Calcul de routages équitables dans les réseaux Internet

Nhat Linh Doan, Dritan Nace.
Laboratoire Heudiasyc UMR CNRS 6599,
Université de Technologie de Compiègne,
60205 Compiègne Cedex, France.

***Abstract*— In this paper we consider the problem of fair routing in multicommodity networks. We present here an algorithm for calculating fair routing in a network where the available resources are shared among competing flows according to a max-min share criterion. Our main interest is computing optimal routing paths with regard to max-min fairness, in stable and known traffic conditions. It is a linear programming based approach which permits a lexicographical maximization of vector of fair-share attributed to the connections competing for network resources.**

***Index Terms*— fair splittable routing, max-min fairness, linear programming.**

I. Introduction

Tous les opérateurs de télécommunication doivent faire face à des problèmes relatifs à la croissance du trafic sur Internet et les congestions associées. Il a été remarqué à plusieurs reprises que TCP permet un contrôle efficace des congestions. De plus, il conditionne le partage des bandes passantes parmi les flots traversant le réseau. Ainsi, TCP applique un contrôle de congestion de bout en bout et il a été prouvé dans une étude très connu de Chiu et Jain [1], que le protocole AIMD (Additive Increase, Multiplicative Decrease), sur lequel est basé TCP, permet de converger à un point d'équilibre connu sous le nom de max-min fairness sous certaines hypothèses. En effet, les auteurs ont démontré que AIMD est favorable à l'égard du partage équitable et qu'il converge bien à un tel point de partage équitable dans le cas d'un réseau ne comportant qu'un seul lien de congestion. Rappelons que pour max-min fairness, les ressources sont partagées de façon équitable entre les connexions tel qu'on ne puisse pas affecter des ressources supplémentaires à une connexion sans diminuer les ressources affectées à une autre connexion moins bien servie.

Or, dans l'Internet, le partage des ressources n'est pas tout à fait équitablement. En effet, les débits affectés aux flots sont dépendant du RTT (Round Trip Time) : plus le flot suit un chemin long, moins il sera servi. Ainsi, dans une étude plus récente de Kelly et al. ([2]), il a été démontré que le partage des ressources tente plutôt vers un point d'équilibre connu sous le nom d'équité proportionnelle. Il semblerait que ce dernier résultat ne décrit pas non plus de façon précise le comportement de TCP et d'autres raffinements ont été apportés récemment. Toutefois, il est évident que le TCP impacte d'une certaine façon le partage des ressources. Si TCP "détermine" les débits affectés aux flots, un bon moyen d'intervenir pour une meilleure distribution et utilisation des ressources est de router les flots en tenant compte du comportement de TCP. En effet, même en dehors de toute considération d'équitabilité, la façon de router les connexions impacte fortement le partage des ressources. Nous proposons donc un modèle de calcul pour un routage optimal du point de vue "max-min fairness" dans les réseaux TCP/IP. L'intérêt de cette étude ne réside pas seulement dans le fait de "lier" la couche réseau (protocole de routage) avec la couche transport (TCP), mais il permet d'étudier et comparer la quantité des ressources nécessaires pour assurer un partage équitables des ressources entre les utilisateurs avec celle permettant de maximiser le trafic sortant.

Cette recherche est financée par France Telecom R&D. Voici des courriers électroniques pour prendre contact avec nous : {doannhat,nace}@hds.utc.fr

A. Etat de l'art :

Le problème d'évitement des congestions a été très peu étudié du point de vue du routage. Parmi les travaux les plus intéressants, on peut citer surtout ceux de Q. Ma ([3]) et S.Boulahia ([4]), ou par S. Chen et K. Nahrstedt ([5]). Ces derniers donnent un algorithme permettant de calculer le routage pour un nouveau flot de sorte que le nouveau " min fair share " soit maximisé. Il semble que cet algorithme trouve le chemin de min-fair-share (partage équitable) maximale, donc il privilégie le throughput (le trafic acheminé), du dernier flot à router. Un algorithme similaire (SW) a été démontré par les travaux de S.Boulahia et de Q. Ma., comme ayant des performances plutôt médiocres. Ces derniers ont étudié divers algorithmes de routage afin de comparer leurs performances sous divers scénarios de charge, de topologie et également en cas de réseau multiservice.

La plupart de ces études tentent d'optimiser le routage IP en utilisant les min-fair-share (les parts de bandes passantes offertes à chaque flot) comme des couts/poids des liens. A chaque nouveau flot, on associe le meilleur chemin correspondant à un critère déterminé (le plus court des plus larges , le plus large parmi des plus courts, d'utilité maximale, de coût minimal, etc..) et cela en fonction de l'état du réseau qui est supposé connu. Il s'agit donc de routage dynamique car utilisant des chemins calculés en temps réel et répondant ainsi au mieux à l'état du réseau. Cette démarche opte pour une optimisation locale, en fonction de l'état du réseau. Des travaux menés par S. Boulahia, ainsi que notre propre expérience sur le sujet ont démontré que certains algorithmes, plus particulièrement celui appelé Maximum Utility (d'utilité maximale), présentent

de bonnes performances ce qui prouve la capacité de ces algorithmes à utiliser les ressources de façon rentable au delà de court terme. Néanmoins, cela ne prouve aucunement les bonnes propriétés globales de ces algorithmes de routage. En effet les performances des algorithmes sont comparées entre elles (et pas avec le routage optimal), sur des critères communs. Pour évaluer l'écart entre un routage optimal et le routage déterminé par ces algorithmes de routage, il faudrait que pour chaque nouveau flot, re-arranger le routage des flots existants dans le réseau (donc remettre en cause toutes les routes affectées aux flots). Cette tâche semble assez difficile vu la lourdeur et la complexité des calculs. De plus en pratique elle est irréalisable.

Megiddo [6], a proposé une méthode élégante pour calculer de façon optimale des flots (multi-routés) dans un réseau avec une source et multiple destinations. D'autres travaux ([2], [7], [8], etc.) se sont également intéressés au problème du calcul des chemins sous des conditions de "fairness". Les auteurs de [2] étudient le cas de flots mono-routable dans un réseau avec source unique et multiple destinations et montrent que le problème est NP-complet, d'où l'intérêt d'étudier des méthodes approchées ou exactes dans certains cas polynomiaux. Cette démonstration a été faite grâce à une " transposition " du problème de routage équitable en tant que problème d'affectation équitable de ressources (capacités des liens) aux jobs (flots). L'équivalence des deux problèmes a montré la " NP complétude " du problème de départ. Toutefois ces travaux ont un intérêt limité dû au cadre de leur utilisation : routage mono-source. D'autres études, souvent inspirées des applications dans les télécommunications ont été faites ces dernières années. On peut citer Fordor et al. ([9]) ou Georgiadis et al. ([10]) qui toutefois ou bien se contente d'une heuristique ([9]) ou étudie le problème dans un cadre limité, à savoir celui des réseaux de communication avec une seule commodité, ([10]). Galtier [11] a posé le problème en tant que programme semi définie positive permettant de traiter plusieurs types d'équité.

Remarquons enfin que pour un routage donné, l'allocation des ressources est unique. Cela pourrait être donnée par l'approche suivant ([12]) :

Augmenter graduellement de façon égale la bande passante aux connexions jusqu'à ce qu'il y a un lien saturé. Figer les connexions traversant ce lien et continuer ainsi avec le reste jusqu'à ce que toutes les connexions soient figées.

Cela démontre également l'intérêt d'intervenir au niveau de routage afin d'équilibrer la distribution des ressources ce qui est notre motivation principale. D'autres approches permettant de calculer le partage min-max pour un routage donné sont données au [9], [13], [14], etc.

B. Contribution de ce travail :

Nous nous proposons d'étudier et de concevoir un modèle basé sur la programmation linéaire, qui permet de calculer les routes à affecter aux flots dans le réseau afin de maximiser le débit minimal affecté aux flots ainsi que la somme des débits affectés. Dans cette étude, nous supposons connues les tendances des demandes de trafic dans un réseau TCP/IP. Cela sera modélisé par le nombre de connexions simultanées pour une paire source-destination. En pratique, les données peuvent être obtenues par exemple grâce à un historique des connexions. On peut en effet supposer que les rapports des demandes ne changent pas brutalement sous divers scénarios de charge dans le réseau. Ces hypothèses permettent de situer dans un premier temps le problème dans un cadre statique. Une évolution dynamique pourrait faire l'objet d'une étude ultérieure.

Le papier est organisé comme suit. Dans la section 2 nous donnons un modèle mathématique et un algorithme pour calculer le routage équitable optimal par la programmation linéaire. Ensuite, dans la section 3, nous présentons quelques résultats numériques. Dans la section finale nous tirons quelques conclusions et traçons les grandes lignes des directions de recherche dans le futur.

II. Modéliser le problème de routage équitable :

Dans cette section, nous étudions le problème de calcul du routage sous condition de partage max-min (max-min fairness) avec l'objectif : l'équité de partage de ressources. Tout d'abord, nous introduisons quelques définitions et notations utiles pour le reste de l'article.

A. Définitions et Notations

1) Définitions: Nous introduisons la notion du vecteur d'allocation équitable R, dont les coordonnées donnent dans l'ordre lexicographique les taux d'allocation de routes (voir [15]).

Mathématiquement, cela peut être exprimé comme suivante :

Soit P est l'ensemble de chemins (ou connexions identifiées par des chemins) p dans un graphe $G(N, L)$ et x_p est son taux associé. Une allocation $x = \{x_p\}$ est dite faisable si $x_p \geq 0$ et toutes les contraintes de capacité sont satisfaites. Alors, l'allocation x respecte le "partage max-min" (max min fairness) s'il est faisable et pour toute autre allocation faisable y telle que : $\exists p \in P : y_p > x_p$, alors $\exists s \in P : y_s < x_s \leq x_p$. Le vecteur d'allocation R, correspondant à l'allocation faisable x, contient au plus $|L|$ coordonnées de valeur distincte, chacune exprime la bande passante maximale offerte à une connexion par les liens du réseau.

En effet, en assumant que les taux d'accès sont illimités, nous pouvons affirmer que chaque chemin est congestionné à un de ses liens, alors son taux d'accès est égal au minimum de la bande passante offerte par des liens appartenant au chemin. Par conséquence, la première coordonnée $R[1]$ donne la part minimale offerte à toutes les connexions dans le réseau. Et chaque $R[i]$ représente la i^{ieme} (dans l'ordre croissant) part offerte aux connexions. Nous proposons de calculer le routage correspondant au vecteur d'allocation R^* en maximisant lexicographiquement l'ensemble de vecteurs possibles. Alors, R^* est le vecteur d'allocation équitable que nous voulons obtenir.

Nous distinguons ici deux cas : multi-routage et mono-routage. Dans le premier cas, une seule demande peut être acheminée par plusieurs routes simultanément. Par contre, le deuxième cas requiert que chaque demande soit servie par une seule route. Le dernier problème est NP-complet. Ici, nous traitons seulement le problème de multi-routage.

2) Notations: Avant de décrire l'algorithme et de présenter en détail le modèle mathématique, nous donnons les notations suivantes :

- Un réseau de télécommunication est présenté par un graphe non-orienté $G(N, L)$. Il comprend N routeurs présentés comme noeuds et L liens. La capacité du lien k est notée par C_k.
- D est un ensemble de demandes d à acheminer.
- T_d est le trafic requis de la demande d, mesuré par le nombre de connexions. $x_{j,d}$ est le nombre de connexions cheminées dans la route j pour la demande d. Notons que $k \in j$ veut dire le lien k est inclus dans la route j.
- S_k est la bande passante maximale offerte à une connexion traversant le lien k.
- $R[]$ est le vecteur d'allocation qu'on veut calculer, où $R[i]$ correspond à i^{ieme} coordonnée.
- K est l'ensemble de liens non-saturés
- $L \backslash K$ l'ensemble de liens saturés
- f_k est le nombre de connexions traversant le lien k et ne traversant aucun lien saturé.

Nous appelons un lien saturé (respectivement non-saturé) si sa capacité est totalement partagée par des connexions le traversant (respectivement s'il y reste de bande passante disponible). Dans la partie suivante nous proposons un algorithme pour calculer le vecteur d'allocation équitable et le routage correspondant.

B. Calcul du routage équitable optimal

Dans cette section, un algorithme appelé MFSR (Max-Min Fair Splittable Routing) est présenté pour résoudre le problème du routage équitable. Nous considérons les flots TCP et supposons que le taux d'accès est illimité.

Algorithm 1 MFSR (Max-Min Fair Splittable Routing)

(I) Initialisation :
$K \Leftarrow L$; $p \Leftarrow 1$;
Résoudre le problème initial P_1 ; {donné dans la suite}
$R[1] \Leftarrow 1/B$; $S_k \Leftarrow 1/B$; {S_k correspondant à l'arc saturé k, on a $B = f_k/C_k$}
(II) Calculer $R[i]$ **:**
tant que $K \neq L$ et $\exists d \in D$ non entièrement routée sur $L \backslash K$ **faire**
  $i \Leftarrow i + 1$;
  $R[i] \Leftarrow R[i-1]$;
  tant que $\epsilon > 0$ **faire** {Les valeurs de ϵ successives sont strictement décroissantes}
    Résoudre le problème P_i ; {donné dans la suite}
    Mettre à jour $R[i]$;
  fin tant que
  Calculer MFC_i ; {MFC_i n'est jamais vide}
  $S_k \Leftarrow R[i]$; { $k \in MFC_i$}
  $K \Leftarrow K \backslash MFC_i$;
fin tant que
(III) Obtenir le routage : Les routes obtenues à la dernière itération ont la propriété d'assurer le partage équitable des ressources.

Nous précisons que les procédures utilisées pour résoudre P_1, P_i et pour calculer MFC_i, citées dans l'algorithme MFRS sont données ci-dessous. Dans l'algorithme MFSR on veut calculer le vecteur d'allocation R. D'abord nous calculons $R[1]$. A chaque itération de étape (II), nous tentons de maximiser la valeur de $R[i]$ courant. Nous initialisons $R[i] = R[i-1]$, pour $i \geq 2$, et nous l'augmentons par une valeur égale à la capacité résiduelle minimale de liens non-saturés et recalculons jusqu'à ce que ε devient zéro. A ce moment là, $R[i]$ a atteint la bande passante maximale offerte aux connexions qui traversent seulement des liens non-saturés. Notons que $R[1]$ est calculé différemment de $R[i]$ ($i > 1$), donc il y a deux formulations différentes pour P_1 et P_i.

1) Le problème P_1: Le problème P_1 peut être modélisé comme un problème de multiflot par la programmation linéaire en utilisant une formulation arc-chemin qui permet d'avoir plus de facilité et de flexibilité en considérant des contraintes particulières comme celles de longueur limitée du chemin. Les multiflots sont représentés par une matrice de trafic. Supposons que F soit la bande passante minimale affectée à chaque flot qu'on doit maximiser. Alors nous obtenons :

$$Maximiser \quad F$$

$$\forall k \in L, \quad F * f_k \leq C_k \qquad (1)$$

$$\forall d \in D, \quad \sum_{j \in h(d)} x_{j,d} = T_d \qquad (2)$$

$$\forall k \in L, \quad \sum_{j \in h(d), d \in D, k \in j} x_{j,d} = f_k \qquad (3)$$

$$\forall d \in D, \forall j \in h(d) \quad x_{j,d} \in \mathbb{N}^* \qquad (4)$$

Cette formulation n'est pas linéaire à cause de l'ensemble de contraintes (1). En effet, la variable F et f_k ne sont pas linéairement dépendent. Donc nous remplaçons F par $1/B$, où B donne le délai potentiel (voir [16]). La fonction objectif devient : Minimiser B et l'ensemble de contraintes (1) est remplacé par (5) données ci-dessous :

$$\forall k \in L, \quad f_k - B * C_k \leq 0 \qquad (5)$$

Remplaçons f_k par (3) et nous obtenons le problème P_1 suivant :

$$Minimiser \quad B$$

$$\forall k \in L, \quad B * C_k - \sum_{j \in h(d), d \in D, k \in j} x_{j,d} \geq 0 \ (\omega_k) \qquad (6)$$

$$\forall d \in D, \quad \sum_{j \in h(d)} x_{j,d} = T_d \ (\pi_d) \qquad (7)$$

$$\forall d \in D, \quad \forall j \in h(d), \quad x_{j,d} \in \mathbb{R}^+ \qquad (8)$$

où ω_k (respectivement π_d) sont des coefficients duaux de contraintes (6) (respectivement les contraintes (7)). Les variables du problème P_1 sont $x_{j,d}$ et B . Les contraintes (6) sont des contraintes de capacité. Et les contraintes de trafic (7) assurent que toutes les connexions de flot soient routées. Par cette étape, nous assurons l'équité du partage de bande passante dans un premier niveau. En fait, minimiser la variable B permet de maximiser le partage minimal pour chaque flot. Notons que les contraintes (8) ne requièrent pas un nombre entier de connexions, c'est à dire $x_{j,d}$ est une variable continue. En réalité, un grand nombre de flots partage les ressources

du réseau simultanément, et dans un tel environnement dynamique, il est bien plus important de s'intéresser à la proportion du trafic routé par chaque chemin.

Finalement, nous décrivons comment nous procédons pour résoudre P_1. Clairement, nous ne pouvons pas énumérer tous les chemins possibles dans le réseau. Ainsi nous utilisons un sous-ensemble de chemins. Puis nous résolvons le modèle par la méthode du simplex. Cependant, la solution n'est généralement pas optimale. Il nous faut ensuite utiliser la méthode de génération des colonnes : nous construisons des nouveaux chemins sur le graphe initial où la longueur de liens k est égal à ω_k. En fait, le coût réduit associé à chaque colonne (chemin $j \in h(d)$) est donné par $\sum_{k\in j} \omega_k - \pi_d$. Rappelons que selon l'algorithme du simplex, un chemin candidat pour entrer dans la base doit avoir un coût réduit négatif ou nul. Par conséquence, nous devons chercher pour chaque demande d, le nouveau chemin j qui minimise ce coût réduit, et en fait minimise $\sum_{k\in j} \omega_k$. Cela correspond à chercher le plus court chemin, dans le graphe au sens de ω_k. Ce chemin correspond à une nouvelle colonne qui doit être ajoutée au problème. Nous continuons à modifier le problème (ajouter des nouveaux chemins), le résolvons, et recommençons à nouveau la génération des colonnes jusqu'à il n'y a plus de tels chemins. La solution du problème à la dernière itération est optimale.

Dans l'implémentation de cet algorithme, pour chercher les chemins plus courts, nous avons utilisé l'algorithme de Dijkstra en cas général, ou l'algorithme de Bellman-Ford (version modifiée) en cas de contrainte de longueur de chemin.

2) Le problème P_i: Nous utilisons aussi une formulation arc-chemin. Notons que dans la formulation ci-dessous $f_{k,k'}$ exprime le nombre de connexions traversant simultanément par les liens k et k' et saturées au lien k'.

$$Maximiser \quad \epsilon$$

$$\forall k \in K, \sum_{k'\in L\backslash K} f_{k,k'} * S_{k'} + f_k * R[i] + \epsilon \leq C_k\ (\omega_k) \quad (9)$$

$$\forall k \in L\backslash K, \sum_{k'\in L\backslash K,\ k'<k} f_{k,k'} * S_{k'} + f_k * S_k = C_k\ (\omega_k) \quad (10)$$

$$\forall d \in D, \sum_{j\in h(d)} x_{j,d} = T_d\ (\varphi_d) \quad (11)$$

$$\forall k \in K, \sum_{\substack{j\in h(d), d\in D, k\in j,\\ s.t.\{(L\backslash K)\cap j\}=\phi}} x_{j,d} = f_k \quad (12)$$

$$\forall k \in L\backslash K, \sum_{\substack{j\in h(d), d\in D, k\in j,\\ s.t.\forall k'\in\{(L\backslash K)\cap j\},\\ S[k]<S[k']}} x_{j,d} = f_k \quad (13)$$

$$\forall k \in K, \sum_{\substack{j\in h(d),\ d\in D,\ k,k'\in j,\\ s.t.\forall k"\in\{(L\backslash K)\cap j\},\\ S_{k'}<S_{k"}}} x_{j,d} = f_{k,k'} \quad (14)$$

$$\forall d \in D, \quad \forall j \in h(d) \quad x_{j,d} \in \mathbb{R}^+ \quad (15)$$

$$\epsilon \geq 0 \quad (16)$$

Les variables du problème P_i sont : $x_{j,d}$, f_k, $f_{k,k'}$ et ϵ. La dernière variable représente le minimum de capacités résiduelles des liens non-saturés. L'ensemble de contraintes (9) et (10) sont des contraintes de capacité correspondant respectivement à liens saturés et non-saturés. Contraintes (11) assurent que toutes les connexions de chaque demande sont entièrement routées. L'ensemble de contraintes (12) (respectivement (13)) exprime le nombre de connexions traversant par le lien non-saturé k (respectivement saturé) et elles ne sont pas saturées sur aucun autre lien situé dans leur chemin. Si nous remplaçons f_k et $f_{k,k'}$ dans (9) et (10) par leur correspondants dans (12), (13) et (14), les coefficients duaux de contraintes (9), (10) (respectivement (11) sont des ω_k (respectivement π_d).

Ici S_k et $R[i]$ ne sont pas considérés comme des variables. Ils sont fixés à chaque itération, et modifiés par la procédure "mettre à jour $R[i]$" dans l'algorithme MFSR. Après avoir obtenu les valeurs des variables (i.e f_k, $f_{k,k'}$) en résolvant le dernier P_i, à l'étape i, nous modifions la valeur $R[i]$ par la formulation suivante :

$$R[i] = \min_{k\in K} \left\{ \frac{C_k - \sum_{k'\in L\backslash K} f_{k,k'} * S_{k'}}{f_k} \right\} \quad (17)$$

Nous résolvons à nouveau P_i et ensuite mettons à jour $R[i]$ de telle façon jusqu'à ce que nous obtenons $\epsilon = 0$.

Le problème de multiflot P_i est résolu de la même façon que P_1. Nous utilisons la méthode de génération des colonnes où des nouvelles colonnes représentent des chemins candidats. Cependant, pour P_i, la génération des colonnes est plus compliquée. Elle correspond au problème du plus court chemin minimisant une fonction de deux paramètres : ω_k et S_k. En fait, le coût réduit pour chaque colonne (chemin $j \in h(d)$) est $-S_{k'} * \sum_{k\in j} \omega_k - \pi_d$. Nous devons chercher les chemins qui minimisent $S_{k'} * \sum_{k\in j} \omega_k + \pi_d$ (ou simplement minimisent $S_{k'} * \sum_{k\in j} \omega_k$), avec $S_{k'} = Min\{S_k, k \in j\}$ et $\forall k \in K, S_k = R[i]$, où $R[i]$ est la dernière coordonnée calculée du vecteur d'allocation R . Contrairement au problème de calcul du chemin le plus court impliquant deux paramètres additifs qui est NP-complet (voit [17]), notre problème de calcul de chemins candidats à entrer dans la base est polynomial. En effet, nous avons implémenté une version modifiée de l'algorithme de Ford pour le résoudre.

3) Calculer MFC_i: En pratique, quand plusieurs liens sont saturés (appelons l'ensemble Q) à la fin de l'itération i, nous ne pouvons pas distinguer ceux qui sont vraiment saturés (appelons MFC_i) d'autres (voir la section suivante). Une fois $\epsilon = 0$, nous calculons MFC_i comme suit :

Algorithm 2 Calculer MFC_i (Min Fair Cut)

$MFC_i \Leftarrow Q$
pour tous liens $j \in Q$ **faire** {Modifier légèrement le problème P_i comme suit}
  Supprimer ϵ dans toutes les contraintes de (9) sauf dans celle du lien j.
  Résoudre le problème courant ;
  si $\epsilon > 0$ **alors**
    $MFC_i \Leftarrow MFC_i\backslash j$;
  fin si
fin pour
A la fin de la procédure nous avons obtenu MFC_i associé avec le P_i

Évidemment, tous les liens inclus dans le même MFC_i offrent la même part de la bande passante aux flots saturés sur eux. Ainsi, à chaque itération d'algorithme MFSR, nous les marquons tous comme des liens saturés k et nous fixons S_k à $R[i]$.

C. Unicité et optimalité :

L'optimalité et unicité de la solution obtenue par notre algorithme itératif ne paraîssent pas évidentes, même en cas de multi-routage. En effet, des multiflots différents peuvent satisfaire toutes les contraintes fixées pour le problème P_i à chaque itération i de l'algorithme. En outre, plusieurs liens distincts pourraient probablement être saturés par des multiflots différents. Rappelons aussi qu'à chaque itération de l'algorithme MFSR nous obtenons la i^{ieme} coordonnée du vecteur d'allocation équitable. Une fois calculée, cette valeur est fixée pour tous les liens $k \in MFC_i$ correspondant et utilisée comme une constante pour le reste des calculs.

Lemme 1: Chaque multiflot obtenu lors de l'itération i de l'algorithme MFSR est saturé simultanément dans un ensemble unique de liens, appelé MFC_i.

Preuve : Considérons l'ensemble de multiflots satisfaisant toutes les contraintes associées au problème P$_i$. Evidemment, chaque multiflot est saturé dans au moins un lien. Nous définissons MFC_i associé au problème P_i comme l'intersection de tous les ensembles de liens saturés. Cet ensemble n'est pas vide. En effet, supposons par l'absurdité qu'à la fin de l'étape i, quand $\epsilon = 0$, il existe deux multiflots Φ_1 et Φ_2 satisfaisant toutes les contraintes relatives à P_i, et la part maximale offerte aux connexions est fixée à $R[i]$. Ces multiflots sont tel que $K_1 \subset$ K (respectivement $K_2 \subset K$) représente l'ensemble de liens saturés dans K pour Φ_1 (resp. Φ_2) et $K_1 \cap K_2 = \emptyset$; Notons Φ= 1/2($\Phi_1 + \Phi_2$). Le multiflot Φ satisfait toutes les contraintes de capacité et de trafic associées avec le problème P_i. De plus, les liens dans K ne sont pas saturés (ε reçoit une valeur strictement positive), qui contredit le fait que Φ_1 ou Φ_2 sont obtenus à la fin de l'étape i. Il peut être facilement montré que ceci est valable même pour le problème P_1. Nous avons prouvé par conséquent le lemme ci-dessus.

La unicité et l'optimalité de la solution obtenue peuvent être données par le théorème suivant :

Théorèm 1: Le vecteur d'allocation et le routage correspondant obtenu à la fin de l'algorithme MFRSR sont max-min équitables.

Preuve : Nous notons R^* (respectivement R) le vecteur d'allocation optimal (resp. le vecteur d'allocation obtenu par MFSR), et Φ^* (respectivement.Φ) le multiflot correspondant. Supposons par l'absurdité que $\exists i$ tel que $R^*[i] > R[i]$ et $\forall j < i$, nous avons $R^*[j] = R[j]$. Il peut être vérifié facilement que le multiflot Φ satisfait toutes les contraintes du problème P_1. Ceci prouve qu'il est contraint dans tous les liens du MFC_1correspondant. De même, il est claire que la propriété citée ci-dessus est aussi vrai pour tous les P_j. Selon la lemme 1, le multiflot Φ^* est saturé dans tous les liens identiques des ensembles consécutifs MFC_j comme Φ. Par conséquent, Φ^* satisfait aussi toutes les contraintes du problème P_i, et il ne pourrait pas donner une meilleure borne pour $R[i]$, ce qui prouve que $R^*[i] = R[i]$.

Réseau	Noeuds	Arcs	Demandes
RES_11	11	23	55
RES_15	15	105	105
RES_26	26	43	264
RES_41	41	70	319

TABLE I
DESCRIPTION DE RÉSEAUX DE TEST

Longueur Max.	RES_11	RES_15	RES_26	RES_41
5	4.60	761.29	-	241.64
6	5.30	999.28	100.16	433.55
7	6.28	1218.92	127.88	486.85
10	8.65	1767.48	219.79	922.40
illimité	8.71	2129.76	272.35	1125.42

TABLE II
TEMPS DE CALCUL (EN SECONDES)

III. RÉSULTATS NUMÉRIQUES

Notre approche a été implémentée en C++ et CPLEX 7.1. Tous les tests ont été exécutés sur une machine avec la configuration suivante : SUN 4000, SOLARIS 2.5, 6 processeurs d'UltraSparc 167Mhz, 256Mb.

Dans le tableau I, nous résumons les caractéristiques principales de quatre exemples de réseau que nous avons utilisés pour nos calculs de routage équitable. Dans le deuxième tableau (II) nous présentons les temps de calcul. Notons que le temps de calcul dépend essentiellement de la taille du réseau et le nombre de demandes. La tâche la plus longue est la génération de colonne.

Nous rappelons que dans notre modèle nous ne cherchons pas à maximiser le débit global du réseau. Il est néanmoins intéressant de comparer le débit global avec le débit sortant total du routage équitable. Quelques comparaisons sont présentées dans le Tableau III. La colonne " Fair Throughput " représente le débit global de toutes les connexions routées à travers les chemins équitables. La colonne "Max. Throughput " donne le débit global maximum de réseau. La différence entre "Fair Throughput" et "Max. Throughput" est donnée dans la colonne "Gap" . La dernière colonne est calculé comme suivant : $\frac{(\text{“}Max.Throughput''--\text{“}FairThroughput'')*100}{\text{“}Max.Throughput''}$. Comme nous l'expliquons dans la prochaine section, beaucoup de demandes sont mono-routées, et nous donnons le taux de telles demandes dans la colonne "Mono Routée".

IV. CONCLUSION

A. Sommaire et discussion :

Dans cet article nous avons présenté un algorithme itératif pour calculer le routage équitable sous condition de trafic stable et connu. C'est une approche qui permet une maximisation lexicographique du vecteur d'allocation. Notons que des contraintes de taux d'accès (supposé illimité) peuvent être facilement intégrées dans notre modèle. Ce problème de routage

Réseau	Fair Thrghpt.	Max Thrghpt.	Gap	Mono-Routée
RES_11	469.698	698	32.71%	78.18%
RES_15	36780.4	42000	12.43%	59.05%
RES_26	4690.38	10280	54.37%	88.64%
RES_41	1094.19	2296	52.34%	88.71%

TABLE III

COMPARAISON LE THROUGHPUT MAXIMAL DU ROUTAGE ÉQUITABLE ET LE THROUGHPUT MAXIMAL NORMAL ET LE TAUX DE DEMANDES MONO-ROUTÉES.

peut aussi être modélisé par une formulation arc-noeud, mais nous avons choisi une formulation arc-chemin qui permet plus de facilité et flexibilité pour considérer des contraintes particulières du chemin, par exemple limitation de longueur de chemin. En outre cette formulation permet de réduire le nombre de routes obtenues (environ $|D| + |L|$, où $|D|$ est le cardinal de l'ensemble de demandes, et $|L|$ donne le nombre de liens). Par conséquent, seulement une certaine proportion de demandes sera multi-routée (généralement ayant deux routes), et le reste sera mono-routée.

Dans un rapport ([18]), nous avons également proposé un algorithme plus simple pour montrer que le problème de multiflot équitable en cas de multi-routage est polynomial.

B. Direction de recherche future

Jusqu'à maintenant nous nous sommes concentrés sur le cas de partage max-min qui à certains égards limite la portée de cette approche. Cependant, la réalité est beaucoup plus complexe. Ainsi, le taux d'accès assigné aux connexions est inversement proportionnelle à la valeur de RTT (Round Trip Time). Une direction de recherche future est d'étudier les autres variantes de l'équité c'est-à-dire l'équité proportionnelle. Une autre direction de recherche est de considérer le cas de mono-routage qui est NP-complet. En outre, les résultats théoriques développés ne sont pas applicables pour ce problème. En effet, il peut facilement montrer que pour le cas de mono-routage, les différents routages peuvent offrir la même équité pour des liens distincts.

REFERENCES

[1] D. Chiu and R. Jain. Analysis of the increase and decrease algorithms for congestion avoidance in computer networks. *Computer Networks and ISDN Systems*, 17 :1–14, 1989.

[2] F. P. Kelly, A. Maulloo, and D. Tan. Rate control in communication networks : shadow prices, proportional fairness and stability. *Journal of the Operational Research Society*, vol. 49, pp. 237-252, 1998.

[3] Q. Ma. *Quality-of-Service Routing in Integrated Service Networks*. PhD thesis, Carnegie Mellon Univ., Pittsburgh, USA, Jan. 1998.

[4] S. Boulahia-Oueslati. *Qualité de service et routage des flots élastiques dans un réseau multiservice*. PhD thesis, ENST, Paris, France, November 2000.

[5] S. Chen and K. Nahrstedt. Maxmin fair routing in connection-oriented networks. Tech. report, Dept. Comp. Science, UIUC, USA, 1998.

[6] M. Megiddo. Optimal flows in networks with sources and sinks. *Mathematical Programming*, 7, 1974.

[7] A. Goel, A. Meyerson, and S. Plotkin. Combining fairness with throughput : Online routing with multiple objectives. In *Proc. 32nd ACM STOC*, 2000.

[8] A. Kumar and J. Kleinberg. Fairness measures for resource allocation. *Proceedings of 41st IEEE Symposium on Foundations of Computer Science*, 2000.

[9] G.Fodor, G.Malicskó, M. Pióro and T. Szymanski. Path Optimization for Elastic Traffic under Fairness Constraints. In *Proc. of ITC 2001*, 2001.

[10] L. Georgiadis, P. Georgatsos, K. Floros , S. Sartzetakis. Lexicographically Optimal Balanced Networks. In *Proc. of ACM INFOCOM'01*, pp. 687-698, 2001.

[11] J. Galtier. Semi-definite programming as a simple extension to linear programming : convex optimization with queueing, equity and other telecom functionals. In *Proc. of AlgoTel 2001*. INRIA, 2001.

[12] A. Bertsekas and R. Gallager. *Data Networks.* Prentice-Hall, Engelwood Cliffs, N.J., 1992.

[13] A. Charny, D. D. Clark and R. Jain. Congestion control with explicit rate indication. In *Proc. of IEEE International Conference on Communications*, 1995.

[14] D. Nace. A linear programming based approach for computing optimal splittable fair routing in *Proc. of the Seventh IEEE Symposium on Computers and Communications 2002 (ISCC'02)*, pp. 468-474, Taormina-Giardini Naxos, Italie, 2002.

[15] J. Kleinberg, Y. Rabani, and E. Tardos. Fairness in routing and load balancing. In *Proc. of the 35th Annual Symposium on Foundations of Computer Science*, 1999.

[16] L. Massoulié and J.W. Roberts. Bandwidth sharing : objectives and algorithms. In *Proc. of IEEE INFOCOM'99*, 1999.

[17] Z. Wang and J. Crowcroft. Quality of service routing for supporting multimedia applications. *IEEE Journal on Selected Areas in Communications.* Vol. 14, pp. 1228-1234, Sept. 1996.

[18] D. Nace. L.N. Doan. A polynominal approach to the fair multi-flow problem, rapport interne, Laboratoire Heudiasyc, UTC, novembre 2002.