Université Polytechnique Hauts-de-France
Institut National des Sciences Appliquées Hauts-de-France **2019-2020**

Master Cyberdéfense et Sécurité de l'Information — Semestre 10
Complexité - Heuristiques & Métaheuristiques — Christophe Wilbaut

Mini-Projet

Introduction :

Ce document vise à présenter synthétiquement le problème que vous allez traiter dans le cadre du travail de groupe du module *Complexité - Heuristiques - Métaheuristiques*, avec les modalités de travail et d'évaluation. Lisez attentivement ce document, et n'hésitez pas à poser des questions si certains points ne vous semblent pas clairs.

Organisation :

Rappel des règles présentées en séance de cours et à respecter :

- Vous devez former des équipes de 3 (**minimum**) à 5 (**maximum**) personnes. La composition des équipes est libre, mais doit être **définitive au plus tard** au début de la seconde séance dédiée au travail de groupe.
- Vous pouvez utiliser toutes les séances de TD et de TP du module pour travailler **en groupe** sur le problème. Ces séances sont également l'occasion de discuter avec l'intervenant présent sur les éventuels points de blocage ou sur vos pistes de réflexion.
- Vous avez la possibilité d'échanger entre vous les résultats obtenus sur les instances tests du problème.
- Le langage de programmation est libre.

Description du problème :

Le problème traité est le problème du sac-à-dos multidimensionnel **avec contraintes de demande**, qui se base sur le problème du sac-à-dos utilisé plusieurs fois en cours via des exemples. A partir d'un ensemble $N = \{1, 2, \ldots, n\}$ de n objets, caractérisés par un poids (a_j) et un profit (c_j), on cherche le sous-ensemble d'objets pouvant être sélectionné pour être mis dans un sac d'une certaine capacité (b) en maximisant la somme des profits. Rappelons la formulation mathématiques du problème du sac-à-dos (SAD) en variables 0-1 :

$$\text{(SAD)} \left[\begin{array}{ll} \max & \sum_{j=1}^{n} c_j x_j \\ \text{s.c. :} & \sum_{j=1}^{n} a_j x_j \leq b \\ & x_j \in \{0,1\} \qquad j \in N = \{1, \ldots, n\} \end{array} \right.$$

Dans la variante multidimensionnelle (SADM) on considère un ensemble $M = \{1, 2, \ldots, m\}$ de contraintes de ressources (avec $m > 1$). Ainsi chaque objet est caractérisé par m poids (a_{ij}), un pour chaque contrainte. On obtient la formulation suivante :

$$\text{(SADM)} \left[\begin{array}{ll} \max & \sum_{j=1}^{n} c_j x_j \\ \text{s.c. :} & \sum_{j=1}^{n} a_{ij} x_j \leq b_i, \quad i \in M = \{1, \ldots, m\} \\ & x_j \in \{0,1\} \qquad j \in N = \{1, \ldots, n\} \end{array} \right.$$

Toutes les coefficients (a_{ij}, c_j, b_i) sont supposés strictement positifs.

Le SADM avec contraintes de demande (SADMD) introduit un ensemble de q (≥ 1) contraintes supplémentaires, dites de demande. L'ensemble des contraintes est alors défini par $M = \{1, 2, \ldots, m, m+1, \ldots, m+q\}$. Chaque objet est toujours caractérisé par un profit c_j et cette fois par $m+q$ poids (a_{ij}), un pour chaque contrainte. On obtient la formulation suivante :

$$\text{(SADMD)} \left[\begin{array}{lll} \max & \sum_{j=1}^{n} c_j x_j & \\ \text{s.c. :} & \sum_{j=1}^{n} a_{ij} x_j \leq b_i, & i \in \{1, \ldots, m\} \\ & \sum_{j=1}^{n} a_{ij} x_j \geq b_i, & i \in \{m+1, \ldots, m+q\} \\ & x_j \in \{0, 1\} & j \in N = \{1, \ldots, n\} \end{array} \right.$$

Le SADMD, comme le SADM, est un problème NP-Difficile pour lequel l'utilisation d'une méthode exacte n'est pas recommandé dès que la taille de l'instance traitée (nombre d'objets et nombre de contraintes) augmente.

Attention : les coefficients du SADMD peuvent être positifs, négatifs ou nuls.

Votre travail consiste à implémenter une ou plusieurs méthodes approchées (heuristiques, métaheuristiques) de manière à obtenir les meilleures solutions **faisables** possibles, dans un temps imparti. Votre programme **devra s'arrêter à la fin de ce temps** (voir section Modalités).

Vous trouverez sur l'espace de cours sur `Moodle` quelques articles scientifiques dont vous avez le droit de vous inspirer, ainsi qu'un ensemble de jeux de données que vous **devez** utiliser pour tester vos approches.

Format des données (en entrée et en sortie) :

Un fichier texte est associé **à chaque instance** à traiter. Tous les fichiers respectent le même format, excepté pour les espaces, tabulations ou lignes blanches **qui peuvent différer** d'un fichier à l'autre. Le format est le suivant :

- 1ère ligne : le nombre d'objets (n), le nombre de contraintes de ressources (m) et le nombre de contraintes de demande (q) (au moins un espace entre deux valeurs).
- Ensuite on trouve, à la suite, les n profits c_j, pour j de 1 à n.
- Les capacités des m contraintes, dans l'ordre (de 1 à m).
- La demande des q contraintes de demande, dans l'ordre (de $m+1$ à $m+q$).
- Puis, m blocs, 1 pour chaque contrainte de capacité, avec à chaque fois n valeurs pour les poids a_{ij}.
- Enfin, q blocs, 1 pour chaque contrainte de demande, avec à chaque fois n valeurs pour les poids a_{ij}.

Vos algorithmes devront pouvoir gérer des instances avec $n \in \{100; 250; 500\}$, $m \in \{5; 10; 30\}$ et $q \in \{1; \lfloor m/2 \rfloor ; m\}$. Ainsi les plus petites instances ont $n = 100$, $m = 5$, $q = 1$, et les plus grandes $n = 500$, $m = 30$, $q = 30$.

Votre programme doit fournir en sortie un fichier texte respectant **exactement** le format suivant :

- 1ère ligne : nom du fichier instance testé (exemple : *100Md10_1_5_pos.txt*).
- 2ème ligne : n, m et q (séparés par **un** espace).
- 3ème ligne : la valeur de la solution obtenue ($\sum_{j=1}^{n} c_j x_j$).
- 3ème ligne : la valeur de chaque variable dans la solution obtenue (c'est-à-dire une suite de n valeurs (0 ou 1), séparées par **un** espace, dans l'**ordre initial des variables**).

Le nom du fichier de sortie doit être celui de l'instance en entrée auquel vous ajoutez le terme "_res", soit sur l'exemple : *100Md10_1_5_pos_res.txt*.

Exécution :

Votre programme doit pouvoir être lancé en ligne de commandes, de manière à pouvoir lui passer des arguments. La ligne de commande doit être de la forme `TPMETA nom_instance cpu`, si votre exécutable s'appelle `TPMETA`, `nom_instance` correspond au fichier .txt de l'instance (qui se trouvera dans le même répertoire que l'exécutable), `cpu` étant le temps maximum (en secondes) alloué pour la résolution. Cela pourrait donc donner par exemple : `TPMETA 100Md10_1_5_pos.txt 30`

Attention : cela veut dire que votre programme doit s'arrêter au bout des `cpu` secondes, sans intervention extérieure (pas d'arrêt forcé en surveillant le chronomètre).

La machine utilisée pour les tests tourne sous Windows 10. Vous devrez fournir en priorité l'ensemble de vos fichiers **sources**, et éventuellement un exécutable. Le temps d'exécution autorisé pour établir le classement des équipes (voir point suivant) sera de 30 secondes par instance. Il pourra être ramené à 15 secondes ou étendu à 60 secondes si nécessaire pour départager des équipes.

Modalités d'évaluation :

A la fin des séances de TD et TP, vous devrez déposer sur `Moodle` un ensemble de documents pour l'évaluation. Cette évaluation sera composée de plusieurs notes, soit individuelles soit collectives. Vous devrez en particulier fournir :

- Un rapport. Il sera constitué d'une partie commune à l'équipe, ainsi que d'une partie individuelle pour chaque membre de l'équipe. **Un seul rapport sera déposé**, les parties individuelles devant être mises à la suite les unes des autres. L'objectif doit être ici d'une part de décrire au mieux les méthodes mises en œuvre, mais aussi de mettre en évidence les contributions de chacun.
- L'ensemble des fichiers **sources** permettant de générer l'exécutable de votre programme, avec à minima un `ReadMe` expliquant comment générer cet exécutable (en précisant notamment les options de compilation à utiliser si nécessaire). Vous déposerez également une version de votre exécutable, uniquement si elle est compatible avec Windows.

ATTENTION

Le rapport, les fichiers sources et l'exécutable devront être fournis au plus tard le vendredi soir (23h50) **suivant la dernière séance** de TD ou TP. Aucun dépôt ne sera possible après, et aucun envoi par mail ne sera accepté. En cas de non dépôt la note sera 0. Le dépôt se fera sous la forme d'une archive (uniquement au format .zip, .rar, .tar ou .7z) dont le nom sera formé de la suite des noms des membres de l'équipe.

L'évaluation portera également sur une soutenance **individuelle**. Chaque membre de l'équipe aura 5 à 10 minutes pour présenter le travail du groupe et surtout ses contributions à l'aide de slides au format .ppt(x) ou .pdf. Chacun devra déposer sur `Moodle` le fichier de ses slides **au plus tard le soir de la soutenance** (23h50) (pas encore planifiée au moment de l'écriture du sujet). En cas de non dépôt à temps la note ne portera que sur l'exposé oral et les réponses aux questions.

Finalement, un classement sera établi sur la base des résultats moyens obtenus par les algorithmes des différentes équipes sur l'ensemble des instances fournies.
La note finale de chacun sera donc calculée sur la base de :

1. une note individuelle pour le rapport (note ramenée sur 5 points),
2. une note individuelle pour la présentation et la qualité des réponses aux questions (note ramenée sur 5 points),
3. une note sur la qualité du code et la maîtrise technique (code et méthodes mises en œuvres), note collective ramenée sur 5 points),
4. une note sur la performance de vos algorithmes (note collective ramenée sur 5 points).

Le respect de l'ensemble des consignes sera également pris en compte.