



Métaheuristiques

Introduction aux Métaheuristiques

Imane EL MALKI et Pr. Imad HAFIDI

imane.elmalki@gmail.com

Master Big Data et Aide à la décision



Introduction

Classification des méthodes

Méthodes de recherche locales

Principe

Méthode de Descente

Recueil simulé

Méthodes d'évolution

Introduction

Algorithmes génétiques

Motivation

- ▶ On s'intéresse au problème d'optimisation,

$$\min_{x \in \mathcal{X}} \{f(x) : x \in \Omega\}$$

Où \mathcal{X} est un ensemble discret et fini, et $\Omega \subseteq \mathcal{X}$ est l'ensemble des solutions réalisables. La fonction objectif f prend ses valeurs sur \mathcal{X} .

Motivation

- ▶ On s'intéresse au problème d'optimisation,

$$\min_{x \in \mathcal{X}} \{f(x) : x \in \Omega\}$$

Où \mathcal{X} est un ensemble discret et fini, et $\Omega \subseteq \mathcal{X}$ est l'ensemble des solutions réalisables. La fonction objectif f prend ses valeurs sur \mathcal{X} .

- ▶ On peut se ramener au problème

$$f^* = \min_{x \in \Omega} f(x)$$

dont l'ensemble des solutions est $\operatorname{argmin}_{x \in \Omega} f(x)$

Motivation

- ▶ On s'intéresse au problème d'optimisation,

$$\min_{x \in \mathcal{X}} \{f(x) : x \in \Omega\}$$

Où \mathcal{X} est un ensemble discret et fini, et $\Omega \subseteq \mathcal{X}$ est l'ensemble des solutions réalisables. La fonction objectif f prend ses valeurs sur \mathcal{X} .

- ▶ On peut se ramener au problème

$$f^* = \min_{x \in \Omega} f(x)$$

dont l'ensemble des solutions est $\operatorname{argmin}_{x \in \Omega} f(x)$

- ▶ L'intérêt de conserver la description avec deux ensembles \mathcal{X} et Ω est de bien comprendre qu'un algorithme peut générer des points intermédiaires non-réalisables, c'est-à-dire dans \mathcal{X} et non dans Ω .

Motivation



- ▶ l'ensemble Ω peut correspondre à des points irréalisables d'un point de vue logique, mais ils doivent être pénalisées dans l'objectif f .

Motivation

- ▶ l'ensemble Ω peut correspondre à des points irréalisables d'un point de vue logique, mais ils doivent être pénalisées dans l'objectif f .

- ▶ On considère que le problème d'optimisation est \mathcal{NP} -dur et que l'on dispose pas d'un algorithme en temps polynomial pour le résoudre, ou de méthode classique efficace.

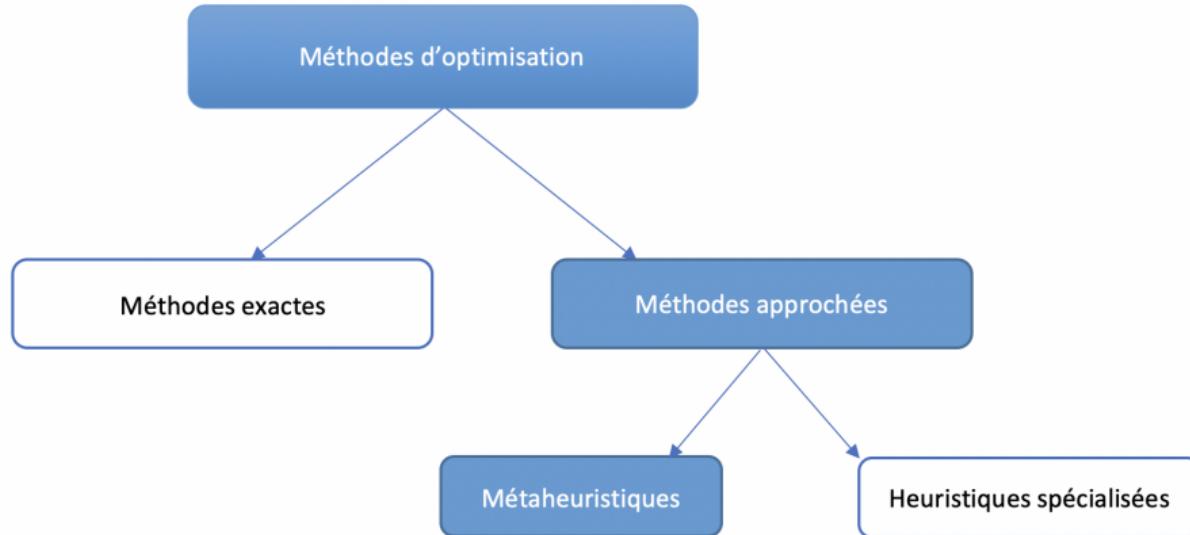
Motivation

- ▶ l'ensemble Ω peut correspondre à des points irréalisables d'un point de vue logique, mais ils doivent être pénalisées dans l'objectif f .

- ▶ On considère que le problème d'optimisation est \mathcal{NP} -dur et que l'on dispose pas d'un algorithme en temps polynomial pour le résoudre, ou de méthode classique efficace.

- ▶ On va donc considérer une **métaheuristique** afin d'obtenir un point de bonne qualité dans un laps de temps raisonnable.

Classification des méthodes d'optimisation



Définitions



- Une **structure de voisinage** (ou un voisinage) est une fonction N qui associe un sous-ensemble de \mathcal{X} à tout point $x \in \mathcal{X}$. Un point $x' \in N(x)$ est dite **voisin** de x .

Définitions

- ▶ Une **structure de voisinage** (ou un **voisinage**) est une fonction N qui associe un sous-ensemble de \mathcal{X} à tout point $x \in \mathcal{X}$. Un point $x' \in N(x)$ est dite **voisin** de x .
- ▶ Un point $x \in \Omega$ est un **minimum local** relativement à la structure de voisinage N si $f(x) \leq f(x')$ pour tout $x' \in N(x) \cap \Omega$.

Définitions

- ▶ Une **structure de voisinage** (ou un **voisinage**) est une fonction N qui associe un sous-ensemble de \mathcal{X} à tout point $x \in \mathcal{X}$. Un point $x' \in N(x)$ est dite **voisin** de x .
- ▶ Un point $x \in \Omega$ est un **minimum local** relativement à la structure de voisinage N si $f(x) \leq f(x')$ pour tout $x' \in N(x) \cap \Omega$.
- ▶ Un point $x \in \Omega$ est un **minimum global** si $f(x) \leq f(x')$ pour tout $x' \in \Omega$. On appellera x une **solution**.

Définitions

- ▶ Une **structure de voisinage** (ou un **voisinage**) est une fonction N qui associe un sous-ensemble de \mathcal{X} à tout point $x \in \mathcal{X}$. Un point $x' \in N(x)$ est dite **voisin** de x .
- ▶ Un point $x \in \Omega$ est un **minimum local** relativement à la structure de voisinage N si $f(x) \leq f(x')$ pour tout $x' \in N(x) \cap \Omega$.
- ▶ Un point $x \in \Omega$ est un **minimum global** si $f(x) \leq f(x')$ pour tout $x' \in \Omega$. On appellera x une **solution**.
- ▶ Les voisinages dépendent du problème. Cet aspect est donc laissé générique lors de la définition d'une métaheuristique.

Définitions

Heuristique

Vient du verbe grec heuriskein qui signifie **trouver**.

Une heuristique est un algorithme qui permet de trouver dans un temps polynomial une solution réalisable, tenant en compte d'une fonction objectif, pas nécessairement optimale (approchée) ou exacte pour un problème d'optimisation difficile.

Définitions

Heuristique

Vient du verbe grec heuriskein qui signifie **trouver**.

Une heuristique est un algorithme qui permet de trouver dans un temps polynomial une solution réalisable, tenant en compte d'une fonction objectif, pas nécessairement optimale (approchée) ou exacte pour un problème d'optimisation difficile.

Remarque :

Ce type de méthodes traduit une stratégie (une manière de penser) en s'appuyant sur la connaissance du problème. **Une heuristique est spécifique** au problème et ne peut pas être généralisée.

Définitions

Heuristique

Vient du verbe grec heuriskein qui signifie **trouver**.

Une heuristique est un algorithme qui permet de trouver dans un temps polynomial une solution réalisable, tenant en compte d'une fonction objectif, pas nécessairement optimale (approchée) ou exacte pour un problème d'optimisation difficile.

Remarque :

Ce type de méthodes traduit une stratégie (une manière de penser) en s'appuyant sur la connaissance du problème. **Une heuristique est spécifique** au problème et ne peut pas être généralisée.

Métaheuristique

vient des mots grecs meta **au delà** et heuriskein **trouver**.

Une métaheuristique est une heuristique générique qu'il faut adapter à chaque problème.

Définitions



Remarque :

Pour des problèmes d'optimisation (NP) où la recherche d'une solution exacte (optimale) est difficile (coût exponentiel), on peut se contenter d'une solution satisfaisante donnée par une heuristique avec un coût plus faible.

Même pour trouver une solution exacte, la connaissance de solutions approchées permet de rendre l'algorithme meilleur, tout en conservant, bien sûr la complexité exponentielle.

Caractéristiques des métahéuristiques



- ▶ Les métahéuristiques sont des stratégies qui permettent de guider la recherche d'une solution.

Caractéristiques des métahéuristiques



- ▶ Les métahéuristiques sont des stratégies qui permettent de guider la recherche d'une solution.

- ▶ Le but visé par les métahéuristiques est d'explorer l'espace de recherche efficacement afin de déterminer des points (presque) optimaux.

Caractéristiques des métahéuristiques



- ▶ Les métahéuristiques sont des stratégies qui permettent de guider la recherche d'une solution.
- ▶ Le but visé par les métahéuristiques est d'explorer l'espace de recherche efficacement afin de déterminer des points (presque) optimaux.
- ▶ Les techniques qui constituent des algorithmes de type métahéuristique vont de la simple procédure de recherche locale à des processus d'apprentissage complexes.

Caractéristiques des métaheuristiques



- ▶ Les métaheuristiques sont des stratégies qui permettent de guider la recherche d'une solution.
- ▶ Le but visé par les métaheuristiques est d'explorer l'espace de recherche efficacement afin de déterminer des points (presque) optimaux.
- ▶ Les techniques qui constituent des algorithmes de type métaheuristique vont de la simple procédure de recherche locale à des processus d'apprentissage complexes.
- ▶ Les métaheuristiques sont en général non-déterministes et ne donnent aucune garantie d'optimalité.
- ▶ Les métaheuristiques peuvent contenir des mécanismes qui permettent d'éviter d'être bloqué dans des régions de l'espace de recherche.

Caractéristiques des mét heuristicques



- ▶ Les concepts de base des mét heuristicques peuvent être décrits de manière abstraite, sans faire appel à un problème spécifique.

Caractéristiques des métahéuristiques



- ▶ Les concepts de base des métahéuristiques peuvent être décrits de manière abstraite, sans faire appel à un problème spécifique.
- ▶ Les métahéuristiques peuvent faire appel à des heuristiques qui tiennent compte de la spécificité du problème traité, mais ces heuristiques sont contrôlées par une stratégie de niveau supérieur.

Caractéristiques des métaheuristiques



- ▶ Les concepts de base des métaheuristiques peuvent être décrits de manière abstraite, sans faire appel à un problème spécifique.
- ▶ Les métaheuristiques peuvent faire appel à des heuristiques qui tiennent compte de la spécificité du problème traité, mais ces heuristiques sont contrôlées par une stratégie de niveau supérieur.
- ▶ Les métaheuristiques peuvent faire usage de l'expérience accumulée durant la recherche de l'optimum, pour mieux guider la suite du processus de recherche.

Critères heuristiques



Qu'est ce qu'une bonne heuristique ? (Manque de résultats théoriques) :

- ▶ Elle est de complexité raisonnable (idéalement polynomiale mais en tout cas efficace en pratique)

Critères heuristiques



Qu'est ce qu'une bonne heuristique ? (Manque de résultats théoriques) :

- ▶ Elle est de complexité raisonnable (idéalement polynomiale mais en tout cas efficace en pratique)
- ▶ Robuste : elle fournit le plus souvent une solution proche de l'optimum, la probabilité d'obtenir une solution de mauvaise qualité est faible

Critères heuristiques

Qu'est ce qu'une bonne heuristique ? (Manque de résultats théoriques) :

- ▶ Elle est de complexité raisonnable (idéalement polynomiale mais en tout cas efficace en pratique)
- ▶ Robuste : elle fournit le plus souvent une solution proche de l'optimum, la probabilité d'obtenir une solution de mauvaise qualité est faible
- ▶ Elle est simple à mettre en oeuvre

Une heuristique est spécifique à un problème et ne peut pas être généralisée

Concepts fondamentaux des mét-heuristiques

Les mét-heuristiques ne nécessitent pas une connaissance particulière sur les problèmes d'optimisation à résoudre. Il suffit d'associer une ou plusieurs variables à une solution (ou plusieurs solutions).

Concepts fondamentaux des métaheuristiques

Les métaheuristiques ne nécessitent pas une connaissance particulière sur les problèmes d'optimisation à résoudre. Il suffit d'associer une ou plusieurs variables à une solution (ou plusieurs solutions).

Il existe deux points critiques pour toute métaheuristique :

Concepts fondamentaux des métaheuristiques

Les métaheuristiques ne nécessitent pas une connaissance particulière sur les problèmes d'optimisation à résoudre. Il suffit d'associer une ou plusieurs variables à une solution (ou plusieurs solutions).

Il existe deux points critiques pour toute métaheuristique :

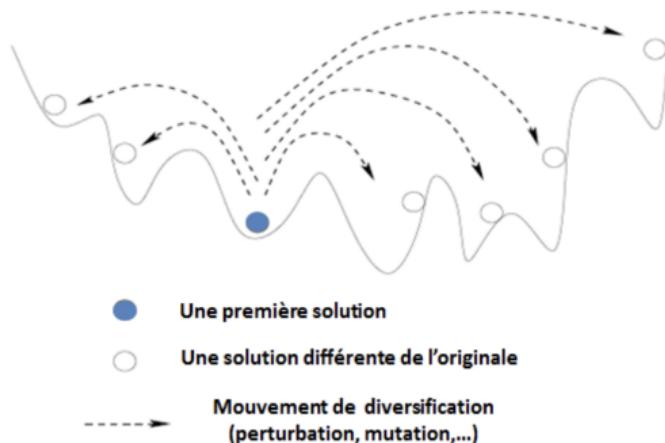
- ▶ **Diversification** : est un mécanisme pour une exploration assez large de l'espace de recherche.

Concepts fondamentaux des métaheuristiques

Les métaheuristiques ne nécessitent pas une connaissance particulière sur les problèmes d'optimisation à résoudre. Il suffit d'associer une ou plusieurs variables à une solution (ou plusieurs solutions).

Il existe deux points critiques pour toute métaheuristique :

- ▶ **Diversification** : est un mécanisme pour une exploration assez large de l'espace de recherche.



Concepts fondamentaux des mét heuristicques



On peut dire que, La diversification essaye de déplacer les solutions dans d'autres zones de l'espace de recherche.

Concepts fondamentaux des métaheuristiques



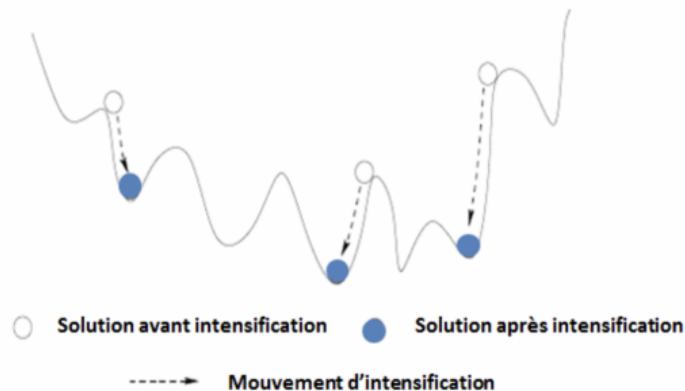
On peut dire que, La diversification essaye de déplacer les solutions dans d'autres zones de l'espace de recherche.

- ▶ **Intensification :** L'intensification vise à forcer une solution donnée à tendre vers l'optimum local de la zone à laquelle elle est attachée. Elle permet une exploitation de l'information accumulée durant la recherche.

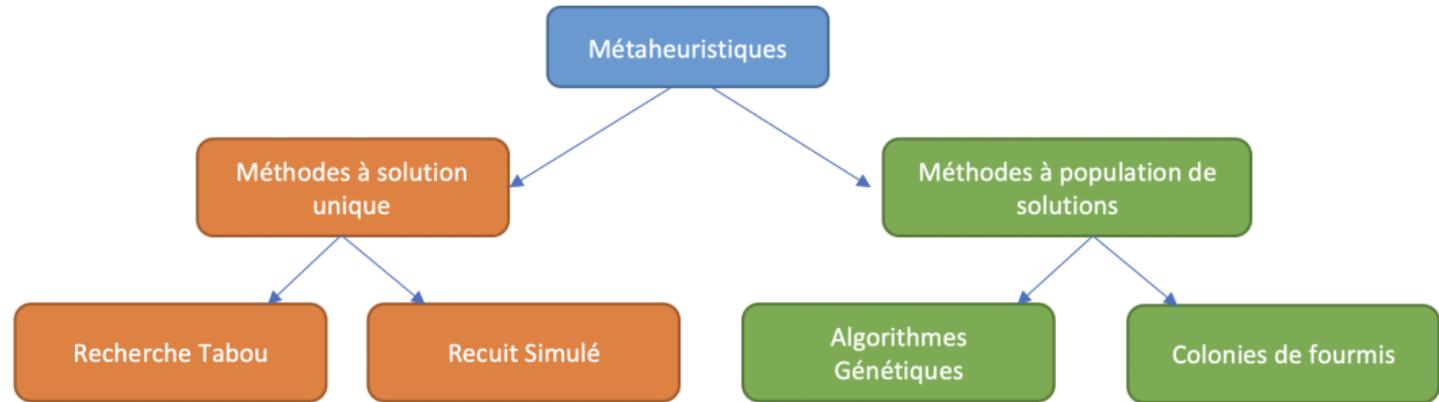
Concepts fondamentaux des mét-heuristiques

On peut dire que, La diversification essaye de déplacer les solutions dans d'autres zones de l'espace de recherche.

- ▶ **Intensification :** L'intensification vise à forcer une solution donnée à tendre vers l'optimum local de la zone à laquelle elle est attachée. Elle permet une exploitation de l'information accumulée durant la recherche.



Classification des méthodes



Méthodes de recherche locales : Présentation



Présentation

Afin de résoudre les problèmes d'optimisation combinatoire, les métaheuristiques à solution unique améliorent une solution unique. Ils pourraient être considérés comme des trajectoires de recherche dans l'espace de recherche du problème à résoudre. Les trajectoires sont effectués par des procédures itératives qui passent de la solution actuelle à une autre dans l'espace de recherche. Elles démontrent leur efficacité pour résoudre divers problèmes d'optimisation dans différents domaines.

Méthodes de recherche locales : Présentation



Principe

Le principe des méthodes de recherche locales applique de manière itérative les procédures :

- **Génération** : un ensemble de solutions candidates est généré à partir des solutions actuelles.
Cet ensemble $C(s)$ est généralement obtenu par transformations locales de la solution.
- **Sélection** : une sélection est effectuée dans l'ensemble de solutions candidates $C(s)$ pour remplacer la solution actuelle, c'est-à-dire qu'une solution $s^* \in C(s)$ est choisie pour être la nouvelle solution.

Remarque :

Ce processus itère jusqu'à un critère d'arrêt donné. Les phases de génération et de sélection peuvent être sans mémoire. Dans ce cas, les deux procédures sont basées uniquement sur la solution actuelle.

Méthodes de recherche locales : Présentation

Principe

Le principe des méthodes de recherche locales applique de manière itérative les procédures :

- **Génération** : un ensemble de solutions candidates est généré à partir des solutions actuelles.
Cet ensemble $C(s)$ est généralement obtenu par transformations locales de la solution.
- **Sélection** : une sélection est effectuée dans l'ensemble de solutions candidates $C(s)$ pour remplacer la solution actuelle, c'est-à-dire qu'une solution $s^* \in C(s)$ est choisie pour être la nouvelle solution.

Remarque :

Ce processus itère jusqu'à un critère d'arrêt donné. Les phases de génération et de selection peuvent être sans mémoire. Dans ce cas, les deux procédures sont basées uniquement sur la solution actuelle.

Où bien, une partie de l'historique de la recherche stockée dans une mémoire peut être utilisée.

Méthodes de recherche locales : Présentation

Algorithme 1 : Algorithme général d'une méthode de recherche locale (Trajectoire)

Input : Solution initiale S_0

Result : Solution S

While (condition d'arrêt n'est pas vérifié) **do**

 /* Génération d'un ensemble de solutions à partir du voisinage de la solution S_t */

 Generate ($C(S_t)$)

 /* Sélection d'une solution à partir de l'ensemble $C(S_t)$ pour remplacer la solution courante S_t */

S_{t+1} = Select ($C(S_t)$);

$t = t + 1$

End while

Return : Meilleur solution trouvé durant toutes les itérations.

Méthodes de recherche locales : Présentation

Génération de la solution initiale

Pour générer la solution initiale deux stratégies principales sont utilisées : une approche **aléatoire** et une approche **glouton**. Il y a toujours un compromis à faire entre l'utilisation de solutions initiales aléatoires et gourmandes en termes de qualité des solutions et de temps de calcul.

Générer une solution initiale aléatoire est une opération rapide, mais la métaheuristique peut nécessiter un nombre beaucoup plus grand des itérations pour converger. Pour accélérer la recherche, une heuristique glouton peut être utilisée. En effet, dans la plupart des cas, les algorithmes gloutons ont une complexité polynomiale réduite.

Méthodes de recherche locales : Présentation

Génération de la solution initiale

Pour générer la solution initiale deux stratégies principales sont utilisées : une approche **aléatoire** et une approche **glouton**. Il y a toujours un compromis à faire entre l'utilisation de solutions initiales aléatoires et gourmandes en termes de qualité des solutions et de temps de calcul.

Générer une solution initiale aléatoire est une opération rapide, mais la métaheuristique peut nécessiter un nombre beaucoup plus grand des itérations pour converger. Pour accélérer la recherche, une heuristique glouton peut être utilisée. En effet, dans la plupart des cas, les algorithmes gloutons ont une complexité polynomiale réduite.

Remarque :

L'utilisation des heuristiques gloutons conduit souvent à des optima locaux de meilleure qualité. Par conséquent, les méthodes trajectoires nécessiteront, en général, moins d'itérations pour converger vers un optimum local.

Méthodes de recherche locales : Présentation

Génération de la solution initiale

Pour générer la solution initiale deux stratégies principales sont utilisées : une approche **aléatoire** et une approche **glouton**. Il y a toujours un compromis à faire entre l'utilisation de solutions initiales aléatoires et gourmandes en termes de qualité des solutions et de temps de calcul.

Générer une solution initiale aléatoire est une opération rapide, mais la métaheuristique peut nécessiter un nombre beaucoup plus grand des itérations pour converger. Pour accélérer la recherche, une heuristique glouton peut être utilisée. En effet, dans la plupart des cas, les algorithmes gloutons ont une complexité polynomiale réduite.

Remarque :

L'utilisation des heuristiques gloutons conduit souvent à des optima locaux de meilleure qualité. Par conséquent, les méthodes trajectoires nécessitera, en général, moins d'itérations pour converger vers un optimum local.

Toutefois, cela ne signifie pas que l'utilisation de meilleures solutions comme solutions initiales conduira toujours à de meilleurs optima locaux.

Méthodes de recherche locales : Présentation



Les méthodes de recherche locales sont des métaheuristiques à base de solution unique.
La recherche se fait localement sur un ensemble de voisin de la solution actuelle.

Méthodes de recherche locales : Présentation



Les méthodes de recherche locales sont des métaheuristiques à base de solution unique.
La recherche se fait localement sur un ensemble de voisin de la solution actuelle.

Le principe se pause sur la construction d'un voisinage de solution, c-à-d, à partir d'une solution de départ S en engendre une suite finie de solution S' détermiées comme plus proche.

Méthodes de recherche locales : Présentation



Les méthodes de recherche locales sont des métaheuristiques à base de solution unique.
La recherche se fait localement sur un ensemble de voisin de la solution actuelle.

Le principe se pause sur la construction d'un voisinage de solution, c-à-d, à partir d'une solution de départ S en engendre une suite finie de solution S' détermiées comme plus proche.

Le voisinage $N(S)$ d'une solution S est souvent défini de manière implicite en terms de **mouvements** (Transformations élémentaires).

Méthodes de recherche locales : Présentation

Les méthodes de recherche locales sont des métaheuristiques à base de solution unique.
La recherche se fait localement sur un ensemble de voisin de la solution actuelle.

Le principe se pause sur la construction d'un voisinage de solution, c-à-d, à partir d'une solution de départ S en engendre une suite finie de solution S' détermiées comme plus proche.

Le voisinage $N(S)$ d'une solution S est souvent défini de manière implicite en terms de **mouvements** (Transformations élémentaires).

Définition :

- Une transformation élémentaire est une modification locale apportée à une solution S .
- Notons l'application d'un mouvement m à une solution S par $S \oplus m$
- Soit \mathcal{M} l'ensemble des mouvements possibles alors,

$$N(S) = \{S \oplus m / m \in \mathcal{M}, s \oplus m \in \mathcal{X}\}$$

Méthodes de recherche locales : Principe



On cite quelque transformations élémentaires :

Méthodes de recherche locales : Principe



On cite quelque transformations élémentaires :

- ▶ **Complément** : pour une solution codé en binaire la transformation élémentaire est de remplacer un bit quelconque par son complémentaire :

Méthodes de recherche locales : Principe



On cite quelque transformations élémentaires :

- **Complément** : pour une solution codé en binaire la transformation élémentaire est de remplacer un bit quelconque par son complémentaire :

$$00\textcolor{blue}{0}111 \longrightarrow 00\textcolor{red}{1}111$$

Méthodes de recherche locales : Principe



On cite quelque transformations élémentaires :

- ▶ **Complément** : pour une solution codé en binaire la transformation élémentaire est de remplacer un bit quelconque par son complémentaire :

$$000111 \longrightarrow 00\mathbf{1}111$$

- ▶ **Echange** : pour une solution codé en caractère (ou binaire), permutez deux caractères données :

Méthodes de recherche locales : Principe

On cite quelque transformations élémentaires :

- ▶ **Complément** : pour une solution codé en binaire la transformation élémentaire est de remplacer un bit quelconque par son complémentaire :

$$00\textcolor{blue}{0}111 \longrightarrow 00\textcolor{red}{1}111$$

- ▶ **Echange** : pour une solution codé en caractère (ou binaire), permutez deux caractères données :

$$\textcolor{blue}{A}\textcolor{black}{B}\textcolor{black}{C}\textcolor{black}{D}\textcolor{red}{E} \longrightarrow \textcolor{red}{A}\textcolor{black}{E}\textcolor{black}{C}\textcolor{black}{D}\textcolor{blue}{B}$$

Méthodes de recherche locales : Principe



On cite quelque transformations élémentaires :

- ▶ **Complément** : pour une solution codé en binaire la transformation élémentaire est de remplacer un bit quelconque par son complémentaire :

$$000111 \longrightarrow 00\mathbf{1}111$$

- ▶ **Echange** : pour une solution codé en caractère (ou binaire), permutez deux caractères données :

$$ABCDE \longrightarrow AECDB$$

- ▶ **Insertion-décalage** : pour une chaîne de caractères (ou binaires), choisir deux positions i et j , mettre le caractère de la position j en position i et décaler les caractères à droite

Méthodes de recherche locales : Principe



On cite quelque transformations élémentaires :

- ▶ **Complément** : pour une solution codé en binaire la transformation élémentaire est de remplacer un bit quelconque par son complémentaire :

$$000111 \longrightarrow 00\mathbf{1}111$$

- ▶ **Echange** : pour une solution codé en caractère (ou binaire), permutez deux caractères données :

$$ABCDE \longrightarrow AECDB$$

- ▶ **Insertion-décalage** : pour une chaîne de caractères (ou binaires), choisir deux positions i et j , mettre le caractère de la position j en position i et décaler les caractères à droite

$$i=2, j=5 : \quad ABCDEF\mathbf{G} \longrightarrow AFBCDEG$$

Méthodes de recherche locales : Méthode de Descente



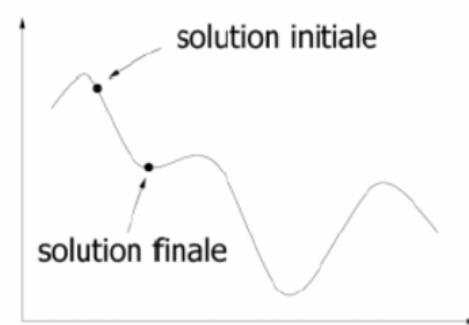
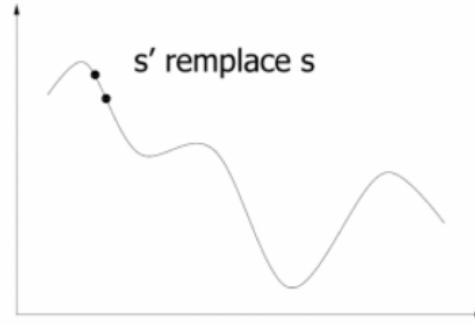
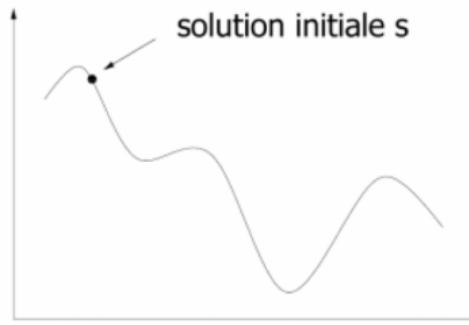
Principe

La Méthode descente (recherche locale) est probablement la méthode métaheuristique la plus ancienne et la plus simple. On démarre avec une solution initiale donnée. À chaque itération, l'heuristique remplace la solution actuelle par un voisin qui améliore la fonction objectif. La recherche s'arrête lorsque tous les voisins candidats sont pires que la solution actuelle, ce qui signifie qu'un optimum local est atteint.

Méthodes de recherche locales : Méthode de Descente

Principe

La Méthode descente (recherche locale) est probablement la méthode métahéuristique la plus ancienne et la plus simple. On démarre avec une solution initiale donnée. À chaque itération, l'heuristique remplace la solution actuelle par un voisin qui améliore la fonction objectif. La recherche s'arrête lorsque tous les voisins candidats sont pires que la solution actuelle, ce qui signifie qu'un optimum local est atteint.



Méthodes de recherche locales : Méthode de Descente



Algorithme 2 : Méthode de descente

Result : Solution S

```
 $S = S_0$  /* Solution initiale générée */  
While (condition d'arrêt n'est pas vérifié ) do  
    /* Génération d'un ensemble de solutions à partir du voisinage de la solution  $S_t$  */  
    Generate ( $C(S_t)$ )  
    /* Sélectionner  $S'$  qui minimise la fonction objectif dans le voisinage  $C(S_t)$  */  
    If  $f(S) < f(S')$  then  
        Condition d'arrêt est vérifiée  
    End if  
     $S = S'$   
End while  
Return  $S$ 
```

Méthodes de recherche locales : Méthode de Descente

Algorithme 2 : Méthode de descente

Result : Solution S

```
 $S = S_0$  /* Solution initiale générée */
While (condition d'arrêt n'est pas vérifié ) do
    /* Génération d'un ensemble de solutions à partir du voisinage de la solution  $S_t$  */
    Generate ( $C(S_t)$ )
    /* Sélectionner  $S'$  qui minimise la fonction objectif dans le voisinage  $C(S_t)$  */
    If    $f(S) < f(S')$    then
        Condition d'arrêt est vérifiée
    End if
     $S = S'$ 
End while
Return  $S$ 
```

Les inconvénients de la méthode :

- ▶ L'arrêt de l'algorithme au premier minimum local rencontré.

Méthodes de recherche locales : Méthode de Descente

Algorithme 2 : Méthode de descente

Result : Solution S

```
 $S = S_0$  /* Solution initiale générée */  
While (condition d'arrêt n'est pas vérifié ) do  
    /* Génération d'un ensemble de solutions à partir du voisinage de la solution  $S_t$  */  
    Generate ( $C(S_t)$ )  
    /* Sélectionner  $S'$  qui minimise la fonction objectif dans le voisinage  $C(S_t)$  */  
    If  $f(S) < f(S')$  then  
        Condition d'arrêt est vérifiée  
    End if  
     $S = S'$   
End while  
Return  $S$ 
```

Les inconvénients de la méthode :

- ▶ L'arrêt de l'algorithme au premier minimum local rencontré.
- ▶ On ne sait pas si on obtient une solution optimale et on ne sait pas quand

Méthodes de recherche locales : Méthode de Descente



On distingue différents types de descente :

- ▶ La descente déterministe,
- ▶ La descente stochastique,
- ▶ La descente vers le premier meilleur.

Les différentes méthodes de descente correspondent à des choix différents pour les directions de descente et les pas de déplacement.

Méthodes de recherche locales : Méthode de Descente



On distingue différents types de descente :

- ▶ La descente déterministe,
- ▶ La descente stochastique,
- ▶ La descente vers le premier meilleur.

Les différentes méthodes de descente correspondent à des choix différents pour les directions de descente et les pas de déplacement.

On commence par la méthode la plus utilisée :

Méthodes de recherche locales : Méthode de Descente



On distingue différents types de descente :

- ▶ La descente déterministe,
- ▶ La descente stochastique,
- ▶ La descente vers le premier meilleur.

Les différentes méthodes de descente correspondent à des choix différents pour les directions de descente et les pas de déplacement.

On commence par la méthode la plus utilisée :

Méthode de descente de Gradient :

Pour déterminer la solution voisine, la méthode de descente de gradient utilise :

Méthodes de recherche locales : Méthode de Descente



On distingue différents types de descente :

- ▶ La descente déterministe,
- ▶ La descente stochastique,
- ▶ La descente vers le premier meilleur.

Les différentes méthodes de descente correspondent à des choix différents pour les directions de descente et les pas de déplacement.

On commence par la méthode la plus utilisée :

Méthode de descente de Gradient :

Pour déterminer la solution voisine, la méthode de descente de gradient utilise :

- Le gradient pour calculer la direction
- Un pas de déplacement pour se déplacer vers la solution voisine.

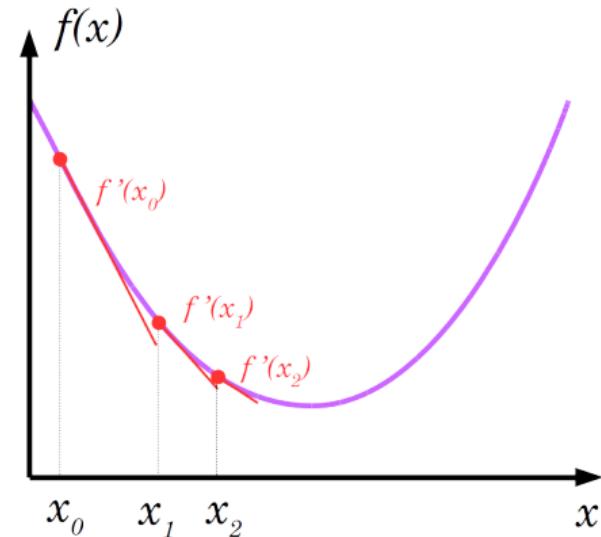
Méthodes de recherche locales : Méthode de Descente

On utilise comme direction : le gradient de la fonction objectif f :

$$x \in \mathbb{R} \quad d_k = \nabla f(x) = f'(x)$$

Et comme pas de déplacement α :

$$x_{k+1} = x_k - \alpha \nabla f(x_k)$$



Méthodes de recherche locales : Méthode Recuit simulé



Principe

Le recuit simulé (ou la méthode de Monte-Carlo) provient de la physique moléculaire où des molécules se positionnent de façon à minimiser leur énergie quand la température baisse.

L'algorithme simule les changements d'énergie dans un système soumis à un processus de refroidissement jusqu'à ce qu'il converge vers un état d'équilibre.

Méthodes de recherche locales : Méthode Recuit simulé



Principe

Le recuit simulé (ou la méthode de Monte-Carlo) provient de la physique moléculaire où des molécules se positionnent de façon à minimiser leur énergie quand la température baisse.

L'algorithme simule les changements d'énergie dans un système soumis à un processus de refroidissement jusqu'à ce qu'il converge vers un état d'équilibre.

L'analogie entre le système physique et le problème d'optimisation est la suivante :

Système physique	Problème d'optimisation
Énergie	Fonction objectif
État de système	Solution
États de basses énergie	Bonne solution
Température	Paramètre de contrôle

Méthodes de recherche locales : Méthode Recuit simulé



Remarque :

Recuit simulé est un algorithme stochastique qui permet dans certaines conditions la dégradation d'une solution. L'objectif est d'échapper aux optima locaux et ainsi de retarder la convergence.

Méthodes de recherche locales : Méthode Recuit simulé



Algorithme 3 : Méthode Recuit Simulé

Result : Solution S

$S = S_0$ /* Solution initiale générée */

$T = T_{Max}$ /* Température initiale */

While (condition d'arrêt n'est pas vérifié) **do**
repeat

Générer S' solution aléatoire dans le voisinage de S ;

Générer un nombre réel aléatoire r entre 0 et 1 ;

If $r < P(s, s', T)$ **then**

$S = S'$

Condition d'équilibre vérifiée

End if

Until Condition d'équilibre non vérifiée

/* On fixe un nombre maximum d'itération pour le choix de la nouvelle solution */

Mettre à jour la température T

End while

Return Meilleur solution trouvée

Méthodes de recherche locales : Méthode Recuit simulé



Algorithme de Metropolis

On part d'une configuration donnée, et on lui fait subir une modification aléatoire. Si cette modification fait diminuer la fonction objectif (énergie du système), elle est directement acceptée.

Sinon, elle n'est acceptée qu'avec une probabilité égale à $\exp(-\frac{\Delta E}{T})$ (où E est l'énergie et T est la température). **C'est le critère de Metropolis**

Méthodes de recherche locales : Méthode Recuit simulé

Algorithme de Metropolis

On part d'une configuration donnée, et on lui fait subir une modification aléatoire. Si cette modification fait diminuer la fonction objectif (énergie du système), elle est directement acceptée.

Sinon, elle n'est acceptée qu'avec une probabilité égale à $\exp(-\frac{\Delta E}{T})$ (où E est l'énergie et T est la température). **C'est le critère de Metropolis**

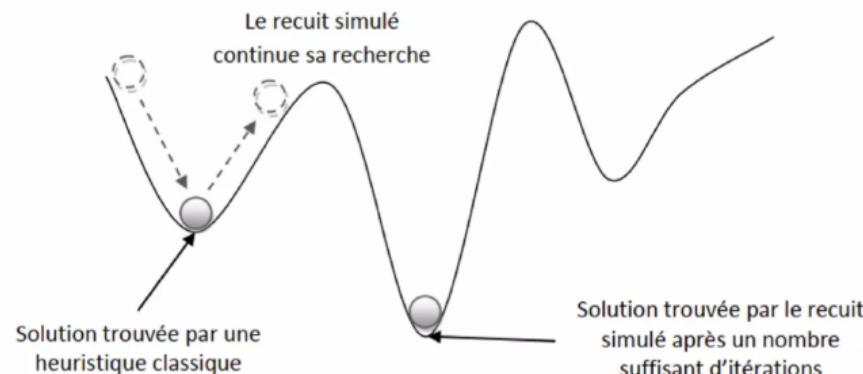
Principe

Dans un premier temps, T_{max} étant généralement choisi très grand, beaucoup de solutions sont acceptées, et l'algorithme équivaut à une visite aléatoire de l'espace des solutions. Mais à mesure que la température baisse, la plupart des solutions augmentant l'énergie sont refusées, et l'algorithme se ramène à une amélioration itérative classique. A température intermédiaire, l'algorithme autorise de fois des transformations qui dégradent la fonction objectif. Il laisse ainsi une chance au système de s'extraire d'un minimum local.

Méthodes de recherche locales : Méthode Recuit simulé

Remarque :

Notons que si la température est égale à 0, seules les solutions optimisant f sont acceptées. L'algorithme se comportera donc comme la méthode de la descente du gradient.



Méthodes de recherche locales : Méthode Recuit simulé



Fonctionnement

À partir d'une solution initiale, Recuit simulé procède en plusieurs itérations. A chaque itération, un voisin aléatoire est généré. Les mouvements qui améliorent la fonction de coût sont toujours acceptés. Sinon, le voisin est sélectionné avec une probabilité donnée qui dépend de la température actuelle et du degré de dégradation ΔE de la fonction objectif. ΔE représente la différence de valeur objective (énergie) entre la solution actuelle et la solution voisine générée. Au fur et à mesure que l'algorithme progresse, la probabilité que ces mouvements soient acceptés diminue.

Méthodes de recherche locales : Méthode Recuit simulé



Fonction Boltzmann

La probabilité en général de la distribution de Boltzmann :

$$P(S, S', T) = \exp\left(-\frac{f(S') - f(S)}{T}\right) = \exp\left(-\frac{\Delta E}{T}\right)$$

avec S est la solution courante et S' est son voisin choisis aléatoirement.

Méthodes de recherche locales : Méthode Recuit simulé



Fonction Boltzmann

La probabilité en général de la distribution de Boltzmann :

$$P(S, S', T) = \exp\left(-\frac{f(S') - f(S)}{T}\right) = \exp\left(-\frac{\Delta E}{T}\right)$$

avec S est la solution courante et S' est son voisin choisis aléatoirement.

Fonction Boltzmann

La fonction de Boltzmann fonctionne de la manière suivante :

- Si $f(S) > f(S')$ (La solution est amélioré), alors $P(S, S', T) > 1$ la condition d'acceptation est vérifiée.
- Dans le cas contraire, nous avons deux cas :
 - Si T a une très grande valeur, alors $P(S, S', T) \cong 1$, et on est donc presque sûr d'accepter s
 - Si T a une très petite valeur alors $P(s, s', T) \cong 0$, on va donc probablement refuser s

Méthodes de recherche locales : Méthode Recuit simulé



Les inconvénients de la méthodes :

- ▶ Le choix aléatoire dans le voisinage. On peut donc être proche de l'optimum et passer juste à côté sans le voir.
- On peut corriger cet inconvénient en ajoutant une descente.

Méthodes de recherche locales : Méthode Recuit simulé

Les inconvénients de la méthodes :

- ▶ Le choix aléatoire dans le voisinage. On peut donc être proche de l'optimum et passer juste à côté sans le voir.
- On peut corriger cet inconvénient en ajoutant une descente.

Mise à jour de T

- ▶ Plus T est petite et moins on a de chance d'accepter un mouvement qui dégrade f .
- ▶ Au début, on choisit T grand pour une plus grande liberté d'exploration.
- ▶ On fait ensuite tendre T vers 0 ce qui empêche de détériorer une solution.
- ▶ Décroître la température est donc une stratégie d'intensification.
- ▶ Convergence : Si on définit T_k tel que

$$\sum_{k=1}^{\infty} e^{\frac{L}{T_k}} = \infty \text{ avec } L \in \mathbb{R}$$

alors, $\lim_{k \rightarrow \infty} P[\text{ optimum trouvé après } k \text{ itérations}] = 1$

Méthodes de recherche locales : Méthode Recuit simulé



Mise à jour de T (suite)

- ▶ Fonctionne avec $g(T) = \frac{L}{\log(T + c)}$ avec $c \in \mathbb{R}$, mais implique une convergence pratique trop lente.
- ▶ On sacrifie donc la convergence théorique pour une convergence pratique plus rapide vers des optima locaux.
- ▶ Choix populaire : $g(T) = \alpha T$ avec $0 < \alpha < 1$ (typiquement $\alpha = 0.95$).
- ▶ On peut aussi décroître T par paliers.
- ▶ On peut aussi utiliser une fonction non-monotone : réhausser la température est alors une stratégie de diversification

Méthodes de recherche locales : Méthode Recuit simulé



Améliorations

- ▶ Méthode sans mémoire, mais on peut ajouter une mémoire à long terme qui stock le meilleur point rencontré
- ▶ Comme critère d'arrêt, on peut choisir une limite sur le temps, ou une limite sur le nombre d'itérations sans modification du meilleur point courant. Lorsqu'on stock le meilleur point rencontré, on peut décider de stopper la recherche dès qu'un certain nombre d'itérations a été effectué sans amélioration de ce point.

Méthodes de recherche locales : Méthode Recuit simulé

Algorithme 3 : Recuit Simulé

Result : Solution S

Initialisation :

```
 $S = S_0$           /* Solution initiale */  
 $T = T_0$           /* Température de départ */  
 $fit = f(S)$ 
```

While (critère d'arrêt non vérifié) **do**

Repeat

 Générer aléatoirement S' dans le voisinage de S

```
 $\Delta = f(S') - fit$   
If    $\Delta \leq 0$  then  
         $S = S'$   
         $fit = f(S)$ 
```

Else

 Générer un nombre aléatoire r entre 0 et 1

```
        If  $r \leq e^{-\frac{\Delta}{T}}$  then  
             $S = S'$   
             $fit = f(S)$   
        End if
```

End if

Until condition d'équilibre soit vérifiée

Mettre à jour la température T

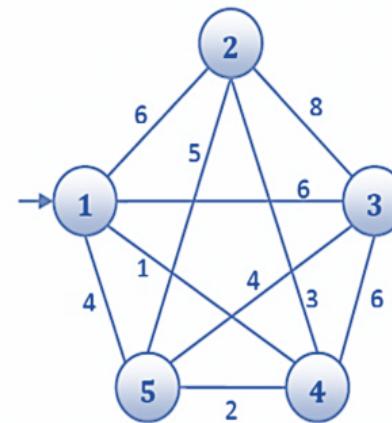
End while

Return : Meilleur solution trouvée

Méthodes de recherche locales : Méthode Recuit simulé

Exemple : Problème du voyageur de commerce :

L'algorithme va donc tenter de minimiser la longueur totale du chemin, en modifiant l'ordre des villes à parcourir. Soit le graphe suivant qui représente l'ensemble des villes parcourus :



L'énergie représentera la distance totale à parcourir, et un état du système représentera le chemin entre les villes.

Méthodes de recherche locales : Méthode Recuit simulé



Figure : Solution initiale (Parcours suivant l'ordre des villes)

$$f(S_0) = 26$$

Méthodes de recherche locales : Méthode Recuit simulé



Figure : Solution initiale (Parcours suivant l'ordre des villes)

$$f(S_0) = 26$$

On se déplace d'un sommet vers son plus proche voisin



Figure : Le résultat donné par l'algorithme glouton

$$f(S_0) = 21$$

Méthodes de recherche locales : Méthode Recuit simulé



Figure : Le résultat obtenu en échangeant les sommets 2 et 3

$$f(S) = 22$$

Le résultat obtenu en échangeant les sommets 2 et 3, la distance totale a augmenté. Pour une heuristique classique cette solution est rejetée car la distance doit être minimisé, mais le recuit simulé peut l'accepter si la température est encore élevée, et cette solution qui est **mauvaise** par rapport à la première va lui permettre de trouver une meilleure solution :

Méthodes de recherche locales : Méthode Recuit simulé



Figure : Le résultat obtenu en échangeant les sommets 5 et 2

$$f(S) = 18$$

Méthodes de recherche locales : Méthode Recuit simulé



Figure : Le résultat obtenu en échangeant les sommets 5 et 2

$$f(S) = 18$$

Résumé :

Le recuit simulé en acceptant une mauvaise solution, a réussi à échapper au minimum local et a obtenu une meilleure solution.

Méthodes de recherche locales : Méthode Tabou



Présentation

- ▶ La méthode Tabou est proposée par Glover en 1986, il a introduit la notion de mémoire dans la stratégie d'exploration.
- ▶ Dans les années 1990, l'algorithme de recherche tabou est devenu très populaire pour résoudre les problèmes d'optimisation de manière approximative.
- ▶ L'utilisation de la mémoire, qui stock des informations relatives au processus de recherche, représente la caractéristique particulière de la recherche tabou.

Méthodes de recherche locales : Méthode Tabou



Principe

- ▶ la méthode Tabou se comporte comme un algorithme de recherche descente, mais il accepte les solutions non améliorantes pour échapper à l'optimum local lorsque tous les voisins sont des solutions non améliorantes.
- ▶ La méthode utilise le principe de mémoire pour éviter les retours en arrière (mouvements cycliques)
- ▶ Pour éviter les cycles, elle élimine les voisins précédemment visités. Il mémorise la trajectoire de recherche récente.

La recherche par tabou gère une mémoire des solutions ou des mouvements récemment appliqués, appelée liste tabou. Cette liste de tabous constitue la mémoire à court terme. A chaque itération, la mémoire à court terme est mise à jour

Méthodes de recherche locales : Méthode Tabou



Principe

- ▶ la méthode Tabou se comporte comme un algorithme de recherche descente, mais il accepte les solutions non améliorantes pour échapper à l'optimum local lorsque tous les voisins sont des solutions non améliorantes.
- ▶ La méthode utilise le principe de mémoire pour éviter les retours en arrière (mouvements cycliques)
- ▶ Pour éviter les cycles, elle élimine les voisins précédemment visités. Il mémorise la trajectoire de recherche récente.

La recherche par tabou gère une mémoire des solutions ou des mouvements récemment appliqués, appelée liste tabou. Cette liste de tabous constitue la mémoire à court terme. A chaque itération, la mémoire à court terme est mise à jour

Remarque :

Le stockage de toutes les solutions visitées prend du temps et de l'espace. En effet, nous devons vérifier à chaque itération si une solution générée ne fait pas partie de la liste de toutes les solutions visitées. La liste des tabous contient généralement un nombre constant de déplacements tabous.

Méthodes de recherche locales : Méthode Tabou

Algorithme 4 : Méthode Tabou

Result : Solution S

Initialisation de la liste Tabou T

$S = S_0$ /* Solution initiale générée */

While (condition d'arrêt n'est pas vérifié) **do**

/* Génération d'un ensemble de solution à partir du voisinage de la solution S et qui sont différent de la liste Tabou */

Generate $C(S_t)$

Sélectionner S' qui minimise la fonction objectif dans le voisinage $C(S_t)$

$S = S'$

Mettre à jour la liste Tabou T

End while

Return Meilleur solution trouvée

Méthodes de recherche locales : Méthode Tabou

Fable des randonneurs :

Un randonneur malchanceux est perdu dans une région montagneuse. Toutefois, il sait qu'une équipe de secours passe régulièrement par le point situé à la plus basse altitude dans la région. Ainsi, il doit se rendre à ce point pour attendre les secours. Comment s'y prendra-t-il ?

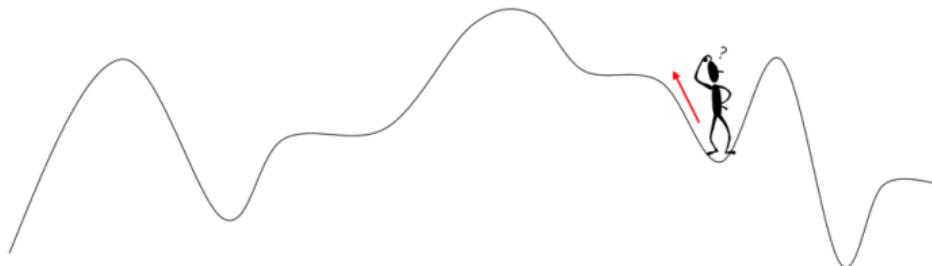
- ▶ Il ne connaît pas l'altitude de ce point,
- ▶ Il ne voit pas autour de lui à cause du brouillard,
- ▶ arrivé à un croisement il doit s'engager dans une direction pour voir si le chemin monte ou descend.

Il commence par descendre tant qu'il peut, en choisissant chemin de plus grande pente à chaque croisement.



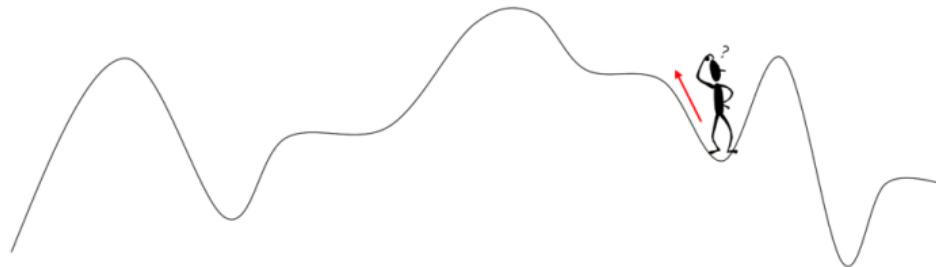
Méthodes de recherche locales : Méthode Tabou

Puis, lorsqu'il n'y a plus de sentier menant vers le bas, il décide de suivre le chemin qui remonte avec la plus faible pente car il est conscient qu'il peut se trouver à un minimum local.

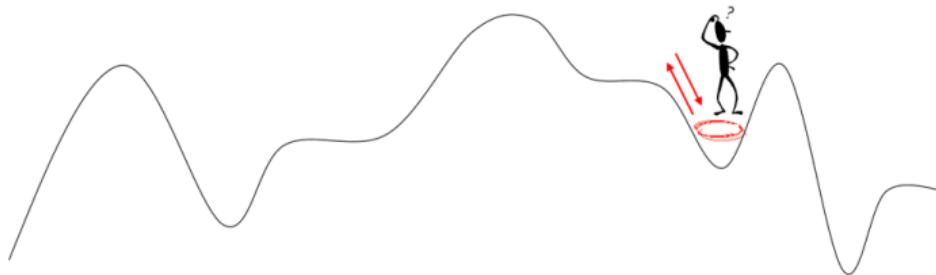


Méthodes de recherche locales : Méthode Tabou

Puis, lorsqu'il n'y a plus de sentier menant vers le bas, il décide de suivre le chemin qui remonte avec la plus faible pente car il est conscient qu'il peut se trouver à un minimum local.

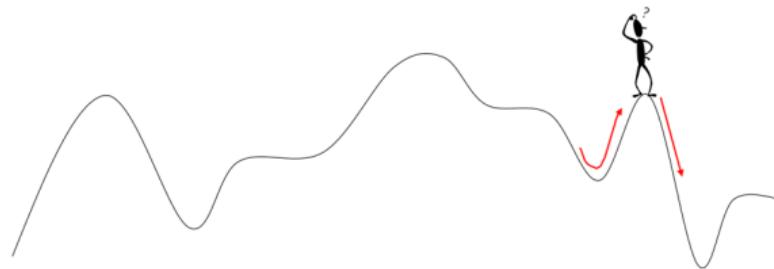


Toutefois, dès qu'il remonte, il redescend vers le point où il était. Par conséquent, il décide de s'interdire de faire marche arrière en mémorisant la direction d'où il vient.



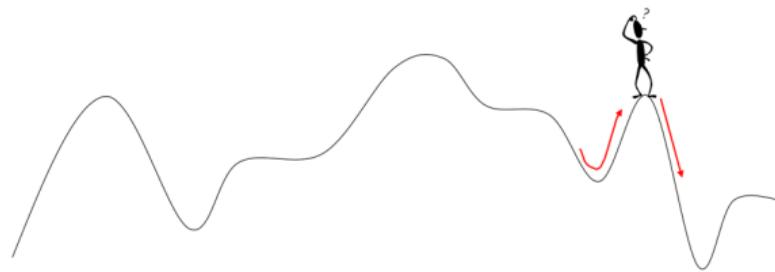
Méthodes de recherche locales : Méthode Tabou

Cette stratégie lui permet d'explorer des minimum locaux et d'en ressortir. À un moment donné, il arrive à un point où il décèle une forte pente descendante vers le sud. Alors, que les directions mémorisées lui interdisent d'aller vers le sud (direction prohibée). Il décide d'ignorer cette interdiction et emprunte ce chemin.

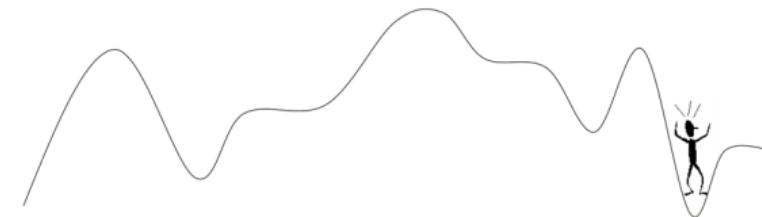


Méthodes de recherche locales : Méthode Tabou

Cette stratégie lui permet d'explorer des minimum locaux et d'en ressortir. À un moment donné, il arrive à un point où il décèle une forte pente descendante vers le sud. Alors, que les directions mémorisées lui interdisent d'aller vers le sud (direction prohibée). Il décide d'ignorer cette interdiction et emprunte ce chemin.



Cette décision fut bénéfique: il arriva au point de plus basse altitude et attendit les secours qui ne tardèrent à arriver.



Méthodes de recherche locales : Méthode Tabou

Améliorations :

La recherche de la solution optimale peut être améliorée,

- ▶ Choix stratégique de la solution initiale. Ceci donnera une bonne valeur de f .
- ▶ **Intensifier** la recherche dans les voisinages des solutions qui semblent propices à mener à des solutions proches ou égales à l'optimum
- ▶ **diversifier** la recherche en éloignant celle-ci des voisinages peu propices à produire de bonne solutions.

Méthodes de recherche locales : Méthode Tabou

Améliorations :

La recherche de la solution optimale peut être améliorée,

- ▶ Choix stratégique de la solution initiale. Ceci donnera une bonne valeur de f .
- ▶ **Intensifier** la recherche dans les voisinages des solutions qui semblent propices à mener à des solutions proches ou égales à l'optimum
- ▶ **Diversifier** la recherche en éloignant celle-ci des voisinages peu propices à produire de bonne solutions.

Modifications à la fonction objectif :

L'intensification et la diversification sont présentées comme des modifications à la fonction objectif. Elles se manifestent pendant quelques itérations seulement.

Méthodes de recherche locales : Méthode Tabou

Améliorations :

La recherche de la solution optimale peut être améliorée,

- ▶ Choix stratégique de la solution initiale. Ceci donnera une bonne valeur de f .
- ▶ **Intensifier** la recherche dans les voisinages des solutions qui semblent propices à mener à des solutions proches ou égales à l'optimum
- ▶ **Diversifier** la recherche en éloignant celle-ci des voisinages peu propices à produire de bonne solutions.

Modifications à la fonction objectif :

L'intensification et la diversification sont présentées comme des modifications à la fonction objectif. Elles se manifestent pendant quelques itérations seulement.

Pour l'intensification, une pénalité est attribuée à des solutions éloignées de l'actuelle. Ceci cause un gonflement de la fonction objectif : les solutions semblables seront donc privilégiées.

Méthodes de recherche locales : Méthode Tabou

Améliorations :

La recherche de la solution optimale peut être améliorée,

- ▶ Choix stratégique de la solution initiale. Ceci donnera une bonne valeur de f .
- ▶ **Intensifier** la recherche dans les voisinages des solutions qui semblent propices à mener à des solutions proches ou égales à l'optimum
- ▶ **Diversifier** la recherche en éloignant celle-ci des voisinages peu propices à produire de bonne solutions.

Modifications à la fonction objectif :

L'intensification et la diversification sont présentées comme des modifications à la fonction objectif. Elles se manifestent pendant quelques itérations seulement.

Pour l'intensification, une pénalité est attribuée à des solutions éloignées de l'actuelle. Ceci cause un gonflement de la fonction objectif : les solutions semblables seront donc privilégiées.

Pour la diversification, l'effet est le contraire. Les solutions proches l'actuelle sont pénalisées.

Donc,

$$\tilde{f} = f + \text{intensification} + \text{diversification}$$

Méthodes d'évolution

Introduction :

Les méthodes d'évolution sont basées sur une population de solution, on commence par une population de solution pour améliorer la solution globale.

Ces méthodes améliorent au fur et à mesure des itérations, une population de solution. Ces algorithmes sont très flexibles et ont la capacité de traiter des problèmes avec des fonctions objectif de différentes propriétés, qu'elles soient continues, discrètes ou mixtes.

Ils font évoluer une population d'individus selon des règles bien précises. Ces méthodes alternent entre les périodes d'adaptation individuelle et des périodes de coopération durant lesquelles les individus peuvent échanger de l'information.

Méthodes d'évolution

Principe :

- ▶ Utiliser un ensemble de solution au lieu d'une seule
- ▶ Algorithme itératif qui change la population à chaque itération
- ▶ Utilisation des procédures de sélection, génération et évaluation des individus
- ▶ La plupart de ses méthodes sont inspirées de la nature, (algorithmes génétiques, colonie de fourmis, recherche par dispersion, essaims de particules,...)

Méthodes d'évolution

Algorithme 5 : Meta heuristique à Population

Initialisation :

$P = P_0$ /* Génération de la population initiale */

$t = 0$

While (critère d'arrêt non vérifié) **do**

 /* Génération d'une nouvelle population */

 Generate (P'_t)

$P_{t+1} = SelectPopulation (P_t \cup P'_t)$

$t = t + 1$

End while

Return : Meilleur solution trouvée

Méthodes d'évolution : Algorithmes génétiques



Principe

Les algorithmes génétiques explorent un espace de solutions en utilisant une analogie avec les mécanismes évolutifs naturels, en combinant les concepts de sélection naturelle, de reproduction et de mutation pour trouver des solutions potentielles à des problèmes d'optimisation ou de recherche. Voici les principes de base :

- ▶ **Initialisation de la population** : Un ensemble de solutions potentielles (appelées individus ou chromosomes) est généré aléatoirement pour constituer une population initiale.
- ▶ **Évaluation** : Chaque individu de la population est évalué en fonction de son aptitude (fitness).
- ▶ **Sélection** : Les individus de la population sont sélectionnés pour la reproduction, en fonction de leur aptitude

Méthodes d'évolution : Algorithmes génétiques



Principe (suite)

- ▶ **Reproduction** : Les individus sélectionnés sont combinés pour créer une nouvelle génération d'individus
- ▶ **Remplacement** : La nouvelle génération remplace l'ancienne, et le processus d'évaluation, de sélection, de reproduction et de remplacement se répète sur cette nouvelle population.
- ▶ **Critère d'arrêt** : Le processus continue jusqu'à ce qu'un critère d'arrêt soit satisfait, tel qu'un nombre maximum d'itérations atteint, une solution satisfaisante trouvée, ou une stagnation de la fitness.

Méthodes d'évolution : Algorithmes génétiques



Fonctionnement :

- Un ensemble de solutions potentielles, représentées sous forme de d'individus, est généré aléatoirement pour former une population initiale.
- Chaque individus est une représentation possible d'une solution au problème donné.
- Chaque individu de la population est évalué en fonction de son aptitude à résoudre le problème.
- Une fonction d'évaluation (fonction objectif) est utilisée pour attribuer une valeur numérique à chaque individu, indiquant à quel point il est performant dans la résolution du problème.
- Les individus de la population sont sélectionnés pour la reproduction en fonction de leur aptitude.
- Les individus les mieux adaptés, ont plus de chances d'être sélectionnés.
- Différentes techniques de sélection peuvent être utilisées, telles que la sélection par roulette, la sélection par tournoi, etc.

Méthodes d'évolution : Algorithmes génétiques



Fonctionnement (suite) :

- Les individus sélectionnés sont utilisés pour créer une nouvelle génération d'individus.
- Cela se fait généralement en utilisant des opérateurs génétiques tels que le croisement et la mutation.
- Le croisement implique le mélange des informations génétiques de deux parents pour créer des descendants.
- La mutation consiste en des modifications aléatoires apportées aux chromosomes pour introduire de la diversité génétique dans la population.
- La nouvelle génération remplace l'ancienne dans la population.
- Les individus moins performants sont éliminés, tandis que les descendants des individus les plus performants sont ajoutés à la population.

Méthodes d'évolution : Algorithmes génétiques

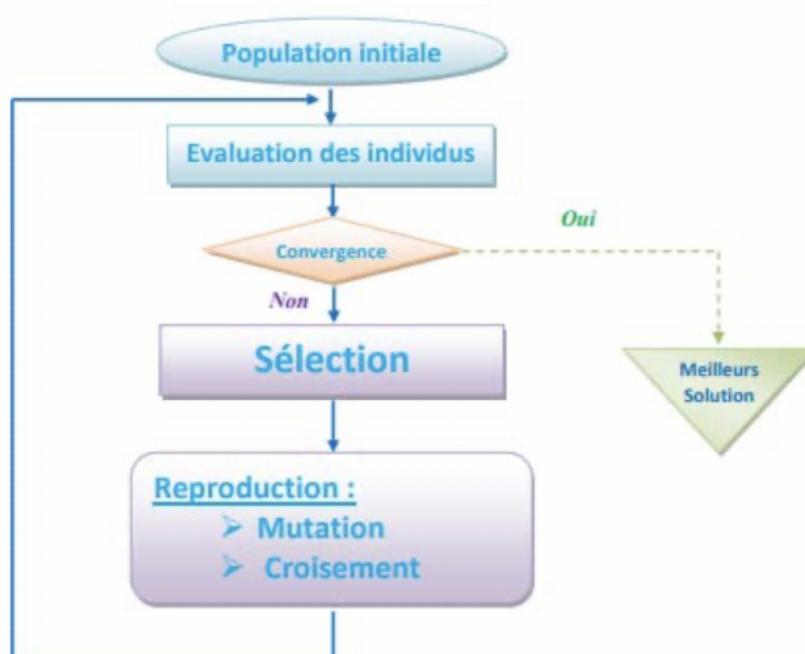


Figure : Cycle de l'algorithme génétique

Méthodes d'évolution : Algorithmes génétiques



Algorithme 6 : Algorithme Génétique

Initialisation :

$P = P_0$ /* Génération de la population initiale */

$t = 0$

While (critère d'arrêt non vérifié) **do**

/* Sélection de la population pour le croisement */

$P_t^1 = SelectCroisement (P_t)$

/* Génération de la population (des enfants) par croisement */

$P_t^C = GenerateCroisement (P_t^1)$

/* Sélection de la population pour la mutation */

$P_t^2 = SelectMutation (P_t)$

/* Génération de la population (des enfants) par mutation */

$P_t^m = GenerateMutation (P_t^2)$

$P_{t+1} = SelectPopulation (P_t \cup P_t^C \cup P_t^m)$

$t = t + 1$

End while

Return : Meilleur solution trouvée durant toute les itérations

Méthodes d'évolution : Algorithmes génétiques



Codage

Avant de générer l'ensemble des individus représentant la population initiale, il faut d'abord penser à définir le codage convenable des individus de la population. Le choix du type de codage dépend du problème à résoudre et de la manière dont les solutions sont le mieux représentées. Il est important de sélectionner un type de codage qui permet une représentation adéquate des solutions et qui facilite l'application des opérateurs génétiques pour générer de nouvelles solutions.

Méthodes d'évolution : Algorithmes génétiques



Codage

Avant de générer l'ensemble des individus représentant la population initiale, il faut d'abord penser à définir le codage convenable des individus de la population. Le choix du type de codage dépend du problème à résoudre et de la manière dont les solutions sont le mieux représentées. Il est important de sélectionner un type de codage qui permet une représentation adéquate des solutions et qui facilite l'application des opérateurs génétiques pour générer de nouvelles solutions.

Codage Binaire

Codage binaire : Le codage binaire consiste à utiliser des bits (0 ou 1) pour représenter les différentes solutions. Le type de ce codage s'adapte bien aux problèmes de type binaire, comme dans les problèmes d'optimisation combinatoire.

Méthodes d'évolution : Algorithmes génétiques



Codage entier

Les solutions sont représentées sous forme de nombres entiers. Utilisé lorsque les solutions peuvent être mieux décrites par des nombres entiers plutôt que par des bits. Ce type de codage est utile pour les problèmes où les variables sont discrètes.

Méthodes d'évolution : Algorithmes génétiques



Codage entier

Les solutions sont représentées sous forme de nombres entiers. Utilisé lorsque les solutions peuvent être mieux décrites par des nombres entiers plutôt que par des bits. Ce type de codage est utile pour les problèmes où les variables sont discrètes.

Codage réel

Ce type de codage est le plus efficace pour représenter des problèmes de type continu, tels que les problèmes d'optimisation continue. Il représente les solutions par des suites de type réel. Il est souvent utilisé avec des opérateurs de mutation et de croisement spécifiques aux nombres réels.

Méthodes d'évolution : Algorithmes génétiques



Sélection :

La sélection est l'une des étapes fondamentales des algorithmes génétiques. Elle intervient dans deux étapes de chaque itération :

- Au début de chaque itération pour sélectionner les individus parents qui vont se reproduire entre eux.
- A la fin de chaque itération pour sélectionner les individus enfants qui vont remplacer les individus parents dans le but de créer une nouvelle population en respectant la taille de la population pour ne pas entraîner une explosion démographique de la population.

Méthodes d'évolution : Algorithmes génétiques



Types de sélection :

les individus de la population sont choisis pour la reproduction en fonction de leur aptitude ou de leur qualité. Il existe plusieurs techniques de sélection, chacune avec ses propres avantages et inconvénients. Voici quelques-unes des techniques de sélection les plus couramment utilisées :

- ▶ **Sélection par roulette** : Cette méthode est la plus connue et la plus utilisée. Avec cette méthode chaque individu a une chance d'être sélectionné proportionnelle à sa performance, donc plus les individus sont adaptés au problème, plus ils ont de chances d'être sélectionnés. Soit f_i la fonction fitness d'un individu p_i , alors la probabilité qu'il soit sélectionné est :

$$p_i = \frac{f_i}{\sum f_i}$$

Méthodes d'évolution : Algorithmes génétiques



- ▶ **Sélection par tournoi** : Cette méthode est simple et efficace, et elle est souvent utilisée pour maintenir une diversité dans la population. Des sous-groupes (ou tournois) aléatoires d'individus sont formés à partir de la population. Le meilleur individu de chaque tournoi est sélectionné pour la reproduction

Méthodes d'évolution : Algorithmes génétiques



- ▶ **Sélection par tournoi :** Cette méthode est simple et efficace, et elle est souvent utilisée pour maintenir une diversité dans la population. Des sous-groupes (ou tournois) aléatoires d'individus sont formés à partir de la population. Le meilleur individu de chaque tournoi est sélectionné pour la reproduction

Croisement :

Le croisement a pour but d'enrichir la diversité de la population en manipulant la structure des chromosomes. Classiquement, les croisements sont envisagés avec deux parents et génèrent deux enfants.

Méthodes d'évolution : Algorithmes génétiques



- ▶ **Sélection par tournoi :** Cette méthode est simple et efficace, et elle est souvent utilisée pour maintenir une diversité dans la population. Des sous-groupes (ou tournois) aléatoires d'individus sont formés à partir de la population. Le meilleur individu de chaque tournoi est sélectionné pour la reproduction

Croisement :

Le croisement a pour but d'enrichir la diversité de la population en manipulant la structure des chromosomes. Classiquement, les croisements sont envisagés avec deux parents et générant deux enfants.

Types de croisement :

Méthodes d'évolution : Algorithmes génétiques



- ▶ **Sélection par tournoi :** Cette méthode est simple et efficace, et elle est souvent utilisée pour maintenir une diversité dans la population. Des sous-groupes (ou tournois) aléatoires d'individus sont formés à partir de la population. Le meilleur individu de chaque tournoi est sélectionné pour la reproduction

Croisement :

Le croisement a pour but d'enrichir la diversité de la population en manipulant la structure des chromosomes. Classiquement, les croisements sont envisagés avec deux parents et génèrent deux enfants.

Types de croisement :

- ▶ **Croisement n-points :** Ce type de croisement consiste à choisir n-points de coupures ($n = 1, 2, \dots$), puis échanger les fragments de gènes délimités par les points de coupe choisis.

Méthodes d'évolution : Algorithmes génétiques



- ▶ **Sélection par tournoi :** Cette méthode est simple et efficace, et elle est souvent utilisée pour maintenir une diversité dans la population. Des sous-groupes (ou tournois) aléatoires d'individus sont formés à partir de la population. Le meilleur individu de chaque tournoi est sélectionné pour la reproduction

Croisement :

Le croisement a pour but d'enrichir la diversité de la population en manipulant la structure des chromosomes. Classiquement, les croisements sont envisagés avec deux parents et génèrent deux enfants.

Types de croisement :

- ▶ **Croisement n-points :** Ce type de croisement consiste à choisir n-points de coupures ($n = 1, 2, \dots$), puis échanger les fragments de gènes délimités par les points de coupe choisis.
- ▶ **Croisement uniforme :** Ce type de croisement est fondé sur la probabilité. En effet, il permet la génération d'un enfant en échangeant chaque gène des deux parents avec une probabilité égale à 0.5.

Méthodes d'évolution : Algorithmes génétiques

Exemples :

Croisement Barycentrique :

Deux éléments (binaires) issus de la génération k

10111010

11110000

On tire une position parmi les 8 bits : 4

Parent 1

Parent 2

10111010

11110000

10110000

11111010

Enfant 1

Enfant 2

Croisement : cas binaire

Deux éléments issus de la génération k : P1 et P2

$$C_1 = aP_1 + (1-a)P_2$$

$$C_2 = (1-a)P_1 + aP_2$$

P1

P2

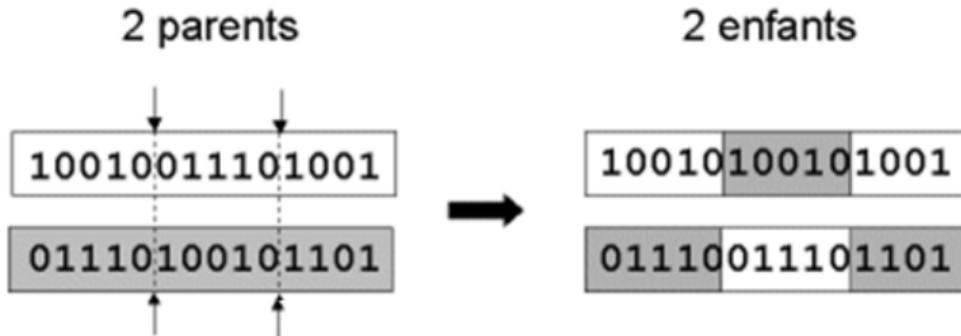
Barycentre de P1 et P2 avec a dans $[0,1]$

Barycentre de P1 et P2 avec a hors de $[0,1]$

Croisement : cas réel

Méthodes d'évolution : Algorithmes génétiques

Croisement deux points :



Méthodes d'évolution : Algorithmes génétiques



Mutation :

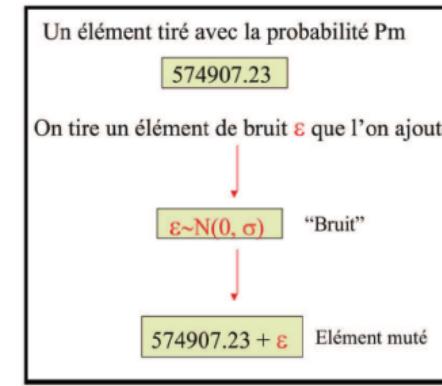
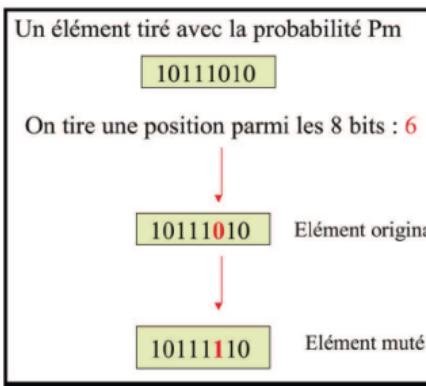
L'opérateur de mutation permet d'apporter des modifications partielles (légères) aux individus avec une certaine probabilité très faible. L'objectif de l'opérateur de mutation est d'aider l'algorithme à échapper au problème de stagnation de la recherche causé par les optima locaux. Il permet d'explorer l'espace de recherche en apportant à l'algorithme la possibilité de couvrir la totalité de l'espace de recherche.

Méthodes d'évolution : Algorithmes génétiques



Mutation :

L'opérateur de mutation permet d'apporter des modifications partielles (légères) aux individus avec une certaine probabilité très faible. L'objectif de l'opérateur de mutation est d'aider l'algorithme à échapper au problème de stagnation de la recherche causé par les optima locaux. Il permet d'explorer l'espace de recherche en apportant à l'algorithme la possibilité de couvrir la totalité de l'espace de recherche.



Méthodes d'évolution : Algorithmes génétiques



Exemple :

Voyageur de Commerce TSP

On considère une instance du PVC est un graphe complet de n sommets dont les arêtes sont pondérées par un coût strictement positif. L'instance sera alors implantée comme une matrice M de dimension $n \times n$ dont les coefficients sont strictement positifs sauf sur la diagonale où ils sont tous nuls. M est appelé matrice de coût. Ainsi la distance entre le sommet j et le sommet i est $M_{i,j}$. On n'a pas forcément $M_{i,j} = M_{j,i}$ et on n'a pas non plus l'inégalité triangulaire dans le cas général.

Méthodes d'évolution : Algorithmes génétiques



Codage : L'espace du recherche est l'ensemble S_n des permutations de $\{1, 2, \dots, n\}$. Un point de l'espace de recherche est une permutation.

Nous utiliserons le codage réel, chaque instance est un tableau de villes qui représente une permutation.

Méthodes d'évolution : Algorithmes génétiques

Codage : L'espace du recherche est l'ensemble S_n des permutations de $\{1, 2, \dots, n\}$. Un point de l'espace de recherche est une permutation.

Nous utiliserons le codage réel, chaque instance est un tableau de villes qui représente une permutation.

Fonction objectif :

$$f(\sigma) = \sum_{i=0}^{n-2} M_{\sigma(i), \sigma(i+1)} + M_{\sigma(n), \sigma(0)}$$

Le dernier terme de la somme permettant de revenir au sommet initial.

Méthodes d'évolution : Algorithmes génétiques



Codage : L'espace du recherche est l'ensemble S_n des permutations de $\{1, 2, \dots, n\}$. Un point de l'espace de recherche est une permutation.

Nous utiliserons le codage réel, chaque instance est un tableau de villes qui représente une permutation.

Fonction objectif :

$$f(\sigma) = \sum_{i=0}^{n-2} M_{\sigma(i), \sigma(i+1)} + M_{\sigma(n), \sigma(0)}$$

Le dernier terme de la somme permettant de revenir au sommet initial.

Sélection : Nous utilisons la méthode de sélection par roulette. On calcule d'abord la valeur moyenne de la fonction d'évaluation dans la population :

$$\bar{f} = \frac{1}{m} \sum_{i=0}^{m-1} f(P_i)$$

P_i est l'individu i de la population et m la taille de la population.

La place d'un individu P_i dans la roulette est proportionnel à $\frac{f}{\bar{f}(P_i)}$.

On sélectionne alors $\frac{m}{2}$ individus pour la reproduction.

Méthodes d'évolution : Algorithmes génétiques



Croisement : Une fois la population intermédiaire sélectionnée, on complète la population avec les enfants de la population intermédiaire. Deux chromosomes se combinant donnent naissance à deux autres chromosomes de la façon suivante :

- 1 Un point d'hybridation est déterminé aléatoirement (entre 0 et la longueur du chromosome).
- 2 On copie dans le premier fils les indices du premier père jusqu'au point d'hybridation.
- 3 On complète ensuite par les indices du deuxième chromosome père ne se trouvant pas déjà dans le fils dans l'ordre donné par le deuxième parent.
- 4 On réitère 2) et 3) avec le même point d'hybridation, mais en inversant le rôle des deux parents.

Méthodes d'évolution : Algorithmes génétiques



Croisement : Une fois la population intermédiaire sélectionnée, on complète la population avec les enfants de la population intermédiaire. Deux chromosomes se combinant donnent naissance à deux autres chromosomes de la façon suivante :

- 1 Un point d'hybridation est déterminé aléatoirement (entre 0 et la longueur du chromosome).
- 2 On copie dans le premier fils les indices du premier père jusqu'au point d'hybridation.
- 3 On complète ensuite par les indices du deuxième chromosome père ne se trouvant pas déjà dans le fils dans l'ordre donné par le deuxième parent.
- 4 On réitère 2) et 3) avec le même point d'hybridation, mais en inversant le rôle des deux parents.

Exemple :

Pour $n = 8$, on a les deux chromosomes suivants qui s'hybrident :

$$C_1 = (10532746), \quad C_2 = (71356204)$$

On hybride à partir du rang 5. On arrive alors aux deux nouveaux chromosomes :

$$C_3 = (10532764), \quad C_4 = (71356024).$$

Méthodes d'évolution : Algorithmes génétiques



Mutation : Une des méthodes les plus simples pour muter un chromosome est d'inverser les positions de deux villes. Par exemple :

$$C_5 = (01364752) \text{ devient } C_5 = (01764352).$$

Nouvelle fonction d'adaptation :

$$\begin{aligned}d_f = & d(i - 1, i) - d(i - 1, j) + d(j, i + 1) - d(i, i + 1) \\& + d(j - 1, i) - d(j - 1, j) + d(i, j + 1) - d(j, j + 1)\end{aligned}$$

Cependant, cette mutation change beaucoup la fonction d adaptation.

Méthodes d'évolution : Algorithmes génétiques



Mutation (suite) : Une solution qui perturbe moins l'individu consiste à inverser tous les sommets entre i et j . Par exemple :

$C_5 = (01364752)$ devient $C_5 = (01746352)$.

La différence de la fonction d'évaluation ne dépend plus que de quatre termes :

$$d_f = df = d(i-1, j) - d(i-1, i) + d(j+1, i) - d(j, j+1)$$

En fait, au lieu d'inverser deux sommets, cette méthode efface deux arêtes et les remplace par deux autres.