# Projet IT45

Le projet IT45 vise à confronter les étudiants en binômes à un problème de recherche opérationnelle concret afin qu'ils puissent appliquer et adapter les techniques qui auront été vues en cours. Il sera évalué suivant 3 critères :
— Le travail réalisé par l'équipe : performances obtenues sur les instances, analyse des résultats expérimentaux, clarté des codes sources,
— Un rapport qui explique la problématique traitée, les méthode développées pour solutionner le problème considéré, ainsi que les résultats expérimentaux obtenus et leur analyse
— Une soutenance d'une dizaine de minutes au cours de laquelle vous exposerez votre travail.
L'ensemble de ces 3 éléments : code source, rapport et soutenance devront être déposés sur moodle avant la soutenance sous la forme d'une archive zip dont le nom sera : 'projet-it45-p25-nom1-prenom1-nom2-prenom2.zip'

## Problème considéré

Le projet portera dans un premier temps sur le problème du voyageur de commerce (TSP) : il s'agit de trouver le plus court chemin qui passe une et une seule fois par chacune des n villes et revient à la ville de départ. L'objectif de ce projet est de traiter le problème du voyageur de commerce (TSP) en utilisant différentes techniques de résolution : exactes et approchées. Les performances obtenues en termes de qualité de la solution et de temps d'exécution seront analysées et comparées sur différentes instances du TSP.
Pour la phase de test, vous pourrez utiliser les 10 premières villes de l'instance 'berlin52.tsp' qui est une instance du problème du voyageur de commerce à 52 villes.
Dans un second temps, le projet sera étendu au Vehicle Routing Problem (VRP). Cette extension consistera à adapter les méthodes développées pour gérer plusieurs véhicules partant d'un dépôt central et desservant chacun un sous-ensemble de clients, avec une contrainte de capacité.

## Démarche et attentes

L'objectif du projet est de traiter le TSP suivant 3 approches :

- une méthode exacte : algorithme de Little
- la métaheuristique "recherche tabou" RT
- la métaheuristique "algorithme génétique" (AG)

L'objectif du projet est de proposer des adaptations de ces algorithmes pour que soit la qualité de la solution soit améliorée (meilleure solution trouvée en terme de distance parcourue), soit le temps d'exécution est plus court à qualité de solution égale.
Les algorithmes seront ensuite appliqués sur différentes instances du TSP, notamment celles qui se trouvent dans le répertoire 'data' : a280.tsp, berlin52.tsp, eil76.tsp, kroA100.tsp. Mais des tests pourront également être menés sur d'autres instances de TSPLIB :

- Symmetric traveling salesman problem (TSP) :
  http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/tsp
- Asymmetric traveling salesman problem (ATSP) :
  http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/atsp

Les résultats obtenus sur ces instances devront ensuite être analysés en termes de qualité de la solution obtenue au bout d'un laps de temps donné (par exemple : 1 minute, 10 minutes ou 1 heure). Dans le cas de l'algorithme de Little, si la solution optimale n'a pas été retenue au bout du temps d'exécution fixé, on retiendra la meilleure solution trouvée. Pour les instances de TSPLIB, la solution optimale est

généralement connue, donc même si l'algorithme de Little n'obtient pas cette solution optimale, il sera toujours possible d'évaluer la qualité des solutions produites par les algorithmes par rapport à la valeur optimale. Pour rappel, la mesure de la qualité d'une solution se mesure généralement par l'écart relatif 'gap' du coût de la solution trouvée 'z' par rapport au coût optimal 'z*' suivant la formule : gap = (z – z*) / z*
D'autre part, il est important également de comparer les performances de rapidité des méthodes d'optimisation en comparant leur temps d'exécution dans le cas où elles obtiennent un résultat de qualité équivalente. Par exemple, si sur de petites instances, toutes les méthodes sont capables de produire la solution optimale, il pourra être intéressant de comparer leur temps d'exécution.
Pour finir, une dernière étape pourra être envisagée qui consiste à hybrider les méthodes pour conjuguer leurs avantages. L'hybridation des techniques d'optimisation consiste à combiner 2 techniques d'optimisation pour obtenir des performances plus intéressantes que si les techniques avaient été utilisées seules. Avec les métaheuristiques RT et GA, il est possible, par exemple, de les hybrider en considérant qu'un des opérateurs de mutation d'un individu I de l'AG soit une RT dont la solution initiale est l'individu I.
Enfin, une étape complémentaire pourra consister à adapter les métaheuristiques RT et AG au problème de tournées de véhicules (VRP), une généralisation du TSP avec plusieurs véhicules et contraintes de capacité. L'objectif sera d'implémenter ces adaptations, puis d'évaluer la qualité des résultats obtenus sur des instances classiques du VRP dont la solution optimale est connue (par exemple, celles issues de la bibliothèque CVRPLIB - http://vrp.atd-lab.inf.puc-rio.br/index.php/en/ ). Une analyse comparative, similaire à celle effectuée sur les instances TSP, sera menée pour mesurer l'écart à l'optimum et les temps de calcul associés.

## Considérations techniques

Les programmes développés devront s'appuyer sur les codes sources fournis pour les TPs 3, 4 et 5, respectivement de l'algorithme de Little, Recherche Tabou et Algorithmes Génétiques. Tous les projets devront être compilable et exécutable sur les ordinateurs de Tps, donc sur des plateformes linux. Il est fortement recommandé que les 3 fichiers suivants soient présents dans le dossier racine du projet :

- un fichier README.md qui donne des indications quant au contenu du projet, comment compiler et exécuter les programmes
- un fichier makefile ou un script de compilation pour compiler les programmes et générer les exécutables
- un script d'exécution pour exécuter les programmes du projet