Feuille de TP n^o 2
Routage de véhicules avec fenêtres temporelles

Durée 4 séances (1TPe 1TPne 1TPe 1TPne)

Un problème important consiste à construire à définir les tournées de véhicules devant passer en un certains nombres de points avec des contraintes temporelles : pour chaque point qui doit être servi par un véhicule, on a deux contraintes définissant une heure minimale de passage et une heure maximale de livraison (exprimée dans une unité de temps normalisée). On a aussi des contraintes de capacité : il faut livrer à chaque point des marchandises ayant un certain poids, et chaque véhicule a une contrainte sur le poids total qu'il peut emmener. Le fichier `routage_a1.txt` contient une ligne par client, de la forme :

Id, X, Y, Quantité, Tmin, Tmax, Durée

où Id est un numéro permettant d'identifier les clients, et Durée indique le temps d'arrêt minimal chez le client (temps nécessaire à la livraison). Quantité est le poids de ce qu'il faut livrer.

On suppose qu'on a un nombre illimité de camions disponibles, tous de même capacité précisée sur la première ligne du fichier `routage_a1.txt`. Tous les camions partent du dépot dont la position X,Y est donnée sur la ligne Id=1 du fichier, cette ligne donne également l'heure maximale de retour au dépot.

Il s'agit de minimiser le temps total de parcours – la somme des temps de parcours de tous les camions. Pour simplifier, on assimilera le temps de parcours entre deux clients à la distance euclidienne entre ces deux clients : $\sqrt{(X_1 - X_2)^2 + (Y_1 - Y_2)^2}$.

I Recherche de routes optimales sans contraintes

On vous demande pour commencer d'implémenter un algorithme de recherche local simple – type montée selon la plus grande pente – avec redémarrages, mais sans mémoire. Dans cette partie on ignorera les contraintes de temps des livraisons et de capacité des camions. Il vous faudra donc réaliser les tâches suivantes :

1. Définir un type de données pour représenter les solutions : une façon directe de représenter un état est un ensemble de k camions $C_1, C_2, ..C_k$, chaque camion étant une séquence ordonnée de clients. Les camions doivent former une partition des clients : chaque client est visité une fois et une seule. Le nombre k pourra bien sûr varier.
2. Implémenter un algorithme de génération de solutions initiales aléatoires ;
   Une notion cruciale pour les recherches locales est la notion de voisinage, c'est-à-dire les solutions que l'on peut définir qui sont "proches", en modifiant légèrement une solution. Il faut pouvoir calculer rapidement les voisins d'une solution courante.
3. Réfléchir à la notion de voisinage, et implémenter des fonctions :
   a) permettant d'énumérer tous les voisins d'une solution donnée ;
   b) permettant de tirer au hasard un nombre fixé de voisins d'une solution donnée ;
4. Il ne reste alors plus qu'à implémenter l'algorithme de montée selon la plus grande pente.
5. Le réglage des méthodes meta-heuristiques demande de l'expérimentation pour avoir les bons réglages. Vous devrez notamment faire varier :
   — le nombre de redémarrage,
   — le nombre d'itération et/ou le critère d'arrêt pour une recherche locale,
   — le choix du voisin (le meilleur parmi tous, parmi un échantillon de taille à fixer, le premier qui améliore la solution).

Vous pouvez éventuellement expérimenter plusieurs définitions de voisinage. Vos expérimentations devront être synthétisées dans votre rapport final.

Les expérimentations doivent sauver la meilleure solution et sa valeur, et vous devrez aussi générer une sortie au format dot (cf `http://www.graphviz.org/pdf/dotguide.pdf`) pour visualiser les parcours.

II Utilisation d'une liste tabou

Dans bien des applications, une telle liste permet des améliorations importantes des temps de calcul et des solutions trouvées. Pour implémenter une recherche avec liste tabou, vous devez d'abord réfléchir à ce que vous pourrez mémoriser dans une telle liste – il est en général peu efficace de mémoriser des solutions récentes en entier, il vaut en général mieux mémoriser certaines caractéristiques de ces solutions, faciles à tester sur les nouvelles solutions générées. Quelle caractéristiques pourrait-on mémoriser ici ? Expérimentez avec différentes tailles de liste tabou.

III Prise en compte des contraintes

Implémentez finalement une recherche locale prenant en compte les contraintes de temps et de capacité : un camion ne doit pas transporter plus que sa capacité maximale ; si on arrive trop tôt chez un client, il faut attendre ; le temps d'arriver doit être suffisamment tôt pour que la fin de la livraison soit avant l'heure maximum de livraison.