 2004, ch. A Proposal for Inter-domain QoS Routing Based on Distributed Overlay Entities and QBGP, pp. 257–267.

[187] Yannuzzi, M., Sanchez-Lopez, S., Masip-Bruin, X., Sole-Pareta, J., and Jordi-Domingo-Pascual. A combined intra-domain and inter-domain QoS routing model for optical networks. In *Conference on Optical Network Design and Modeling* (2005), pp. 197–203.

[188] Yuan, X. Heuristic algorithms for multiconstrained quality-of-service routing. *IEEE/ACM Transactions on Networking 10*, 2 (2002), 244–256.

[189] Zhang, R., and Vasseur, J.-P. MPLS Inter-Autonomous System (AS) Traffic Engineering (TE) Requirements. RFC 4216 (Informational), Nov. 2005.

[190] Zhang, T., Cui, Y., Zhao, Y., Fu, L., and Korkmaz, T. Scalable BGP QoS Extension with Multiple Metrics. In *International conference on Networking and Services (ICNS)* (July 2006).

[191] Zhang, Z.-L., Duan, Z., Gao, L., and Hou, Y. T. Decoupling QoS control from core routers: a novel bandwidth broker architecture for scalable support of guaranteed services. *ACM SIGCOMM Computer Communication Review 30*, 4 (2000), 71–83.

[192] Zheng, Y., Korkmaz, T., Dou, W., and Tian, J. Highly responsive and efficient QoS routing using pre- and on-demand computations along with a new normal measure. *Computer Networks 50*, 18 (2006), 3743 – 3762.

Glossary

Acronym	Description	Chapter
***k*ID-MCP**	k-limited Inter-Domain MCP	11
AAA	Authentication Authorization and Accounting	3
AC	Admission Control	6
AF	Assured Forwarding	4, 6
aIDMCP	Approximation for Inter-Domain MCP	10
AS	Autonomous System	4
ASN	Autonomous System Number	4
BB	Bandwidth Broker	4
BE	Best-Effort	4, 6
BGCF	Breakout Gateway Control Function	3
BGF	Border Gateway Function	3
BGP	Border Gateway Protocol	4
BN	Border Node	5, 9
BRPC	Backward Recursive PCE-based Computation	5
CAC	Connection Admission Control	1, 6
COPS	Common Open Policy Service	3
CoS	Class of Service	4, 6
CR-LDP	Constraint-based Label Distribution Protocol	4
CSCF	Call Session Control Function	3
CSPF	Constrained Shortest Path First	4
DiffServ	Differentiated Services	3, 4
DSCP	DiffServ Code Point	4
DSLAM	Digital Subscriber Line Access Multiplexer	2
eBGP	external Border Gateway Protocol	4
ECMP	Equal-Cost Multi-Path	4
ECN	Explicit Congestion Notification	6
EF	Expedited Forwarding	4, 6
EGP	Exterior Gateway Protocol	4
EIGRP	Enhanced Interior Gateway Routing Protocol	4
ERO	Explicit Route Object	4, 9

Acronym	Description	Chapter
FEC	Forwarding Equivalence Class	4
FIB	Forwarding Information Base	4
FPTAS	Fully Polynomial Time Approximation Scheme	8, 10
iBGP	internal Border Gateway Protocol	4
ID-MCP	Inter-Domain MCP	1
IETF	Internet Engineering Task Force	2, 5
IGP	Interior Gateway Protocol	4
IMS	IP Multimedia Sub-system	2
IntServ	Integrated Services	4
IPSF	IPSphere forum	5
ISP	Internet Service Provider	2
IXP	Internet eXchange Point	4
L2TP	Layer 2 Termination Point	3
LSP	Label Switched Path	4
LSR	Label Switch Router	4
MAE	Metropolitan Area Exchange	4
MCP	Multi-Constrained Path	8
MCPP	Multi-Constrained Path with Positive rounding	10
MED	Multi-Exit Discriminator	4
MGCF	Media Gateway Controller Function	3
MGCP	Media Gateway Control Protocol	3
MPLS	Multi-Protocol Label Switching	4
MRFC	Multimedia Resource Function Controller	3
MRFP	Multimedia Resource Function Processor	3
NAP	Network Access Point	4
NAT	Network Address Translation	3
NGN	Next-Generation Network	2, 3
OSI	Open Systems Inter-connection	4
OSPF	Open Shortest Path First	4
PCC	Path Computation Client	5, 9
PCE	Path Computation Element	5
PCEP	Path Computation Element (PCE) Communication Protocol	5, 9
PCN	Pre-Congestion Notification	6
PCRep	Path Computation Reply	5
PCReq	Path Computation Request	5
PDP	Policy Decision Point	3
PEP	Policy Enforcement Point	3

Acronym	Description	Chapter
PHB	Per-Hop Behavior	4, 6
PSTN	Public Switched Telephony Network	2
QoE	Quality of Experience	4
QoS	Quality of Service	2
RACF	Resource Admission Control Function	3
RACS	Resource Admission Control Sub-system	3
RCEF	Resource Control Enforcement Function	3
RIP	Routing Information Protocol	4
RSP	Restricted Shortest Path	7
RSVP	Resource reSerVation Protocol	3, 4
RTCP	Real Time Control Protocol	3
RTP	Real Time Protocol	3
SDP	Session Description Protocol	3
SIP	Session Initiation Protocol	3
SLA	Service Level Agreement	3, 5
SLS	Service Level Specification	3
STB	Set Top Box	2
TCP	Transmission Control Protocol	4, 6
TE	Traffic Engineering	4
TED	Traffic Engineering Database	4
TM	Traffic Matrix	6
UDP	User Datagram Protocol	4
VLAN	Virtual Local Area Network	4
VoD	Video on Demand	2
VoIP	Voice over IP	7
VPN	Virtual Private Network	2
VSPT	Virtual Shortest-Path Tree	5
WDM	Wavelength Division Multiplexing	7

Acknowledgment

I would like to thank everybody who has positively contributed to my work. In particular, I am extremely grateful to the people with whom I have most closely worked. First of all, Dr. Géraldine Texier followed my activity from the early beginning. Later, I had the pleasure to collaborate with Dr. Samer Lahoud and Dr. Miklós Molnár. Their involvement, support, and expert insights have considerably helped me to enrich my thesis. Most recently, I have worked with Dr. Mohand Yazid Saidi, who I would like to thank for our interesting discussions on MCP approximation algorithms. Moreover, I have cooperated with Alia Bellabas and Ahmed Frikha, who I would like to thank for our collaboration in the scope of the Citric project. In addition, I would like to express my gratitude to Prof. Xavier Lagrange for being the supervisor of this thesis.

I would like to thank the members of my Ph.D.-committee for reading my dissertation. I feel greatly honored that they have accepted to evaluate my work.

I am thankful to the French government for partially funding the work presented in this dissertation through the projects VoD@IMS and NextTV4All and through the PHC Orchid Arami6. In addition, I have been supported by the European networks of excellence EuroNGI, EuroFGI, and EuroNF.

I am especially grateful to my parents, Éloi and Catherine, my sister, Nathalie, and my brothers, Philippe, Jean-Marie, Dominique, Pascal, and Pierre for having supported me during my studies.

I would like to thank Kim Chi Tran for her constant care and for her marvelous support throughout the long hours of work that have led to this dissertation. In particular, she proofread early versions of this document.

I would like to thank Eddy Hung Sik Yan and Rayene Ben Rayana for reading some parts of the thesis. Their comments have been helpful and I have appreciated their efforts.

Finally, during my years with Telecom Bretagne, I have had many great people as colleagues. I would like to thank them, and especially the members of the Ph.D. student union: they made my efforts to complete the thesis more comfortable.

List of Publications

Book Chapters

- Bertrand, G., Lahoud, S., Molnár, M., and Texier, G. *Recent Advances in Providing QoS and Reliability in the Future Internet Backbone.* Nova Science, Hauppauge, NY, USA, 2010, ch. Inter-Domain Path Computation with Multiple QoS Constraints. To appear (2010).

- Bertrand, G., Lahoud, S., Molnár, M., and Texier, G. *Intelligent Quality of Service Technologies and Network Management: Models for Enhancing Communication.* IGI Global, Hershey, Pennsylvania, USA, ch. QoS Routing and Management in Backbone Networks. To appear (January 2010).

- Bertrand, G., and Texier, G. *Mobility Management and Quality-of-Service for Heterogeneous Networks.* River Publishers, Gistrup, Denmark, ISBN: 978-87-92329-20-2, ch. Ad-Hoc Recursive PCE-Based Inter-Domain Path Computation (ARPC) Methods. (April 2009).

International Book Series

- Bertrand, G., Lahoud, S., Texier, G., and Molnár, M. A Distributed Exact Solution to Compute Inter-Domain Multi-Constrained Paths. In *EUNICE—The Internet of the Future*, Lecture Notes in Computer Science, volume 5733, (September 2009), Springer.

International Conferences

- Bertrand, G., Lahoud, S., Texier, G., and Molnár, M. Computation of Multi-Constrained Paths in Multi-Domain MPLS-TE Networks. In *Next Generation Internet Networks (NGI)*, Aveiro, Portugal (July 2009), IEEE Press, p. 1–8.

- Bertrand, G., and Texier, G. Ad-hoc Recursive PCE Based Inter-domain Path Computation (ARPC) Methods. In *Fifth International Working Conference on Performance Modelling and Evaluation of Heterogeneous Networks (Het-Nets)*, Karlskrona, Sweden (February 2008). **(Selected to be published as a book chapter)**

- Bertrand, G., and Texier, G. DiffServ-Aware Flow Admission Control and Resource Allocation Modeling. In *EuroFGI Workshop on IP QoS and Traffic Control*, Lisbon, Portugal (December 2007).

- Cauvin, A., Niger, P., Orozco, J., and Bertrand, G. An optical burst switched network architecture with passive access. In *Proceedings of the Third International Conference on Networking and Services (ICNS)* (June 2007), IEEE Computer Society, p. 97. (**Best paper award**)

National Conferences

- Bertrand, G., and Texier, G. Intégration du routage PCE aux réseaux de prochaine génération avec IMS. In *Journées Doctorales en Informatique et Réseaux (JDIR), Villeneuve d'Ascq, France* (January 2008).

Research Reports and Project Deliverables

- Frikha, A., Bertrand, G., and Lahoud, S. Pré-calcul de chemins inter-domaines soumis à plusieurs contraintes de qualité de service. Tech. Rep. 1935, IRISA, ISSN 2102-6327, 2009.

- NextTV4All Project, SP2 L2.2a, Impacts généraux sur l'architecture. Abdelkarim Najah (Ed.). (October 2009).

- Bertrand, G., Lahoud, S., Molnár, M., and Texier, G. Inter-Domain Path Computation with Multiple Constraints. Tech. Rep. 1902, IRISA, ISSN 1166-8687, 2008. `http://hal.inria.fr/inria-00319401`.

- VoD@IMS Project, SP2 Quality of Service in Next-Generation Networks. Géraldine Texier (Ed.). (July 2007).

Posters

- Bellabas, A., Bertrand, G., Lahoud, S., Texier, G., and Molnár, M. Routage avec qualité de service dans les réseaux de prochaine génération. In *Colloque Francophone en Ingénierie des Protocoles (CFIP), Strasbourg, France* (October 2009).

- Bertrand, G., Texier, G., Bellabas, A., Lahoud, S., and Molnár, M. Quality of Service Routing in Next-Generation Networks. In *NEM Summit — Towards Future Media Internet, Saint Malo, France* (September 2009).

- Bertrand, G., Texier, G., Lahoud, S., and Molnár, M. Recent Advances in Multi-Constrained Routing for Multi-Domain Networks. In *Trilogy - Architecting the Future Internet, Louvain la neuve, Belgium* (August 2009).

Presentations

- Bertrand, G., Lahoud, S., Molnár, M., and Texier, G. Calcul de chemins inter-domaines soumis à plusieurs contraintes. Journées Automnales ResCom, Strasbourg (October 9–10, 2008).

VU :

Le Directeur de Thèse

VU :

Le Responsable de l'École Doctorale

VU pour autorisation de soutenance

Rennes, le

Le Président de l'Université de Rennes 1

VU après soutenance pour autorisation de publication :

Le Président de Jury,