Métaheuristiques - TP

Sujet : Optimisation de la durée de réalisation d'un problème d'ordonnancement

- github du code fourni
- document drivre de suivi d'avancement
- chaine discord des séances (aux horaires prévus sur ADE)

Le travail s'effectue en mônome.

Prendre des notes au fur et à mesure de l'avancement de votre travail :

- qu'avez-vous compris : sur le problème, les représentations de solution, les heuristiques gloutonnes, les méthodes de descente, les voisinages, la métaheuristique développée
- quels résultats avez-vous obtenus avec les différentes méthodes ?

Chronologie des séances :

- **TP1** : mise en place de l'environnement de travail + compréhension du sujet + lien avec l'exercice fait lors de la dernière séance de cours
  - **Code fourni : compréhension et premières implémentations**
    - représentation d'une instance (classe jobshop.Instance);
    - représentation directe d'une solution (classe jobshop.Schedule) : cf représentation section 3.1 de l'exercice fait en cours;
      - implémentation de la méthode toString() de la classe Schedule
    - représentation par numéro de job d'une solution (classe jobshop.encoding.NumJobs) : cf représentation section 3.3 de l'exercice fait en cours)
      - Lire et comprendre l'implémentation
      - Lire et comprendre les heuristiques gloutonnes fournies (appelée basic et random)
  - **Créez la représentation par ordre de passage sur les ressources** (cf représentation section 3.2 de l'exercice fait en cours)
    - implémentation de la classe jobshop.encodings.ResourceOrder
    - implémentation de la méthode toSchedule()

        - testez vos implémentations sur l'exemple fourni
    - **Question** : pourquoi toutes ces représentations d'une solution ?
- **TP2** :
    - ***// Changement de représentation : [NEWS : 31/03/20020] Optionnel (pour le moment)***
        - implémentation d'une méthode pour passer de la représentation directe (jobshop.Schedule) vers la représentation par numéro de job (jobshop.encoding.NumJobs)
        - implémentation d'une méthode pour passer de la représentation directe (jobshop.Schedule) vers la représentation par ordre de passage sur les ressources (jobshop.encodings.ResourceOrder)
        - vérifiez que le changement de représentation ne change pas la solution !
    - **Question** : pourquoi toutes ces représentations d'une solution ?
    - **Heuristiques gloutonnes** :
        - lire et comprendre les heuristiques gloutonnes fournies (appelée basic et random) pour la représentation par numéro de job
        - implémentation d'une heuristique gloutonne pour la représentation par ordre de passage sur les ressources (plusieurs priorités demandées)
        - testez sur les exemples fournis
        - ***[NEWS : 02/04/20020] Optionnel***
            - ajoutez de l'aléatoire dans votre heuristique gloutonne (heuristique randomisée)
            - utilisez votre heuristique randomisée pour faire plusieurs "runs" sur les exemples fournis et conserver la meilleure solution
    - Quelles sont vos conclusions sur vos heuristiques gloutonnes
- -- vacances scolaires --
- **TP3** :
    - **Recherche exhaustive** : sera fournie.
        - testez la recherche heuristique sur les exemples fournis
        - quelles sont vos conclusions sur cette recherche exhaustive ?

- **Voisinage** : implémentation du voisinage de Laarhoven et al. (basée sur la représentation par ordre de passage sur les ressources)
  - * la notion de chemin critique est importante !
  - à compléter .....
- **Méthode de descente** :
  - Implémentation d'une méthode de descente basée sur le voisinage de Laarhoven et sur une seule solution initiale.
  - Lors de l'exploration du voisinage considérer deux variantes : prendre le meilleur voisin sur l'ensemble du voisinage ou prendre le premier voisin améliorant
  - ***[NEWS : 02/04/20020] Optionnel***
    - Descente multi-start : Amélioration de la méthode de descente en utilisant plusieurs solutions initiales avec l'heuristique gloutonne randomisée et le voisinage de Laarhoven
- Quelles sont vos conclusions sur vos méthodes de descente ?

- **TP4 et 5** : à compléter - **Métaheuristique basée sur les principes de la recherche Tabou**
  - **Mise en place du schéma général** de la recherche Tabou (voir support de cours diapo 107 à 119)
    - représentation par ordre de passage sur les ressources
    - même voisinage que pour la méthode de descente
    - conditions d'arrêt : nombre d'itérations
    - ne pas oublier de mémoriser la meilleure solution obtenue au cours des différentes itérations
  - **Spécificité de la méthode** :
    - Définir une ou plusieurs représentations pour mémoriser les "solutions tabou" (les mouvements tabou)
    - Testez différentes valeurs sur la "longueur de la liste tabou "
    - Autorisez l'acceptation d'une solution tabou améliorante (critère d'aspiration)
  - **Visualisez l'évolution de votre méthode de recherche** :
    - cout de la solution courante en fonction du nombre d'itérations

        - .....
    - **Evaluez les différentes variantes** de votre méthode sur les exemples fournis
    - **Quelles sont vos conclusions sur votre méthode (et ses variantes) ?**
    - **Analyse expérimentale** :
        - produire un tableau comparatif des différentes méthodes vues au cours de ces TP sur les exemples fournis

- **Bonus** :
    - (+) ajouter de l'aléatoire dans votre heuristique gloutonne et tester plusieurs "runs" de cette variante "randomisée". Utiliser ensuite cela pour implémenter une méthode de descente multi-start
    - (++) pour votre recherche tabou mettre en place différentes stratégies d'exploration du voisinage (tous les voisins, une sélection de voisins, ...)
    - (+++) : reprendre votre recherche tabou en considérant cette fois la représentation par numéro de job (et donc une nouvelle structure de voisinage, une nouvelle représentation pour la liste Tabou, ...)