ScatterSearch

Résumé de la stratégie de résolution du SADMD

ScatterSearch

Metaheuristique utilisée : ScatterSearch.

ScatterSearch a été développé par Glover, qui a donné un "template" en 1998 pour créer une ScatterSearch adaptée en "cinq méthodes".

L'idée principale de ScatterSearch est de manipuler des **ensembles**, ou **populations** de solutions, plutôt que de traiter les solutions de manière individuelles.

Les cinq méthodes génériques de la ScatterSearch sont :

- Méthode de génération diversifiée : fabrique une population de solution diversifiée à partir d'une solution arbitraire, appellée "seed"
- Méthode d'amélioration : transforme une solution en une ou plusieurs meilleures solutions.
- Méthode de mise à jour de l'ensemble de référence : Construit et met à jour un ensemble des "b" meilleures solutions, avec b généralement petit (e.g. b < 20). Les solutions incluses dans cet ensemble le sont au vu de leur qualité **ou** de leur diversité. Le but étant de pouvoir contenir des solutions de bonnes qualités et des solutions variées pour les combiner par la suite.
- Méthode de génération de sous-ensemble : Extrait un sous-ensemble de l'ensemble de référence afin de créer un base pour créer des solutions combinées
- Méthode de mélange de solution : Combine un sous-ensemble de solutions en une ou plusieurs solutions.

On utilise généralement les notations suivantes :

- P est l'ensemble produit par la première méthode, $PSize$ étant sa taille.
- RefSet est l'ensemble des b "meilleures solutions". Il est généralement de l'ordre de dix fois plus petit que P. On a donc que b est la taille de $RefSet$, ou encore noté $b = |RefSet|$
- On coupe souvent b en deux, en ayant $b1 + b2 = b$. $b1$ correspond aux solutions séléctionnées pour leur qualités, $b2$ pour leur diversité.

Dans la suite, on utilisera les notations suivantes :

- n représente le nombre d'objets de notre instance, x représente une solution. Une solution est un vecteur de 0 et de 1 de taille n. On note x_i la valeur à la i-ème position de x, avec $i \in \{0, 1, \dots, n-1\}$. Si la valeur x_i vaut 0, celà signifie que l'on ne prend pas l'objet i. Inversement si i vaut 1. Par exemple, supposons que nous avons trois objets n_0, n_1, n_2. Une solution pourrait être $x = (0, 1, 0)$, qui peut se lire comme "On prend seulement l'objet n_1".

Implémentation

Voici les idées d'implémentation de chacune des méthodes pour résoudre notre problème de sac à dos multidimensionnel avec contrainte de demande

Les idées données ici sont l'objectif final. Il est probable que nous n'arriverons pas à implémenter tout ça dans les temps, et que nous simplifions les différentes parties.

Méthode de génération diversifiée

Cette méthode sera composée de quatres étapes :

- Génération d'une solution de départ : afin de maximiser la diversité, une solution sera générée aléatoirement, puis "réparée", c'est à dire que pour la rendre faisable, nous enlèverons des objets jusqu'à ce que les contraintes de ressources soient respectées, et on en ajoutera pour satisfaire les contraintes de demandes.
- On va générer le premier tiers de P grace à une méthode fournie par Glover. Soit n le nombre d'objet de notre instance. On considère l'ensemble d'entiers H composé des entiers de 2 à $n-1$, soit $H = \{2, 3, \dots, n-1\}$.

  A partir d'une solution x on génére des solutions x' pour chaque $h \in H$ de la manière suivante : $x'_{1+kh} = 1 - x_{1+kh}$, avec $k = 0, 1, 2, \dots, E(\frac{n}{h})$.

  Tous les indices qui ne sont pas traités par cette règle possèdent la même valeur que la solution originale. On itère jusqu'à ce que le nombre de 1 dans la solution générée est égal à $k = E(\frac{n}{h})$ **ou** que l'ajout d'un un rend la solution infaisable.
- Une fois ce premier tiers de solution généré, on va calculer le score de chaque indice afin d'estimer sa contribution à la qualité des solutions. Soit $\overline{f}_A$ là moyenne de la fonction objective sur l'ensemble A de solution. On a :

  $score(i) = \frac{\overline{f}_{P_i^1}}{\overline{f}_{P_i^1} + \overline{f}_{P_i^0}}$, avec $P_i^1 = \{x \in P | x_i = 1\}$. Idem pour P_i^0.

  Une fois les scores calculés, on génère une ensemble de solution de taile $PSize/3$ de manière constructive. On démarre d'une solution où toutes les valeurs sont à 0. On séléctionne aléatoirement les les x_i de notre solution et on la passe à 1 selon la formule suivante : $Prob(x_i = 1) = min(0, 1 + score(i), 1)$. (En prenant en compte la faisabilité)
- Enfin, on génère le dernier tiers des solutions de départ avec une méthode destructive. On part d'une solution avec des 1 partout, on séléctionne les x_i au hasard et on les passe à 0 avec $Prob(x_i = 0) = 1 - Prob(x_i = 1)$. (En prenant en compte la faisabilité)

Après chaque construction de solution, on recalcule les scores.

Méthode d'amélioration

Pour la méthode d'amélioration, pour chacune des $b/2$ meilleures solutions du $RefSet$, on ordonnes les x_i en fonction de leur score (les plus élevés en premier).
On itère sur tous les x_i en flippant leurs valeurs, et en gardant la modification si elle a augmenté la valeur de la fonction objective. Une fois cela fait, on prend chaque valeur, et on l'échange avec la valeur suivante. Si cela n'améliore pas la fonction objective, on annule et on essaie de l'échanger au augmentant l'indice de 1. On fait ainsi jusqu'à ce que le fonction objective s'améliore ou qu'aucun changement ne soit bénéfique. De plus, on ne garde que les changements qui laissent la solution faisable.

Méthode de mise à jour de l'ensemble de référence

On calcule la fonction objective de chaque solution de P, on choisit les $b/2$ meilleures. On calcule ensuite la distance de hamming d'une solution de avec toutes les autres solutions. On prend les $b/2$ solutions dont la distance de hamming **minimale** est la plus importante. Cela constitue notre $RefSet$.

On supprime également de P les solutions que l'on rajoute dans le $RefSet$.

Méthode de génération de sous-ensemble

Afin de générer des sous ensemble, on va créer toutes les paires de solutions possibles de notre $RefSet$.

Méthode de mélange de solution

A nouveau, on trie les solutions de $RefSet$ selon la valeur objective des solutions. Considérons deux solutions x et y, ainsi que la solution z crée à partir des deux premières.

A partir de là, plusieurs solutions sont possibles, en voici deux exemples :

- On commence avec z à zéro partout. On applique une méthode constructive sur z,. avec la condition qu'une valeur de z ne peut passer à 1 que si la valeur au même indice de x ou de y est à 1.
- Soit f la fonction objective. On calcule pour chaque i les poids suivants : $weight(i) = \frac{f(x)x_i + f(y)y_i}{f(x)+f(y)}$. On passe ensuite les valeurs de z à 1 avec $Prob(z_i = 1) = weight(i)$.

On vérifie bien sûr la faisabilité de la solution à chaque itération. De même, pour chaque nouvelle solution crée, on met à jour le score des i.

Pseudo code

1. GlobalIter = 0, $P = \emptyset$. On utilise la méthode de génération diversifiée pour obtenir $PSize$ éléments dans P
2. On utilise la méthode de mise à jour de l'ensemble de référence pour créer $RefSet$ d'une taille $b \approx PSize/10$. On trie les solutions en fonction de la fonction objective. On initialise la variable *NewSolutions* à **True**.
3. On applique la méthode d'amélioration sur les $b/2$ meilleures solutions de $RefSet$.

   Tant que NewSolutions est True :

   4. On définit $Pool = \emptyset$.
   5. On génère l'ensemble $NewSubstets$ avec la méthode de génération de sous ensemble. On fait $NewSolutions$ = **False**.

      Tant que $NewSubsets$ n'est pas vide :

      6. On séléction la paire suivante de $NewSubset$.
      7. On applique la méthode de mélange de solution pour obtenir une solution x.
      8. Si $x \notin Pool$, alors on ajoute x au $Pool$.
      9. On supprime la paire de $NewSubset$.

   Fin tant que

   10. On applique la méthode d'amélioration sur les $b/2$ meilleures solutions de $Pool$. On remplace les solutions de $Pool$ avec les nouvelles solutions améliorées ainsi créees.
   11. On applique la méthode de mise à jour de l'ensemble de référence sur $RefSet \cup Pool$.

Si $RefSet$ a changé:

12. On fait $NewSolutions$ = **True**

Sinon:

13. On applique la méthode de mise à jour de l'ensemble de référence. On reconstruit $RefSet$ en remplacant les $b/2$ pires solutions avec de nouvelles solutions diverses de P
14. On fait $NewSolutions$ = **True**

Fin si

Si le temps d'éxécution est supérieur au temps imposé maximal:

15. On retourne la meilleure solution de $RefSet$

Fin si

Fin tant que

Quelques sources

- https://www.uv.es/~rmarti/paper/docs/ss10.pdf
- Articles sur le Moodle
- http://leeds-faculty.colorado.edu/glover/Publications/SS%20Chapter%202.pdf