INTRODUCTION :
LA DEMARCHE DE L'AIDE A LA DECISION

Cette introduction permet de situer l'objectif de ce cours que l'on pourrait résumer par "du problème à la décision".

Pour situer une partie du champ d'applications de l'aide à la décision, citons un extrait du site d'Eurodécision, SSII, spécialisée dans l'intégration des technologies de l'optimisation et de la recherche opérationnelle dans des applications d'aide à la décision (www.eurodecision.fr).

Cette société propose à ses clients des produits orientés vers :
- "l'optimisation stratégique et tactique d'une organisation industrielle et logistique en tenant compte des contraintes de capacités industrielles et des coûts de l'ensemble de la chaîne : fournisseurs - usines - dépôts – clients "et dans ce cadre, l'optimisation "d'un plan transport (approvisionnement, distribution ou messagerie) en tenant compte de contraintes de qualité de service et d'exploitation des sites (plages d'ouverture, capacité des hubs...)"
- "la planification annuelle des ressources humaines, comprenant l'annualisation, le positionnement des congés, la modulation et le dimensionnement des effectifs" et plus particulièrement la couverture de la charge journalière ou hebdomadaire, la construction des horaires des chauffeurs de bus (habillage)...
- des problèmes orientés vers la planification de production court terme et de l'ordonnancement.

Le champ d'application de l'aide à la décision comporte aussi des problèmes tels que :
- la constitution d'emploi du temps
- la planification et le dimensionnement de réseaux, l'optimisation du routage de trafic, l'optimisation de tarification
- les problèmes de découpe industrielle.....

et tout problème d'allocation de ressources.

Dans le cadre de ce cours, les problèmes traités seront bien sûr plus modestes et tous les domaines ne seront évidemment pas abordés. Son objectif est de sensibiliser à la démarche de l'aide à la décision et de présenter quelques-uns des outils de recherche opérationnelle sur lesquels elle s'appuie.

I Exemples d'introduction

Commençons par deux exemples simples.

Exemple 1
Il s'agit de programmer votre prochain déplacement afin de vous rendre sur votre lieu de vacances.
Le calcul de votre itinéraire repose sur l'analyse de la carte qui vous permet d'envisager les différents trajets possibles.
Et comme vous avez le choix, il faut aussi que vous spécifiez le critère que vous allez retenir : est-ce le plus rapide que vous cherchez ou celui passant par le plus de lieux touristiques ?
Si vous consultez un des très nombreux sites où l'on peut rechercher un itinéraire, on vous demandera souvent de préciser le critère que vous souhaitez privilégier.

Exemple 2
Le problème diététique.
La constitution optimale d'un régime alimentaire optimal est un problème ancien : son analyse "rationnelle" a été conduite dans les années 1930 pour le compte de l'armée américaine qui souhaitait proposer aux GI une alimentation équilibrée mais... pas trop cher.
Il s'agit de déterminer parmi un ensemble de produits possibles dont on connaît les constituants nutritifs ceux qui devront être retenus pour respecter les consignes diététiques. Plusieurs possibilités existent et il faut donc spécifier le critère selon lequel on les évalue. Pour ce problème, c'est généralement le coût qui est retenu.

Les caractéristiques communes à ces deux exemples mettent en évidence un **problème** qui consiste à sélectionner une **décision** parmi un ensemble de **décisions possibles**, de manière à **optimiser** un certain **critère.**

Ce type de problème de décision est un **problème d'optimisation.**

Un problème est une question que l'on se pose.
Analyser le problème, c'est préciser clairement :
- ce que l'on veut faire : le type de décision,
- ce que l'on peut faire : les décisions possibles,
- comment on choisit : le critère de sélection.

Si on représente :
- une décision par "x" : x peut être par exemple un chemin ou un menu,
- l'ensemble des décisions possibles par S : S peut être par exemple l'ensemble des parcours qui vous intéresse ou l'ensemble des menus possibles,
- le critère par une application f qui à une décision x associe son évaluation f(x) : f(x) peut, par exemple, représenter la longueur du parcours x ou le coût du menu x.
La forme très générale d'un problème d'optimisation est :

$$\left\{\begin{array}{l}\text{Min } f(x)\\ x \in S\end{array}\right. \quad \text{ou} \quad \left\{\begin{array}{l}\text{Max } f(x)\\ x \in S\end{array}\right.$$

La **décision** est représentée par une **variable de décision** x.
Les **décisions possibles** sont représentées par un ensemble S symbolisant les **contraintes** portant sur les décisions.
Le **critère** est représenté par une fonction appelée **fonction objectif** ou plus brièvement **objectif.**

II La démarche de l'aide à la décision et de la recherche opérationnelle

A - Du problème au modèle

Considérons le problème :

$$\begin{cases} \text{Min } f(x) \\ x \in S \end{cases} \quad \text{ou} \quad \begin{cases} \text{Max } f(x) \\ x \in S \end{cases}$$

Résoudre ce problème, c'est trouver parmi toutes les valeurs possibles de x dans S celle(s) qui donnera à l'objectif la valeur la plus petite ou la plus grande possible.

Sans autre précision sur la nature de x, de S ou de f, la seule possibilité pour résoudre le problème est d'énumérer toutes les solutions.

Considérons le problème suivant :
On doit organiser une tournée de livraisons d'un camion qui, partant d'un entrepôt, doit se rendre successivement chez un certain nombre de clients.
Enumérer la totalité des parcours possibles revient à lister toutes les permutations des clients.
Par exemple, avec 4 clients a, b, c, d, l'ordre des visites possibles est :
a b c d   a b d c   a c b d   a c d b   a d b c   a d c b .....
Ici nous n'avons que celles qui commencent par le client a !

Pour les dénombrer, on peut constater qu'il y a 4 choix possibles pour le client placé en 1, puis dans chaque cas 3 pour le deuxième, puis 2 pour le troisième, le quatrième étant alors celui qui reste, soit en tout 4*3*2*1 possibilités.

Rappelons que n! = 1*2*3....*n (factorielle n) représente le nombre de permutations de n éléments.
Pour 4 clients il y a donc 4! = 24 possibilités.
Ceci devient vite totalement inexploitable.
Pour n = 10 on a déjà 10 ! = 10*9*8*... *3*2*1 = 3 628 800 tournées possibles !

L'énumération étant généralement impossible, pour résoudre un problème il faut construire un **modèle** qui lui soit adapté, c'est-à-dire préciser la nature de x, de S, et de f(x).

Parmi les différentes acceptions du mot modèle, le petit Larousse® donne la définition suivante : "structure formalisée utilisée pour rendre compte d'un ensemble de phénomènes qui possèdent entre eux certaines relations".

Le problème étudié se place dans un certain contexte qui va être traduit par des paramètres exogènes sur lequel le décideur ne peut agir. Ce sont des données pour le problème.
Le modèle doit servir pour explorer les solutions ou simuler toutes les relations entre les données et les variables.

Il peut prendre la forme d'un modèle mathématique, auquel cas il s'appuiera sur un ensemble d'équations mathématiques et de relations. Il peut aussi exploiter des représentations graphiques.

La modélisation procède nécessairement par simplifications et omissions. Le modèle ne peut être qu'une représentation partielle et schématique. Seuls les éléments jugés pertinents en vue de l'exploration des solutions sont retenus.

Dans la construction d'un modèle, il faudra tenir compte du caractère plus au moins aléatoire de l'environnement : on peut être dans un environnement déterministe ou présentant un certain degré d'incertitude mais modélisable par des lois de probabilité, mais on peut aussi se trouver face à l'incertitude.

Un autre aspect de la conception d'un modèle concerne les critères qui seront mis en avant pour évaluer les décisions.
Dans certains cas, il peut être suffisant de ne retenir qu'un critère unique : ce sera généralement le cas de problèmes à caractère opérationnel. Dans un contexte de décisions plus stratégiques, on sera amené à prendre en compte plusieurs critères qui risquent d'être contradictoires.

Il n'existe pas vraiment de règles pour construire un modèle.
"On a pu prétendre que la modélisation constituait d'avantage un art qu'une technique. Plus modestement, nous pensons qu'elle relève plutôt d'un artisanat alliant rigueur scientifique et savoir-faire pragmatique" (Extrait de "*Aide à la décision : Une approche par les cas" Vallin- Vanderpooten ELLIPSES)*

Le premier objectif de ce cours est de donner quelques **exemples** de **modèles** permettant de représenter certains problèmes d'optimisation dans un contexte déterministe.

B - Du modèle à la solution

Une fois le problème correctement représenté, il s'agira alors de le résoudre. Pour cela, pour chaque type de modèle, on dispose de méthodes de résolution.

La nature du problème peut conduire à des modèles pour lesquels il est encore aujourd'hui très difficile d'obtenir une solution optimale. Dans ce cas, on s'attachera essentiellement à construire en un temps raisonnable une bonne solution en utilisant des méthodes approchées encore appelées "heuristiques".
Le modèle peut aussi être exploité comme outil de simulation : les variables et données sont liées entre elles par des relations. On peut utiliser le modèle pour simuler la conséquence de variation de certaines données ou paramètres du problème et mesurer l'impact de certaines décisions dans un contexte où l'expérimentation est souvent impossible.

Le deuxième objectif du cours est de donner pour quelques modèles une méthode de résolution.

Dans la plupart des cas que nous verrons, ces méthodes reposent sur des processus itératifs, c'est à dire des **algorithmes** de détermination des solutions.

Cette partie de la démarche concerne plutôt la recherche opérationnelle dont la définition issue du Larousse® est :
« Méthode d'**analyse scientifique** orientée vers la recherche de la **meilleure façon** de prendre des **décisions** pour aboutir aux **meilleurs** résultats ».

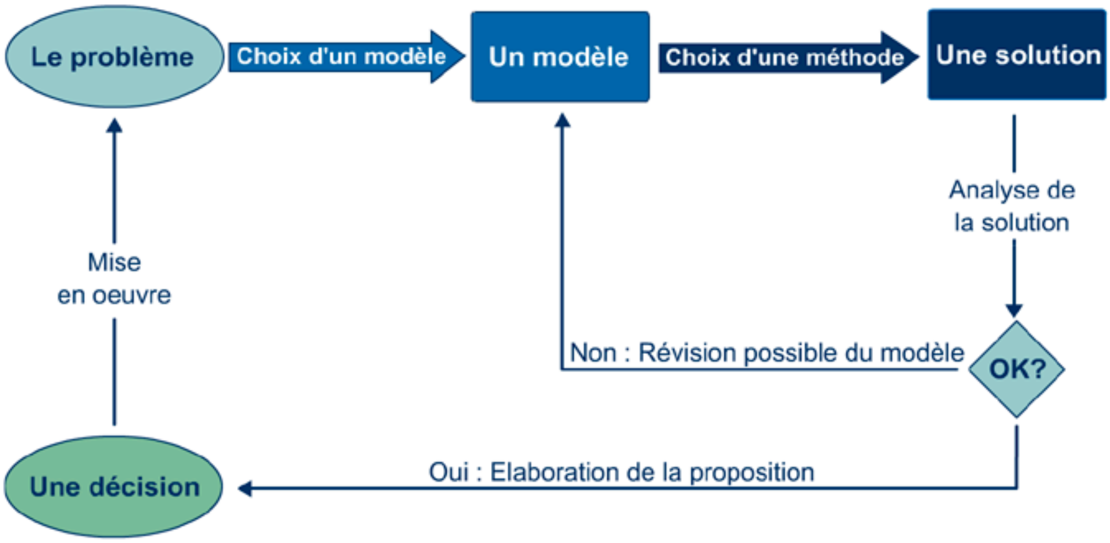
C - De la solution à la décision

Une fois une solution élaborée, on peut dans un premier temps l'analyser dans l'objectif de valider le modèle. Nous avons vu que le modèle n'était qu'une représentation schématique de la réalité. Une solution pourrait mettre en évidence que certaines contraintes n'ont pas, ou ont mal, été prises en compte.
Si ce n'est pas le cas, il s'agira alors de traduire la solution obtenue dans les termes du problème tel qu'il avait été posé et laisser au décideur le soin de l'accepter, la modifier ou... la rejeter !

D - Schéma de la démarche

La démarche précédente qui illustre le passage du problème à la décision, peut être représentée dans le schéma suivant - qui lui-même ne constitue qu'un modèle ! - :

La démarche de l'aide à la décision et de la recherche opérationnelle



III Notions élémentaires sur les algorithmes

Nous présentons ici les éléments nécessaires à la compréhension des algorithmes qui seront présentés dans ce cours.

Définition d'un algorithme
Un algorithme est une suite finie d'opérations (élémentaires) qui, à partir de données, conduit à la solution d'un problème.

Le terme algorithme a pour origine la traduction en latin, au XIIIème siècle, d'un mémoire de Mohammed Ibn Musa Abu Djefar Al-Khwarizmi commençant par : « Algoritmi dixit... ».

Pour un modèle donné, il faut avoir des algorithmes de résolution qui soient :
- efficaces de manière à pouvoir traiter des "grands" problèmes ; nous verrons que ce n'est pas toujours le cas,
- robustes : ils ne doivent pas être tributaires d'une mauvaise appréciation des données,
- souples : ils doivent pouvoir s'adapter aisément à des cas particuliers.

Ecriture d'un algorithme en pseudo-langage
Avant de traduire un algorithme en langage de programmation (C, Java, Visual basic ou autre), il est d'abord décrit dans un langage de plus haut niveau qui peut commencer par être... le français.
On peut aussi utiliser un jeu limité d'instructions permettant de décrire clairement son principe. Ces instructions sont écrites à partir d'un langage simplifié appelé "pseudo-langage".
Nous rappelons ici les principales instructions nécessaires pour décrire de manière plus ou moins formalisée un algorithme.

Affectation d'une valeur à une variable
Une instruction d'affectation permet de donner une valeur à une variable. On peut l'écrire sous la forme :
Poser x = 100 ou x : = 100 ou x <- 100

Description d'une suite d'opérations
Si plusieurs instructions doivent être exécutées successivement, on les encadrera par les mots FAIRE et FINFAIRE.

```
FAIRE
   Instruction 1
   Instruction 2
   ….
FINFAIRE
```

Instruction de test permettant des branchements
Ceci permet d'exécuter une suite d'instructions si une condition est réalisée et une autre si elle ne l'est pas.

```
SI condition vérifiée
          ALORS  suite 1 d'instructions
          SINON  suite 2 d'instructions
FINSI
```

La partie SINON est optionnelle.
On peut aussi avoir plusieurs instructions SI et SINON imbriquées.

Instruction de boucle permettant la répétition
Lorsque la même séquence d'instructions doit être répétée tant qu'une condition n'est pas vérifiée, on utilisera la commande TANTQUE.
Si cette suite doit être exécutée un nombre précis de fois, on pourra utiliser un compteur et l'instruction POUR qui permet d'incrémenter le compteur de 1 à chaque passage.

```
TANTQUE "condition vérifiée"
    FAIRE
        ........
        ........
    FINFAIRE
FINTANTQUE


POUR compteur de 1 à n
    FAIRE …..
        ........
        ........
    FINFAIRE
FINPOUR
```

Exemples d'algorithme
Exemple 1
Calcul de la somme des carrés des n premiers nombres entiers.

```
Lire n
Somme : = 0
Résultat : = 0
POUR i de 1 à n
     FAIRE
```

```
        I2 : = i*i
        Somme : = Somme + I2
    FINFAIRE
FINPOUR
Résultat : = Somme
LIRE Résultat
```

Le principe de cet algorithme est, à chaque itération, d'additionner à une variable appelée "Somme" le carré du nombre entier "i", i prenant successivement toutes les valeurs de 1 à n.

Exemple 2

Considérons les deux algorithmes suivants :

Algorithme 1

```
LIRE n
LIRE x
Résultat : = 1

POUR i de 1 à n
      FAIRE
                  Terme : = 1
                  POUR j de 1 à i
                  Terme : = Terme * (x / j )
                  FINPOUR
                  Résultat : = Résultat + Terme
        FINFAIRE
FINPOUR
LIRE Résultat
FIN
```

Algorithme 2

```
LIRE n
LIRE x
Résultat : = 1
Terme : = 1

POUR i de 1 à n
      FAIRE
                  Terme : = Terme * (x / i )
                  Résultat : = Résultat + Terme
        FINFAIRE
FINPOUR
LIRE Résultat
FIN
```

Ces deux algorithmes fournissent le même résultat :

Résultat = $1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + ... + \frac{x^n}{n!}$

qui donne une valeur approchée de e^x pour les petites valeurs de x.

Dans le premier algorithme, on recalcule à chaque itération le terme $\frac{x^i}{i!}$ en multipliant progressivement la variable Terme par x/1 puis par x/2 ....puis par x/i ce qui donne bien $\frac{x^i}{i!}$ après i multiplications et i divisions soit 2i opérations.

On recommence pour chaque valeur de i.

Le nombre d'opérations que l'on va devoir faire est donc de l'ordre de :
- première itération : calcul de Terme * (x / 1) soit 2 opérations
- deuxième itération : calcul de $\frac{x^2}{2!}$ soit 4 opérations
- ième itération : calcul de $\frac{x^i}{i!}$ soit 2 i opérations ; on multiplie i fois x par lui-même et le calcul de i! nécessite i multiplications
- nième et dernière itération : calcul de $\frac{x^n}{n!}$ soit 2 n opérations

En tout on aura donc effectué de l'ordre de :
2 (1 +2+ ....+ n) = 2 n(n+1)/2 = n(n+1) opérations.

Dans le deuxième algorithme, pour calculer $\frac{x^i}{i!}$ on garde en mémoire le calcul de $\frac{x^{i-1}}{(i-1)!}$.

Ceci permet d'arriver au résultat en n'effectuant que 2 opérations à chaque itération et donc de n'avoir en tout que de l'ordre de 2n opérations à effectuer.

Ce deuxième algorithme est donc plus efficace que le premier, le nombre d'opérations croît comme n alors que pour l'algorithme 1 le nombre d'opérations croit comme n^2.

Pour un problème donné, on peut avoir plusieurs algorithmes de résolution. Afin de comparer leur performance respective, on s'attachera à déterminer le nombre d'opérations que nécessite leur mise en oeuvre, indépendamment de la machine utilisée et ceci lorsque la taille du problème devient importante.

Dans les dernières leçons de ce cours, nous présenterons des problèmes pour lesquels les algorithmes actuellement connus conduisent à un nombre d'opérations, et donc à un temps de calcul, qui croit de manière exponentielle avec la taille du problème, ce qui interdit de les mettre en oeuvre sur des "grands" problèmes.