TP métaheuristiques
4e IR

MJ. HUGUET
homepages.laas.fr/huguet

Objectifs

L'objectif des séances de TP est de résoudre un problème d'optimisation combinatoire en utilisant des heuristiques et métaheuristiques. Pour cela vous allez mobiliser vos compétences en :

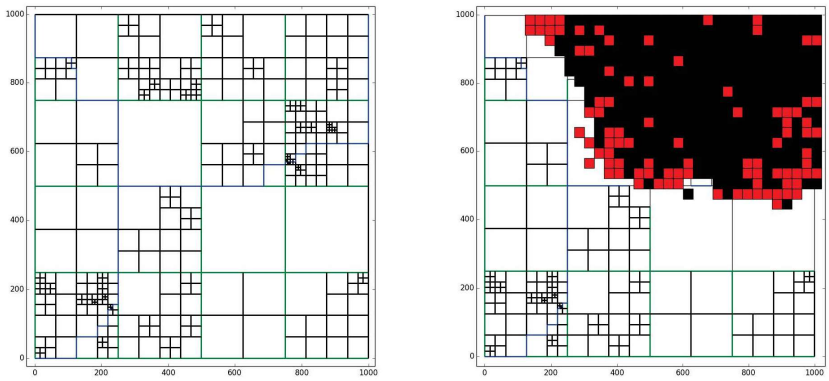
- métaheuristique et optimisation combinatoire
- algorithmique, structure de données, graphes, complexité
- développement
- bon sens

Vous travaillez en binômes. Il y a 5 séances de TP encadrées. L'évaluation s'effectue sur la base d'un rapport expliquant à la fois les méthodes conçues et développées ainsi que les résultats obtenus.

1 Contexte

Nous allons nous intéresser à la planification d'évacuations pour sauver des vies humaines en cas d'incendie. Ce sujet s'inspire d'études menées au LAAS (équipe ROC) dans le cadre d'un projet collaboratif (GeoSafe) se déroulant entre la France et l'Australie : `http://geosafe.lessonsonfire.eu/`.

On supposera que la zone à évacuer et que le modèle de propagation d'incendie sont connus. On dispose alors de données telles que représentées sur les schémas de la figure 1 où le schéma (a) représente les secteurs concernés et les routes existantes et le schéma (b) correspond à la propagation d'un incendie.



(a) road network (b) simulated wildfire

FIGURE 1 – Exemple de zone à évacuer et de propagation d'incendie

A partir de ces données, différents secteurs critiques à évacuer ont été identifiés et les trajets d'évacuation reliant ces secteurs critiques à un secteur sécurisé ont été définis comme illustré sur la figure 2. Ces trajets d'évacuation forment un arbre enraciné au sommet sécurisé.

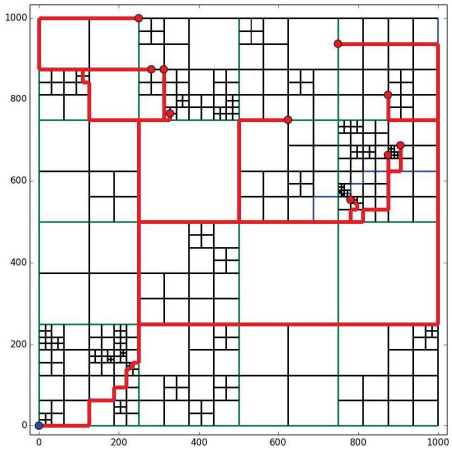


FIGURE 2 – Exemple de trajets d'évacuation

Le problème de planification d'évacuation revient à gérer l'utilisation partagée des différents tronçons des trajets afin de permettre l'évacuation de tous les secteurs critiques dans un délai minimal.

2 Compréhension du problème

2.1 Début de formalisation

On dispose d'un graphe $G(\mathcal{X} = \mathcal{E} \cup \mathcal{S} \cup \mathcal{T}, \mathcal{A})$ tel que :

- $\mathcal{E}$: ensemble de sommets à évacuer
- $\mathcal{S}$: ensemble de sommets sécurisés
- $\mathcal{T}$: ensemble de sommets de transfert

Les arcs $\mathcal{A}$ de ce graphe forment une arborescence allant des sommets à évacuer vers les sommets sécurisés (ainsi à chaque intersection il n'y a qu'une seule possibilité de trajet d'évacuation à suivre). Dans cette arborescence, les sommets de $\mathcal{S}$ sont les racines, les sommets de $\mathcal{E}$ représentent les feuilles et les sommets de $\mathcal{T}$ sont les noeuds internes.

Chaque évacuation vers un sommet sécurisé formant une arborescence, il n'y a qu'un seul trajet d'évacuation pour un secteur à évacuer donné $x_i \in \mathcal{E}$ vers un sommet sécurisé $x_j \in \mathcal{S}$, ce trajet est composé d'un ensemble de tronçons (arcs de l'arborescence) : $A_i = (a_i^1, ..., a_i^{|A_i|})$. Deux sommets à évacuer x_i et x_i' partagent donc des tronçons en commun : $A_i \cap A_i' \neq \emptyset$. Le passage par ces tronçons en commun contraint la planification de l'évacuation car il faut veiller à ne pas les saturer.

L'ensemble des trajets d'évacuation formant une arborescence enraciné au sommet sécurisé, s'il y a plusieurs sommets sécurisés, il y a plusieurs arborescences disjointes. Sans perte de généralité, on supposera pour la suite que le sommet sécurisé est unique.

Sur ce graphe, on dispose des informations suivantes :

- c_{ij}, $\forall x_i, x_j \in \mathcal{X}$: capacité des arcs (x_i, x_j), (en nombre de personnes / unité de temps)
- d_{ij}, $\forall x_i, x_j \in \mathcal{X}$: durée de traversée de chaque arc (x_i, x_j) (en unité de temps)
- f_{ij}, $\forall x_i, x_j \in \mathcal{X}$: date limite d'utilisation de chaque arc (x_i, x_j)
- n_i, $\forall x_i \in \mathcal{E}$: nombre de personnes à évacuer par secteur
- $\overline{\mu_i}$, $\forall x_i \in \mathcal{E}$: taux d'évacuation maximal par secteur, c'est à dire la quantité maximale de personnes pouvant être évacuées par unité de temps

2.2 Hypothèses

Une fois que l'évacuation a débuté pour un secteur $x_i \in \mathcal{E}$, on suppose qu'elle ne peut pas être interrompue (pour éviter des mouvements de panique) et que son taux d'évacuation reste constant.

Il ne peut pas y avoir d'arrêt sur les noeuds internes de l'arborescence ($x_k \in \mathcal{T}$). Ainsi lorsque l'évacuation d'un secteur débute, elle se poursuit sans arrêt jusqu'à atteindre le sommet sécurisé.

Afin de limiter les difficultés, on considèrera (pour le moment) que les dates limites des tronçons ne sont pas à prendre en compte. Quelles sont les conséquences de la prise ou non en compte de ces dates limites (sur le problème à résoudre et pas sur le risque pour les personnes) ?

2.3 Liens avec le RCPSP

Le problème étudié présente des similitudes avec un problème d'ordonnancement de projet sous contraintes de ressources (appelé RCPSP pour Resource Constraint Project Scheduling Problem) pour lequel on considère :

- un ensemble de tâches de durées connues et reliées entre elles par des contraintes temporelles (par exemple des contraintes de précédence)
- un ensemble de ressources de capacité quelconque (lorsque la capacité est supérieure à 1, on parle de ressource cumulative).
- chaque tâche demande une ou plusieurs ressources pour sa réalisation avec une certaine quantité (correspondant à une "consommation" de la capacité des ressources)
- lorsque les ressources redeviennent totalement disponibles après la réalisation des tâches, on dit qu'elles sont renouvelables

Un exemple d'instance d'un problème de RCPSP est fourni sur le schéma 3. Cet exemple comporte 8 tâches et 2 ressources. Les relations de précédence entre tâches sont représentées par les arcs du graphe et les durées par la longueur du rectangle. Sur ce schéma sont également représentées les demandes en resources de chaque tâche (identifiant de la ressource et consommation demandée).

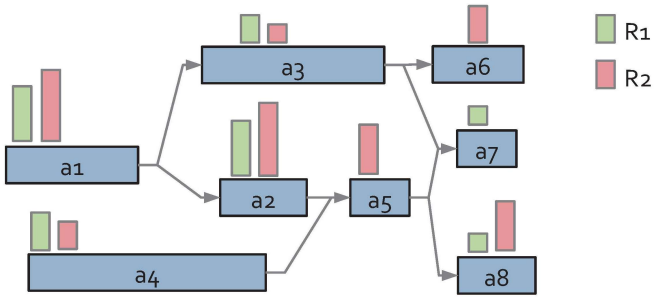


FIGURE 3 – Exemple d'un problème de RCPSP

Les problèmes de RCPSP ont fait l'objet de nombreuses études (théoriques ou appliquées). Plusieurs variantes ont été considérées, parmi lesquelles :

- des fenêtres de réalisation peuvent être associées à chaque tâche (présence de "release date" et/ou de "due date")
- le graphe des contraintes temporelles peut présenter différentes caracéristiques (graphe orienté acyclique, arbre, chaine, ...)
- la durée des tâches peut être réduite en fonction de la quantité de ressources allouées. On parle alors de tâches malléables
- les tâches peuvent être interrompues pendant leur réalisation (la durée totale est connue mais une tâche peut être réalisée en plusieurs fois). On parle alors de problèmes préemptifs.
- ....

Il existe également de nombreuses variantes liées aux caractéristiques des ressources (renouvelables, non renouvelables, indisponibilité, ...) ou aux fonctions objectifs considérées (durée totale, retard, ...).
Si cela vous intéresse, regardez des références bibliographiques sur le RCPSP (voir sur la page web du cours).

2.4 Analyse du problème

Partons d'un exemple Supposons un secteur x_i à évacuer avec 100 personnes et un taux d'évacuation de 20 personnes par unité de temps, alors la tâche d'évacuation du secteur e_i dure 5 unités de temps. Supposons que l'évacuation de x_i passe par 3 tronçons a_1, a_2 et a_3 avec comme durée respective 6, 7 et 12 et que l'on décide de débuter l'évacuation à la date st_i, alors :

- la tâche d'évacuation de x_i occupe le premier tronçon a_1 à partir de la date st_i pendant 5 unités de temps
- la tâche d'évacuation de x_i débute à l'instant $st_i + 6$ sur le deuxième tronçon a_2 (et elle dure 5 unités de temps)
- la tâche d'évacuation de x_i débute à l'instant $st_i + 6 + 7$ sur le troisième tronçon a_3 (et elle dure 5 unités de temps)
- la tâche d'évacuation de x_i commence à arriver au sommet sécurisé à l'instant $st_i + 6 + 7 + 12$ (et se termine au bout de 5 unités de temps).

Résoudre le problème de planification revient alors à déterminer le taux d'évacuation μ_i et les dates de début d'évacuation des différents sommets $x_i \in \mathcal{E}$ pour minimiser la durée totale du plan d'évacuation sachant que :

1. toutes les personnes doivent être évacuées
2. la capacité des tronçons ne doit pas être dépassée

Quel rapport avec le problème étudié ?

- Dans le problème de planification d'évacuation, on peut associer une ressource à chaque tronçon de route, la capacité de la ressource est celle du tronçon.
- L'évacuation d'un secteur peut être représentée par une succession de tâches consommant chacune une ressource différente (ie. les tronçons successifs du plan d'évacuation) et la quantité de chacune des ressources consommées est égale au taux d'évacuation du secteur.
- Pour un secteur donné, lorsque le taux d'évacuation est fixé, on connait la durée d'évacuation du secteur. Toutes les tâches de la séquence d'évacuation d'un secteur ont alors la même durée.
- Ces tâches s'enchainent en respectant l'absence d'attente aux sommets de transfert et en intégrant le temps de traversée des tronçons.

Attention : la durée d'une tâche n'est pas la durée de traversée d'un tronçon !
La prise en compte des dates limites sur chaque tronçon d'évacuation reviendrait à associer des dates de fin au plus tard ("due date") aux différentes tâches utilisant la ressource correspondant à ce tronçon.

2.5 Construction d'un exemple

Considérons un petit exemple avec 3 sommets à évacuer x_1, x_2, x_3 comportant respectivement 48, 30 et 33 personnes. On suppose pour le moment qu'il n'y a pas de date limite associée aux arcs. Les trajets d'évacuation sont : $A_1 = \{a_1, a_4, a_5\}$, $A_2 = \{a_2, a_4, a_5\}$, $A_3 = \{a_3, a_5\}$. Pour chaque arc les valeurs des capacités et des durées de traversée sont :

	a_1	a_2	a_3	a_4	a_5
Capacité	8	5	3	10	11
Traversée	7	4	6	9	12

TABLE 1 – Valeurs des arcs

- Représentez le graphe associé à ce problème.
- Quelle serait la durée minimale d'évacuation si le premier secteur est le seul à évacuer ? Même question pour les autres secteurs.
- En déduire une borne inférieure pour la durée minimale du plan d'évacuation.
- Quelles sont les secteurs d'évacuation potentiellement en conflit (si on fixe le taux d'évacuation de chaque secteur à la valeur maximale permise par chaque premier tronçon) ?
- Quelle serait la durée minimale d'évacuation pour deux secteurs à évacuer en même temps (considérer toutes les paires) en gardant le taux maximal d'évacuation maximum de chaque secteur ? N'hésitez pas à faire un diagramme de Gantt pour visualiser les solutions.
- Si on considère que le taux maximal d'évacuation pour x_1 est de 7 et que le taux maximal d'évacuation de x_2 est de 3, quelle est la durée minimale pour l'évacuation simultanée de x_1 et de x_2 ?
- Pouvez-vous proposer une solution pour évacuer les 3 secteurs simultanément ?
- Pour concevoir une heuristique à ce problème d'évacuation, quels seraient les éléments importants à considérer ?
- Avez-vous fait le lien avec le problème de RCPSP ?

Considérons maintenant des dates limites pour la traversée des arcs afin d'intégrer la propagation d'un feu débutant à proximité du sommet x_1.

	a_1	a_2	a_3	a_4	a_5
Dates limites	13	26	28	33	46

TABLE 2 – Propagation du feu

Remarque : Les valeurs des dates limites sur les arcs peuvent être combinées (par exemple, la date limite de 26 sur l'arc a_2 n'est jamais atteignable compte tenu du fait que la date limite de l'arc a_4 est de 34 est qu'il a une durée de traversée de 9.

- La solution obtenue sans prise en compte des dates limites reste-t-elle réalisable ?
- Considérez séparément l'évacuation de chaque sommet en prenant en compte les dates limites.
- Pouvez-vous en tirer des conclusions sur les taux d'évacuation ?
- Pouvez-vous en tirer des conclusions sur les évacuations en conflit ?

3 Déroulement des TP

Ces TP constituent la seule évaluation pour l'enseignement de métaheuristiques (pas d'évaluation écrite). Ils nécessitent un minimum de travail personnel. Vous êtes en binôme, partagez-vous efficacement le travail et prenez des notes au fur et à mesure des séances.
Commencez par lire le sujet dans son intégralité.

3.1 Première étape (1 séance)

3.1.1 Lecture des jeux de données

Téléchargez les jeux de données fournis et regardez dans un de ces fichiers comment les lire. Lors de la lecture, considérez que la capacité d'un arc est un entier.

Travail à faire Ecrivez un programme permettant de lire les jeux de données fournis.

3.1.2 Vérification et évaluation d'une solution

Soit le format de solution suivant :

- <nom de l'instance résolue>
- <nombre de sommets à évacuer>
- pour chaque sommet à évacuer : <son identifiant>, <son taux d'évacuation>, <sa date de début d'évacuation>
- nature de la solution : <valid ou invalid>
- <valeur de la fonction objectif>
- <temps de calcul>
- <méthode> : le nom de la méthode utilisée et la version de l'implémentation
- champ libre (paramètre de la méthode, nom du binôme, ...)

Par exemple :
dense_10_30_3_8
10
233 1354 0
427 1941 0
370 3690 0
70 1763 0
493 3381 0
346 3726 0
489 3012 0
174 838 0
236 116 0
273 841 0
invalid
inf
10.5
handmade 0.1.0
"everyone evacuates from start ; no constraint check"

Remarque : Il est important de garder trace de comment la solution présentée dans le fichier a été générée. Cela peut vous servir à vérifier que vous avancez dans la bonne direction, à vous rendre compte à quel point vous avez amélioré votre méthode, et à donner des métriques de votre avancée dans votre rapport de TP. Le format est assez libre sur ce que vous pouvez mettre dans ce champ, mais voici les informations recommandées :
Méthode : donnez le nom de la méthode (implémentation spécifique) que vous avez utilisée. Exemples : taboo_de_marie_et_pierre, scipy.optimize.anneal, ...

Travail à faire

- Créez un fichier représentant les données de l'exemple de la section 2.5 et un fichier représentant une solution pour cette instance (borne inférieure et/ou solution réalisable).
- Ecrivez une méthode permettant de vérifier si la solution fournie est réalisable et de vérifier si la valeur de la fonction objectif est correcte.
- Appliquer votre méthode de vérification sur l'exemple.
- D'autres pistes de vérification de solution vous seront apportées en séance (accès à un vérificateur existant utilisable en complément de celui que vous aurez développé).

En prévision du rapport, écrivez **l'algorithme** de cette méthode. Vous pouvez aussi illustrer à partir d'un exemple. Pouvez-vous donner la complexité de votre algorithme ? Quel est l'intérêt de cette méthode de vérification pour le problème d'optimisation étudié ?

Si vous avez terminé avant la fin de la séance, passez à l'étape suivante.

3.2 Deuxième étape (1 séance)

3.2.1 Calcul d'une borne inférieure

Le problème à résoudre est un problème de minimisation. Qu'est-ce qu'une borne inférieure ? A quoi ça sert ? Comment en obtenir ? (voir le cours).

Travail à faire

- Proposez une méthode pour calculer une borne inférieure de la fonction objectif.
- Appliquez votre méthode sur l'exemple puis sur les instances fournies (les solutions doivent être écrites dans le format défini).
- Vérifiez à chaque fois si la solution produite est correcte (réalisable et valeur annoncée de l'objectif correcte).
- Pensez que vous pouvez écrire un script pour lancer tous ces calculs.

En prévision du rapport, écrivez **l'algorithme** permettant ce calcul. Vous pouvez aussi illustrer à partir d'un exemple. Pouvez-vous donner sa complexité ? Les résultats obtenus sont-ils cohérents avec ce qui était attendu ?

3.2.2 Calcul d'une borne supérieure

Le problème à résoudre est un problème de minimisation. Qu'est-ce qu'une borne supérieure ? A quoi ça sert ? Comment en obtenir ? (voir le cours).

Travail à faire

- Proposez une heuristique pour calculer une borne supérieure de la fonction objectif. Toute heuristique même donnant une solution de mauvaise qualité est acceptable, la seule condition à respecter est que la solution générée doit être réalisable et que le résultat obtenu soit écrit dans un fichier respectant le format défini.
- Appliquez votre heuristique sur l'exemple.
- Appliquez votre heuristique sur les instances fournies. Vérifiez à chaque fois si la solution produite est correcte (réalisable et valeur annoncée de l'objectif correcte). Aviez-vous pensé à écrire un script pour lancer ces tests ?
- Transmettez les fichier solutions produits (avec si besoin les instances correspondant) à un autre binôme (ou plusieurs). Une fois la validation de vos solutions effectuée, notez le nom du groupe ayant validé cette solution.
- Vous devrez également vérifier des solutions pour les autres binômes.

En prévision du rapport, écrivez **l'algorithme** de cette heuristique. Vous pouvez aussi illustrer à partir d'un exemple. Pouvez-vous donner sa complexité ?
Vous pouvez commencer à réfléchir à la présentation des résultats obtenus par votre heuristique sur les différentes instances. Un tableau de résultat sera indispensable mais il peut être complété par des graphiques.

Si vous avez terminé avant la fin de la séance, passez à l'étape suivante.

3.3 Troisième étape (1,5 séance) : Recherche locale - Intensification

On souhaite améliorer la solution calculée par une heuristique (comme celle développée à l'étape précédente). Pour cela, vous allez mettre en place une méthode de recherche locale (troisième et quatrième étapes des TP).
Le but de cette étape consiste à concevoir une méthode de recherche locale exploitant uniquement un processus d'intensification (méthode de descente). Retrouver l'algorithme général d'une méthode de descente dans le cours.

Pour ceux qui le souhaitent : il est possible de prendre en compte les dates limites dès cette étape. Regardez la section 3.5 à ce sujet.

Travail à faire

- Définissez un ou plusieurs voisinage(s) permettant d'explorer l'espace de recherche. Pour ces voisinages, définissez également une ou plusieurs fonctions d'évaluation.
- Réflechissez à la stratégie d'exploration des voisinages ainsi qu'aux conditions d'arrêt de la méthode de descente.
- Pensez à collecter des informations sur le déroulement de la méthode (nombre de voisins explorés, nombre d'itérations, ...)
- Implémentez la méthode de descente basée sur le(s) voisinages défini(s). Comment vous assurer que votre méthode est bien une méthode de descente ?
- Appliquez votre méthode sur l'exemple et les instances fournies. Vérifiez à chaque fois si la solution produite est correcte (réalisable et valeur annoncée de l'objectif correcte). Si vous avez écrit un script pour les tests, il va encore vous servir !
- Comparez les résultats obtenus à ceux obtenus par le calcul de borne inférieure et l'heuristique initiale (borne supérieure). Là aussi, pensez à écrire un script pour effectuer cette comparaison (et pourquoi pas générer directement des tableaux ou des graphiques)

En prévision du rapport, décrivez les voisinages retenus et les fonctions d'évaluation associées. Vous pouvez illustrer à partir d'exemples. Pouvez-vous donner les tailles des voisinages, les complexités des fonctions d'évaluation ?
Par rapport à l'algorithme général d'une méthode de descente, précisez les spécificités de votre méthode (conditions d'arrêt, exploration des voisinages, utilisation ou non de plusieurs fonctions d'évaluation, ...).
Vous pouvez continuer réfléchir à la présentation des résultats obtenus par votre méthode de descente sur les différentes instances. Un tableau de résultat sera indispensable mais il peut être complété par des graphiques.
Vous pouvez également réfléchir à la présentation des résultats comparatifs entre l'heuristique et la méthode de descente.

3.4 Quatrième étape (1,5 séance) : Recherche locale - Diversification

La méthode de descente peut rester bloquer dans un optimum local. Lors de cette étape, vous allez mettre en place un processus de diversification afin de finaliser votre méthode de recherche locale.

Travail à faire

- Choisissez un processus de diversification (par exemple : multi-start, voisinages variables, recuit, tabou, ...).
- Pensez à collecter des informations sur le déroulement de la méthode (nombre de voisins explorés, nombre d'itérations, spécificité en fonction du processus de diversification, ...)
- Implémentez la méthode de recherche locale incluant les deux processus d'intensification et de diversification). Comment vous assurer que votre méthode comporte bien un processus de diversification ?
- Appliquez votre méthode sur l'exemple et les instances fournies. Vérifiez à chaque fois si la solution produite est correcte (réalisable et valeur annoncée de l'objectif correcte). C'est encore le même script pour les tests.
- Comparez les résultats obtenus à ceux obtenus par le calcul de borne inférieure, l'heuristique initiale et la méthode de descente. Il reste à compléter le script permettant de comparer des méthodes.

En prévision du rapport, décrivez le processus de diversification retenu et expliciter les différents paramètres de la méthode développée.
Vous pouvez continuer à réfléchir à la présentation des résultats obtenus par votre méthode avec diversification sur les différentes instances. Un tableau de résultat sera indispensable mais il peut être complété par des graphiques.
Vous pouvez également continuer à réfléchir à la présentation des résultats comparatifs entre les différentes méthodes développées lors de ces TP.

3.5 Pour aller plus loin

Intégration de nouvelles contraintes Les dates limites associées aux arcs des chemins d'évacuation ont été ignorées. Il est possible de les intégrer dès l'étape de conception de la méthode de descente. Une piste pour cela peut être de modifier la fonction objectif pour y ajouter une information sur le non respect des dates limites. L'objectif se ramène alors à la minimisation de la somme des contraintes non satisfaites et de la durée totale du plan d'évacuation. Des pondérations sur les différentes parties de l'objectif sont à réfléchir.
Le non respect de dates limites peut également être exploité lors des explorations de voisinages.

Autres jeux de données Pour ceux qui le souhaitent, vous pouvez récupérer le générateur d'instances écrit par Emmanuel Hébrard : `https://github.com/ehebrard/evacsim` (attention, le fichier readme n'est pas forcément à jour).
Le code fourni utilise python 2.7 et nécessite différents packages dont networkx. Pour l'installer, tapez la commande : `python2.7 -m pip install networkx -user` ou téléchargez le code source : `https://pypi.org/project/networkx/2.2/#files` puis décompressez l'archive récupérée et lancer la commande (depuis le dossier décompressé).
La génération de jeux de données dans le format qui nous intéresse est `python2.7 gen_dataset.py`. Les autres commandes servent à générer d'autres types de jeux de données.

4 Evaluation

Le travail réalisé devra être rendu via un dépot git (code, jeux de données, solutions produites et rapport). La note est attribuée pour moitié au travail réalisé et pour moitié au rapport. Le détail du barême est exposé ci-après (il est succeptible de connaitre de légères variations dont vous serez informés au plus tard en fin des séances de TP).

4.1 Travail réalisé sur 20

- 8 points : réalisation des étapes 1 et 2 (format des solutions, vérification de solutions, calcul de bornes inférieures, calcul de bornes supérieures). Présence d'un readme pour utiliser ces fonctionnalités. Les résultats obtenus doivent être corrects et l'implémentation efficace.
- 7 points : réalisation de l'étape 3 (méthode de recherche locale intégrant uniquement un processus d'intensification). Présence d'un readme pour utiliser ces fonctionnalités. Les résultats obtenus doivent être corrects et l'implémentation efficace.
- 5 points : réalisation de l'étape 4 (méthode de recherche locale intégrant un processus d'intensification et de diversifiation). Présence d'un readme pour utiliser ces fonctionnalités. Les résultats obtenus doivent être corrects et l'implémentation efficace.

4.2 Rapport sur 20

- 4 points : respect des consignes (qui seront fournies) : date de rendu, entête du mail, dépot git, format de fichier, structure du rapport (introduction, conclusion, plan, références, ...), grammaire et orthographe.
- 6 points : présentation du travail demandé dans les étapes 1 et 2 (vérification de solutions, calculs de bornes inférieures et supérieures) : explication des algorithmes implémentés, analyse de leur complexité, analyse critique des résultats obtenus, analyse des performances de l'heuristique (en termes de valeur de la fonction objectif et de temps de calcul. Fournir le tableau des résultats (éventuellement en annexe).
- 6 points : présentation du travail demandé dans l'étape 3 (intensification : méthode de descente) : explication de l'algorithme de la méthode implémentée (spécificité de votre méthode) ; explication de(s) voisinage(s) exploré(s) ; explication de la méthode d'évaluation des voisinages. Analyse de complexité des différents algorithmes implémentés et analyse des performances (valeur fonction objectif et temps de calcul) de la méthode avec intensification. Fournir le tableau des résultats (éventuellement en annexe). Comparaison des résultats de la méthode de descente et de l'heuristique.
- 4 points : présentation du travail demandé dans l'étape 4 (diversification) : explication du processus de diversification implémenté. Analyse des performances (valeur fonction objectif et temps de calcul) de la méthode avec intensification et diversification. Fournir le tableau des résultats. Comparaison des résultats de cette méthode avec la méthode de descente et de l'heuristique.

Pour la rédaction du rapport, nous vous conseillons d'utiliser LaTeX. Vous disposez d'éditeurs LaTeXen local sur les postes de travail des salles de TP (ou vous pouvez en installer un sur votre ordinateur personnel) et en utilisant les fonctionnalités de git vous pouvez travailler ensemble sur un même document. Des éditeurs en ligne et collaboratifs sont également accessibles à tous (par exemple Overleaf ou shareLateX qui ont fusionné).