# Introduction :

## Mise en situation :

Nous sommes dans une ère mondiale de prise de conscience de la nécessité de réduire la consommation d’énergie et les émissions de gaz à effet de serre. Les premiers engagements mondiaux sur l’écologie ont été pris lors de la signature du protocole de Kyoto en 1997. Malheureusement il a fallu attendre 2005 pour que le protocole entre en vigueur. Les scientifiques ont constaté que les efforts entrepris pour réduire le réchauffement climatiques sont insuffisants et ne parviennent pas à ralentir le phénomène.

Depuis d’autres engagements ont vu le jour comme, par exemple, diviser par 4 les émissions de gaz à effets de serre en France. A l’heure actuelle, les pouvoirs publics ne peuvent pas forcer les entreprises et particuliers à changer leurs habitudes. La solution est donc de tenter de faire changer les comportements. Un des moyens permettant de réduire les émissions de gaz à effets de serre est l’amélioration et l'optimisation des modes de transports et déplacements.

## Qui sommes-nous ?

Nous sommes une petite structure nommée « CesiCDP », très reconnue dans le milieu de l’expérimentation de nouvelles solutions de mobilités pour les personnes et les marchandises adaptées à différents types de territoires. Nous avons travaillé avec beaucoup de client, dans toute la France, sur : le thème de la Mobilité Multimodale Intelligente, les nouvelles technologies de transport (plus économiques et moins polluante) et l’optimisation de la gestion des ressources.

Notre travail s’inscrit dans la modification des comportements des entreprises vis-à-vis du réchauffement climatique. Nous essayons d’optimiser les trajets de nos clients. Leurs demandes peuvent être variées : distribution du courrier, livraison de produits, traitement du réseau routier, ramassage des ordures …

Grâce à cela nous permettons chaque année de diminuer l’impact des entreprises françaises sur le climat.

# Description de l'algorithme :

Modélisation du problème algorithmique :

### Définition du problème formel :

L'ADEME à lancé un appel pour promouvoir la réalisation et l'expérimentation de nouvelles solutions de mobilité des personnes et des marchandises sur différents types de territoires. C'est dans cet objectif que nous devons réaliser un algorithme permettant de calculer, sur un réseau routier, une tournée permettant de relier entre elles un sous-ensemble de villes, puis de revenir à son point de départ, de façon à minimiser la distance totale parcourue.

Afin de répondre au problème de notre client, nous allons utiliser la méthode de Recherche Opérationnelle pour générer une tournée de livraison correspondant à ce problème. Nous allons ajouter dans le problème une contrainte permettant de coller un peu plus avec la réalité : notre algorithme pourra trouver une optimisation de la distance pour N camions.

Afin de prouver et de pouvoir justifier à nos clients que nous proposons un service de qualité, nous allons générer des statistiques à l’issue de la résolution du problème de recherche du chemin le plus court. Cela nous permettra de démontrer l’efficacité de notre algorithme.

### Étude de complexité :

Le problème du VRP est un problème qui est NP-Difficile. Cela veut donc dire qu’il est, au moins NP-Complet.

## Présentation du choix et description de l'algorithme :

### Fonctionnement, paramètres

Afin d'obtenir des résultats rapidement, nous avons pris la décision de rapidement découper le problème du VRP (vehicle routing problem) en plusieurs problèmes de type voyageur de commerce.

Pour ce faire, nous utilisons un algorithme de k-means permettant de calculer X points dont nous allons nous servir pour regrouper les sommets du graphe en clusters. Cet algorithme prend en paramètre deux listes X et Y, les coordonnées des points du graphe ainsi que le nombre de cluster à créer. Une fois les points réunis en clusters, nous n'avons plus qu'à résoudre de multiples problèmes du voyageur de commerce.

Pour ce faire nous utilisons un algorithme simple qui nous permet d'avoir une solution cohérente et rapidement : un algorithme glouton. L'algorithme est appliqué sur chaque cluster, il prend en entrée la liste des points du cluster ainsi que le point faisant office de départ et d'arrivée. Il calcule ensuite la distance du point de départ à tous les autres sommets et choisi le plus proche. Ce sommet devient le sommet courant et est supprimé de la liste des sommets restant. L'algorithme calcule de nouveau les distances, se déplace et fait de même jusqu'à avoir parcouru tout le sous-ensemble de points qu'est le cluster. Il revient ensuite à son point de départ et renvoie la liste des sommet dans l'ordre dans lequel il les a parcourus.

### Points positifs et négatifs de la solution

Positifs :

* Ces deux algorithmes sont rapides à concevoir et à mettre en œuvre.
* L'exécution est rapide bien que le problème soit NP-Complet puisque l'algorithme ne parcours qu'une seule fois chaque ville et n'introduit pas modification ou de mutation une fois le chemin réalisé.
* Cette implémentation nous permet de nous assurez que nous pouvons répondre à l'appel d'offre et fournir un optimum local pour le parcours des villes.
* Permet de configurer un nombre de camion effectuant les déplacements, permettant ainsi de minimiser la durée de retour des camions au dépôt.

Négatifs :

* Ces deux algorithmes combinés limites les solutions, les sommets étant "enfermés" dans des clusters, l'optimal global ne se trouve peut-être pas dans la configuration trouvée lors de l'application du k-means.
* De la même manière, un point légèrement éloigné dans un cluster ne va être rejoint qu'à la fin du tour du graph et augmenter drastiquement la distance parcourue alors qu'il aurait été préférable de ne pas prendre la distance la plus courte mais ainsi éviter un détour. C'est le problème de l'algorithme glouton, il s'enferme dans un optimum local.
* Le risque de s'enfermer dans un optimum local étant présent dans chaque cluster, la probabilité de trouver l'optimum global sur la totalité du graphe est quasiment nulle.

### Améliorations possibles

* En gardant les mêmes algorithmes et le même fonctionnement, on peut améliorer la vitesse de résolution en réalisant une matrice d'adjacence, ce qui permet de n'avoir à calculer les distances entre les villes qu'une seule fois.
* Pour permettre d'atteindre un maximum local plus proche de l'optimum global et même d'avoir beaucoup plus de chance de trouver cet optimum global, on peut introduire un nouvel algorithme, remplaçant les deux précédents, comme le recuit simulé. En effet cet algorithme permet d'éviter de s'enfermer dans un optimum local et va également pouvoir faire changer les points de clusters.

### Modélisation du problème selon le formalisme de l'algorithme

* Pour modéliser le problème en termes de programmation, les villes sont stockées dans un tableau, puis les tableaux représentant les clusters sont stockés dans un autre tableau et les tableau cluster vont être ordonné dans l'ordre de passage du camion dans les villes.

## Illustration avec différents cas de tests :

# Étude statistique du comportement expérimental, à l'aide de dataset générés aléatoirement, mais aussi éventuellement de dataset issus de la recherche scientifique :

## Statistiques descriptives, voire prédictives, du comportement de l'algorithme, mises en regard avec l’industrie :

Notre algorithme sauvegarde après son exécution :

* Dans le fichier « stats.csv »
  + Le nombre de villes sur le graphe
  + Le temps de résolution du problème
  + Le nombre de camion
* Dans le fichier « résultat.csv » :
  + Le nom du graphe résolu
  + Le chemin le plus court trouvé
  + Le nombre de camion utilisé

Afin de faire des observations sur nos résultats, nous avons généré des statistiques descriptives à partir des données du fichier « stats.csv ». Nous générons automatiquement un fichier HTML contenant les stats sur toutes les résolutions effectuées mais aussi sur les résolutions effectuées par nombre de camion utilisé.

Les statistiques générées contiennent :

* La médiane
* La moyenne
* L’intervalle de la série
* Le premier quartile
* Le troisième quartile
* La variance
* Le nuage de point correspondant (avec le nombre de villes et le temps de résolution de l’algorithme)
* La courbe de régression avec ses coefficients

Ces statistiques nous permettent de voir le lien qu’il y a entre nombre de ville et le temps de résolution de l’algorithme.

Malheureusement nous n’avons pas encore implémenté notre second algorithme. Nous ne pouvons donc pas comparer les résultats entre les deux algorithmes pour le moment.

## Exploitation des paramètres de l'instance du problème (taille et largeur du graphe, degré, nombre d'objets, taille et valeur des objets, nombre de véhicules, nombre de points de livraison...), et des paramètres de l'algorithme (température, taille de liste tabou, nombre de mutations etc.) :

Afin de répondre au besoin de notre client et également de tester notre algorithme, nous avons eu besoin de faire varier les entrées du l’instance du problème (nombre de villes, position des villes) et le paramètre de l’algorithme (nombre de cluster = nombre de camion).

Nous avons pu constater que le temps de résolution de l’algorithme varie en fonction des paramètres suivant :

* A nombre de villes égal, plus il y a de camions (clusters), plus le temps de résolution augmente
* A nombre de camion égal, plus il y a de villes, plus le temps de résolution augmente

L’augmentation de la distance entre les villes n'a pas d'impact sur le temps de résolution

## Analyse et commentaire des résultats d'analyse (qualité des solutions, temps de convergence, nombre d'itérations, espace mémoire...) :