Projet RO : Application de Recherche Opérationnelle

Introduction

Dans le cadre de ce projet de Travaux Pratiques (TP) en Recherche Opérationnelle, ce projet vous propose de réaliser une application informatique utilisant des techniques de programmation linéaire (PL), programmation linéaire en nombre entier (PLNE) et programmation mixte (PLM) pour résoudre des problèmes d'optimisation.Vous travaillerez en groupe de 5 étudiants et utiliserez le langage de programmation Python.

Objectifs

Les objectifs de ce projet sont les suivants:

- Développer une interface graphique intuitive et conviviale (pour saisir les données du problème, visualiser les résultats et interagir avec l'application.
- Modéliser des problèmes d'optimisation (PL / PLNE / PLM) en utilisant le langage de programmation **Python**.
- Résoudre les problèmes modélisés en utilisant le solveur **Gurobi.**
- Analyser les résultats obtenus et les interpréter dans le contexte des problèmes étudiés.

Déroulement du projet

1. Attribution du sujet de projet

Votre sujet est déterminé par une simple combinaison de deux nombres : Le numéro de problème RO **(n)** et le numéro de l'application **(p)** à partir du tableau de répartition des groupes.

- Déterminer le problème d'optimisation (à partir de n) : Par exemple : Problème de Transport, Problème d'Affectation, Problème de Mélange, …)
- Déterminer l'application spécifique (à partir de p) : Applications 1, 2, 3, 4 ou 5

Le sujet final est déterminé par lla combinaison du problème (n) et de l'application (p) à partir du tableau des sujets.

2. Modélisation

La modélisation de chaque problème devra inclure:

- La définition des variables de décision
- La formulation de la fonction objectif
- L'expression des contraintes

3. Développement de l'IHM

L'Interface Homme-Machine (IHM) sera développée en Python, en s'appuyant sur le framework robuste **PyQt/PySide** afin d'assurer une gestion événementielle performante et une interface utilisateur professionnelle.

L'IHM devra permettre de réaliser les fonctions critiques suivantes :

- ***Saisie structurée des données :*** Permettre l'entrée et la modification des paramètres du modèle (coûts, capacités, contraintes) de manière organisée, idéalement via des composants comme le `QTableWidget` pour les données tabulaires (matrices).

- ***Contrôle du calculn non bloquant :*** Fournir des contrôles clairs (boutons de lancement) pour exécuter le solveur Gurobi. L'implémentation devra utiliser le multithreading (`QThread`) pour garantir que l'interface utilisateur reste réactive et ne gèle pas pendant l'exécution des calculs d'optimisation (particulièrement pour les problèmes PLNE de grande taille).

- ***Visualisation et interprétation des solutions :*** Afficher de manière synthétique et détaillée les résultats obtenus :
  - La *valeur optimale* de la fonction objectif.
  - Le détail des *variables de décision* optimales.
  - Une visualisation graphique pertinente (par exemple, utilisation de Matplotlib ou d'une bibliothèque de graphes intégrée) pour représenter la solution, comme la tournée optimale (pour le TSP/VRP) ou l'affectation des ressources.

4. Résolution

Les problèmes d'optimisation modélisés seront résolus en utilisant le solveur Gurobi.

5. Tests et validation:

- Testez rigoureusement l'application avec différents jeux de données pour garantir son bon fonctionnement.
- Validez les résultats obtenus par rapport à des solutions analytiques ou à d'autres méthodes de résolution.

6. Analyse et interprétation des résultats

Pour chaque problème, les résultats obtenus seront analysés et interprétés dans le contexte de l'application étudiée.

7. Rapport

Chaque groupe d'étudiants rédigera un rapport documentant son travail, incluant:

- Une présentation des membres du groupe (Prénom et nom, photo d'identité pour chaque étudiant)
- Une description des applications de problèmes d'optimisation traitées
- La modélisation mathématique des problèmes
- La description de l'IHM développée
- Les résultats obtenus et leurs analyses

Livrables:

- Code source complet de l'application (.py, .ui)
- Documentation du projet (PDF)

Evaluation

L'évaluation du projet sera basée sur les critères suivants:

- La qualité de la modélisation des problèmes
- L'ergonomie et la fonctionnalité de l'IHM
- L'efficacité du code Python
- La précision et l'interprétation des résultats
- La qualité du rapport et de la présentation

Il est impératif de noter que la qualité et l'évaluation finale de votre travail dépend directement du degré de complexité de la modélisation et de la richesse des données traitées dans votre application.

1. Complexité de la modélisation et des paramètres

L'évaluation prendra en compte la rigueur et l'exhaustivité du modèle :

Nombre de paramètres : Un modèle intégrant un nombre important de données d'entrée (capacités, coûts unitaires, distances, temps, ressources, etc.) démontre une meilleure appropriation du problème réel.

Intégration de contraintes : La capacité à traduire les subtilités du cas d'application en contraintes mathématiques complexes (e.g., contraintes de temps, contraintes logiques binaires, contraintes de dépendance) sera un facteur de pondération majeur. *Exemple :* Pour le VRP, l'intégration des fenêtres de temps ou des capacités multiples (poids *et* volume) augmentera la valeur du travail.

2. Critères et attributs traités

Le niveau d'ambition et la fidélité au contexte métier seront valorisés :

Critères multiples : Les applications qui tentent d'optimiser plusieurs critères (par exemple, minimiser les coûts *et* maximiser la qualité, ou minimiser le temps *et* minimiser le nombre de ressources) seront mieux notées.

Attributs spécifiques : L'ajout de règles métier spécifiques et réalistes (e.g., jours de repos obligatoires, incompatibilités de produits, flexibilité des machines) qui rendent la modélisation moins triviale est fortement encouragé.

3. Impact sur l'évaluation

l'évaluation du projet sera directement corrélée à la manière dont vous avez enrichi et complexifié le problème initial : Une modélisation simple avec un faible nombre de contraintes et de paramètres correspondra à la note de base (moyenne), tandis qu'un modèle riche en variables entières/binaires (PLNE ou PLM), intégrant des attributs complexes et un grand nombre de paramètres, permettra d'atteindre les meilleures notes. Il est donc recommandé de ne pas se limiter à la formulation la plus simple du problème, mais de l'adapter en ajoutant au moins deux à trois niveaux de complexité réalistes tirés de votre application spécifique.

Logiciels et outils

- Langage de programmation : Python
- Framework : PyQt / PySide
- Solveur : Gurobi

Contraintes

- Le projet doit être réalisé par des groupes de 5 étudiants.
- La date limite de remise du projet est le **[12 Décembre 2025]**.