Rapport TP MLA (Barreaux Alexis - Le Bozec-Chiffoleau Sulian)

Exercice 2

Question 1

On peut voir dans notre fichier de résultat (sommaire.txt) que la version relaxée nécessite en général moins d'itérations d'ajouts de coupes pour le cas bnd = 1. Il n'y a que pour benders1 qu'il en faut plus et alors seulement une. De plus, les coupes relâchées suffisent dans ce cas et on n'ajoute pas de coupe entière une fois la partie relâchée terminée.

Question 2

De même, on observe que la version relachée nécessite moins d'itérations pour le cas bnd = 3, 35% de moins environ pour benders1. Cela ne semble cependant pas suffisant pour résoudre le problème en temps raisonnable (plus de dix minutes d'exécution sans solution réalisable à la fin sur benders2). En effet, bien que le problème arrive a priori rapidement à une solution dont la valeur est proche de l'optimum, il continue à trouver des coupes violées et les ajoute donc successivement, ralentissant peu à peu le PLNE.

Exercice 3

Question 1

On peut remarquer dans la formulation initiale du problème les contraintes d'inégalité sont en fait toutes saturées pour une solution optimale. Dans le cas où bnd = 1, pour faire transiter une quantité x (entière) il faut ouvrir exactement x liaisons et qui seront saturées. Ainsi pour ouvrir un minimum de liaisons il faut passer par un nombre minimum d'arêtes pour chaque terminal, cela revient à calculer les plus courts chemins entre la source et les terminaux.

Question 2

On peut voir que la version algorithmique avec Dijkstra donne de bien meilleurs résultats : on obtient bien la même valeur et ce beaucoup plus rapidement. Les temps d'exécutions renvoyés étaient même parfois nuls ou alors tous égaux aux même valeurs, ce qui laisse à penser que sur ces instances et nos machines les temps étaient négligeables et même difficiles à évaluer avec time().

Autres tests réalisés

Calculs de bornes

Dans le cas où la bande passante est strictement supérieure à 1 on peut exploiter la méthode avec Dijkstra (cas bnd = 1) pour obtenir des bornes inférieure et supérieure de notre problème.

En effet si on note V_b la valeur de l'objectif d'une instance donnée pour bnd = b, on a V_b <= ceil(V_1 / b), où ceil(x) est la partie entière supérieure de x. Cela nous fait donc une borne inférieure du problème.

De plus, considérer uniquement les plus courts chemins pour faire transiter la demande est une solution réalisable du problème qui se ramène à ajouter des liaisons comme si on avait bnd = 1. Calculer le coût correspondant nous permet d'obtenir une borne supérieure du problème.

Nous avons donc ajouté ces bornes dans le problème maître mais cela ne permet pas d'améliorer le temps de calcul pour toutes les instances. En effet sur certaines instances on réduit le temps et sur d'autres on l'augmente. Nous n'avons donc pas conservé ces bornes dans la comparaison des méthodes.

Utilisation du programme non décomposé

Par curiosité, nous avons aussi voulu voir les performances du PLNE brut sans décomposition de Benders sur nos instances, car le problème nous semblait ne pas être si fortement contraint que cela. Cela nous a permis d'obtenir pour bnd = 3 l'optimal en quelques secondes sur benders4, de valeur 101, ce que nous n'arrivions pas à obtenir en des temps raisonnable avec les coupes de benders. Nous avons pour ces résultats ajouté une colonne "WITHOUT BENDERS" dans le sommaire des résultats.

Comparaison des différentes méthodes

Nos résultats sont disponibles dans le fichier sommaire.txt.