« L'optimisation est une branche des mathématiques cherchant à modéliser, à analyser et à résoudre analytiquement ou numériquement les problèmes qui consistent à minimiser ou maximiser une fonction sur un ensemble. » Wikipédia

<https://fr.wikipedia.org/wiki/Optimisation_(math%C3%A9matiques)>

Dans le cadre de ce projet, nous allons utiliser des outils numériques pour résoudre un problème d’optimisation : la couverture de cible. Nous allons résumer rapidement le problème, les détails complets se trouvant dans le sujet en annexe 1.

**La problématique :**

Nous possédons un parc de cibles que nous voulons entièrement couvrir. Pour cela une flotte de capteurs est disponible. Chaque capteur possède sa liste de cible à portée et un cout d’activation. Nous souhaitons activer seulement les bons capteurs pour couvrir toutes les cibles pour le cout minimal.

**Modélisation linéaire :**

Le but de cette partie est de prendre un fichier d’instance en entrée (format texte) et de le transformer dans un format intelligible par le solveur linéaire GLPK (format lp). Ce type de solveur donne le résultat optimal ce qui nous permettra d’avoir un point de repère pour juger les algorithmes que nous développerons. Cependant, contrairement aux algorithmes que nous développerons, le temps de calcul de ce type de solveur ne tient pas le « passage à l’échelle » de l’augmentation du nombre de variable.

* **Fichiers d’instances**

Dans un premier temps nous avons extrait la structure suivante pour les fichiers d’instances :

[Nombre de cible=M] [Nombre de capteur=N]

*Pour i allant de 1 à N*

[Cout du capteur i] *(regroupé par 12 capteurs par ligne)*

*Fin pour*

*Pour j allant de 1 à M*

[Nombre de capteur couvrant la cible j]

[Liste des numéros des capteurs] *(regroupé par 12 capteurs par ligne)*

*Fin pour*

* **Formats lp**

Nous allons maintenant voir la structure d’un fichier « lp », plus précisément un squelette de fichier que nous allons générer.

Minimize

z :

Subject To

*Pour j allant de 1 à M*

cible\_j :

*Fin Pour*

Binaries

*Pour i allant de 1 à N*

"x"i *(par exemple « x1 » pour le capteur n°1 jusqu’à « x1000 » pour le capteur n°1000)*

*Fin Pour*

End

* **Réalisation**

Une fois le format des deux fichiers maitrisés nous avons réalisé en C++ un parser qui lit le fichier d’instance et donne un fichier lp bien formé en sortie. Etant donné que le code source est disponible avec ce document, il ne nous semble pas pertinent de développer cette partie.

IV. Algorithme génétique

1. Principes

**Les algorithmes génétiques :**

Les algorithmes génétiques utilisent des méthodes de calcul bio inspirées, plus précisément inspirées de la théorie de l’évolution. Le principe est de faire évoluer une population d’individu en se basant sur le principe de sélection naturelle pour faire tendre le résultat vers le minimum global (ou le maximum global). Le principe de diversité joue également un rôle majeur qui permet d’éviter de bloquer le résultat dans un minium local.

Schéma

2. Détails d'implémentation

Warning : Batman et les guerriers

* Init pop
* Croisement
* Mutation
* New pop
* Choix des paramètres

3. Procédures de test

* TODO avec différents paramètres + laïus

4. Résultats et interprétation

