31/12/2016

Maxime DEGRES Jean-Baptiste DURIEZ Jordane QUINCY

Rapport de TP

Métaheuristique, problème de sac à dos

# Introduction du problème

# Méthode de résolution

## Structure de données

Afin de modéliser le problème nous avons utilisé la même structure de données que celle fournie en TD pour le sac à dos multidimensionnel, nous avons juste ajouté les contraintes de demandes à la suite des contraintes de capacités dans la matrice ainsi que le nombre de contraintes de demandes.

Pour récapitulatif, le problème est donc représenté par un pointeur (mkp), pointant vers 3 entiers : n, cc et cd, représentant respectivement le nombre de variables, le nombre de contraintes de capacités et le nombre de contraintes de demandes. Ce pointeur pointe également vers la matrice d’entiers « a ». Les lignes de la matrice correspondent aux contraintes, et les colonnes aux variables. Ainsi dans la case a[3][10] on va retrouver la valeur de la 10ème variable pour la 3ème contrainte. En a[0][x] on retrouve l’apport de la variable x pour la fonction objectif. Enfin dans cette matrice, on différencie les contraintes de capacités des contraintes de demandes grâce à l’index. Ainsi les contraintes a[1][x] à a[mkp->cc][x] correspondent à des contraintes de capacités et celles qui suivent à des contraintes de demandes.

Pour la représentation de la solution nous avons également utilisée la représentation donnée en TD, nous avons simplement fait de « slack » un tableau à 2 dimensions pour y ajouter la slack des contraintes de demandes.

Pour récapitulatif, une solution est donc représentée par un pointeur (sol), pointant vers un entier (objValue), un tableau à 1 dimension d’entiers (x) et un tableau à 2 dimensions d’entiers (slack). objValue correspond au résultat de la fonction objectif, x correspond à ce qu’on va retrouver dans le sac. Ainsi x a pour taille le nombre de variables + 1, dans x[0] on a le nombre d’objet mis dans le sac et x[n] vaut 1 si la nième variable est dans le sac et 0 sinon. Enfin slack correspond à ce qui reste pour les contraintes. slack[0] correspond aux contraintes de capacités (taille = nombre de contraintes de capacités + 1) et slack[1] correspond aux contraintes de demandes (taille = nombre de contraintes de demandes + 1). Dans slack[0[0] et slack[1][0] on retrouve le nombre de contraintes non respectées pour la solution en cours et dans slack[0][n] et slack[1][n’] on retrouve ce qu’il nous reste pour la nième et n’ième contraintes. Exemple pour une contrainte de poids (la 2ème contrainte), si on a déjà mis 10 alors qu’on peut mettre en tout 15, alors on va mettre 5 dans slack[0][2]. Ainsi une solution réalisable est une solution pour laquelle slack[0][0] et slack[1][0] vaut tous les deux 0.

# Résultats

Pour tous nos résultats, nous avons utilisé un ordinateur avec 6Go de RAM, un processeur Intel Core i7 à 2,40GHz sous Windows 10.

Afin de vous montrer les meilleurs résultats possibles, pour chaque instance nous avons lancé notre programme plusieurs fois en partant de solution initiale différente, ainsi nous pourrons spécifier à chaque fois avec quelle solution initiale nous avons abouti à la meilleure de nos solutions.

Les solutions initiales ont été obtenues en remplissant le sac (voir méthode de résolution) suivant différentes manières de tri qui peuvent être retrouvé dans le code.

Ce comportement est donc différent du programme final que nous vous avons transmis puisque ce dernier utilise au maximum le temps qu’il a. Donc au sein d’une seule exécution nous utilisons plusieurs solutions initiales afin de trouver la meilleure solution possible en une exécution dans le temps imparti.

## Instance 100Md5\_1\_1pos.txt

Meilleure solution : **33018** en améliorant une solution initiale basée sur « ByCoeffDemandeSurPoids » (donnant une solution initiale à 14694) et en utilisant l’algorithme tabou en parcourant tous les voisins (en prenant le premier meilleur voisin : 32961).

## Instance 100Md5\_1\_2pos.txt

Meilleure solution : **29073** en améliorant une solution initiale basée sur « ByCoeffDemandeSurPoids » (donnant une solution initiale à 14681) et en utilisant l’algorithme tabou en prenant le premier meilleur voisin (même résultat en parcourant tous les voisins).

## Instance 100Md5\_1\_5pos.txt

Meilleure solution : **22130** en améliorant une solution initiale basée sur « ByCoeffDemandeSurPoids » (donnant une solution initiale à 13206) et en utilisant l’algorithme tabou en parcourant tous les voisins (en prenant le meilleur voisin : 22051).

## Instance 100Md5\_2\_1pos.txt

Meilleure solution : **30644** en améliorant une solution initiale basée sur « ByCoeff » (donnant une solution initiale à 29477) et en utilisant l’algorithme tabou en prenant le premier meilleur voisin (même résultat en parcourant tous les voisins).

## Instance 100Md5\_2\_2pos.txt

Meilleure solution : **27879** en améliorant une solution initiale basée sur « ByCoeffPoidsSurDemandePlusValeur » (donnant une solution initiale à 15516) et en utilisant l’algorithme tabou en parcourant tous les voisins (en prenant le meilleur voisin : 27829).

## Instance 100Md5\_2\_5pos.txt

Meilleure solution : **26236** en améliorant une solution initiale basée sur « ByCoeffDemandeSurPoids » (donnant une solution initiale à 16820) et en utilisant l’algorithme tabou en prenant le premier meilleur voisin (même résultat en parcourant tous les voisins).

## Instance 250Md5\_1\_1pos.txt

Meilleure solution : **90085** en améliorant une solution initiale basée sur « ByCoeff » (donnant une solution initiale à 87910) et en utilisant l’algorithme tabou en prenant le premier meilleur voisin (en parcourant tous les voisins : 90073).

## Instance 250Md5\_1\_2pos.txt

Meilleure solution : **79807** en améliorant une solution initiale basée sur « ByCoeffPoidsSurDemandePlusValeur » (donnant une solution initiale à 56501) et en utilisant l’algorithme tabou en parcourant tous les voisins (en prenant le meilleur voisin : 79762).

## Instance 250Md5\_1\_5pos.txt

Meilleure solution : **68045** en améliorant une solution initiale basée sur « ByCoeffDemandeSurPoids » (donnant une solution initiale à 46894) et en utilisant l’algorithme tabou en parcourant tous les voisins (en prenant le meilleur voisin : 67986).

## Instance 250Md5\_2\_1pos.txt

Meilleure solution : **82821** en améliorant une solution initiale basée sur « ByCoeffPoidsSurDemandePlusValeur » (donnant une solution initiale à 59622) et en utilisant l’algorithme tabou en parcourant tous les voisins (en prenant le meilleur voisin : 82664).

## Instance 250Md5\_2\_2pos.txt

Meilleure solution : **76835** en améliorant une solution initiale basée sur « ByCoeff » (donnant une solution initiale à 75010) et en utilisant l’algorithme tabou en parcourant tous les voisins (en prenant le meilleur voisin : 76878).

## Instance 250Md5\_2\_5pos.txt

Meilleure solution : **61651** en améliorant une solution initiale basée sur « ByCoeffPoidsSurDemandePlusValeur » (donnant une solution initiale à 37302) et en utilisant l’algorithme tabou en prenant le meilleur voisin (en parcourant tous les voisins : 61647).

## Instance 500Md5\_1\_1pos.txt

Meilleure solution : **177554** en améliorant une solution initiale basée sur « ByCoeffDemandeSurPoids » (donnant une solution initiale à 82784) et en utilisant l’algorithme tabou en parcourant tous les voisins (en prenant le meilleur voisin : 177394).

## Instance 500Md5\_1\_2pos.txt

Meilleure solution : **146114** en améliorant une solution initiale basée sur « ByCoeffDemandeSurPoids » (donnant une solution initiale à 81643) et en utilisant l’algorithme tabou en parcourant tous les voisins (en prenant le meilleur voisin : 146043).

## Instance 500Md5\_1\_5pos.txt

Meilleure solution : **136437** en améliorant une solution initiale basée sur « ByCoeff » (donnant une solution initiale à 128898) et en utilisant l’algorithme tabou en parcourant tous les voisins (en prenant le meilleur voisin : 136415).

## Instance 500Md5\_2\_1pos.txt

Meilleure solution : **180886** en améliorant une solution initiale basée sur « ByCoeff » (donnant une solution initiale à 175415) et en utilisant l’algorithme tabou en prenant le meilleur voisin (en parcourant tous les voisins : 180839).

## Instance 500Md5\_2\_2pos.txt

Meilleure solution : **161938** en améliorant une solution initiale basée sur « ByCoeffDemandeSurPoids » (donnant une solution initiale à 96009) et en utilisant l’algorithme tabou en prenant le meilleur voisin (en parcourant tous les voisins : 161850).

## Instance 500Md5\_2\_5pos.txt

Meilleure solution : **145044** en améliorant une solution initiale basée sur « ByCoeff » (donnant une solution initiale à 141980) et en utilisant l’algorithme tabou en parcourant tous les voisins (en prenant le meilleur voisin : 145018).

# Organisation au sein de l’équipe