
LES ALGORITHMES GÉNÉTIQUES

Sommaire

3.1	Introduction	37
3.2	Les metaheuristiques	38
3.3	Optimisation à l'aide de l'algorithme génétique	40
3.4	Définition sur algorithmes génétiques	40
3.5	Fonctionnement général des algorithmes génétiques	42
3.6	Le principe de l'algorithme génétique	43
3.7	Le codage	44
3.8	Opérateurs de l'algorithme génétique	45
3.9	Avantages et Inconvénients des AGs	48
3.10	Conclusion	48

3.1 Introduction

L'apparition des "métaheuristiques" remonte aux années quatre-vingts. Ces algorithmes stochastiques d'optimisation globale peuvent être appliqués à tout problème, du moment qu'il est formulé sous la forme de l'optimisation de critère(s). Ces algorithmes sont inspirés par des analogies avec la physique (recuit simulé, recuit microcanonique), avec la biologie (algorithmes évolutionnaires les algorithmes génétique, L'intelligence en essaim) ou avec l'éthologie (colonies de fourmis, essaims particuliers).

3.2 Les métaheuristiques

Le mot métaheuristique est dérivé de la composition de deux mots grecs : méta, du grec *μετα* signifiant « au-delà » (ou « à un plus haut niveau ») et heuristique. En effet, ces algorithmes se veulent des méthodes génériques pouvant optimiser une large gamme de problèmes différents, sans nécessiter de changements profonds dans l'algorithme employé. Les métaheuristiques sont en général non-déterministes, elles peuvent ne pas trouver la solution optimale, et encore moins prouver l'optimalité de la solution trouvée. On peut distinguer les métaheuristiques qui font évoluer une seule solution sur l'espace de recherche à chaque itération et les métaheuristiques à base de population de solutions. En général, les métaheuristiques à base de solution unique sont plutôt axées sur l'exploitation de l'espace de recherche, on n'est donc jamais sûr d'obtenir *l'optimum*. Les métaheuristiques à base de population sont plutôt exploratoires et permettent une meilleure diversification de l'espace de recherche.

3.2.1 Les métaheuristiques à solution unique

Dans cette section, nous présentons les métaheuristiques à base de solution unique, aussi appelées méthodes de trajectoire. Contrairement aux métaheuristiques à base de population, les métaheuristiques à solution unique commencent avec une seule solution initiale et s'en éloignent progressivement, en construisant une trajectoire dans l'espace de recherche. Les méthodes de trajectoire englobent essentiellement la méthode de descente, la méthode du recuit simulé, la recherche taboue, la méthode GRASP, la recherche à voisinage variable, la recherche locale itérée, et leurs variantes.

- **Méthode de descente :** Un algorithme à directions de descente est un algorithme d'optimisation différentiable (l'optimisation dont il est question ici est une branche des mathématiques), destiné à minimiser une fonction réelle différentiable définie sur un espace euclidien ou, plus généralement, sur un espace hilbertien.
- **Le recuit simulé :** Le recuit simulé adapté aux problèmes continus a été utilisé pour résoudre le problème de segmentation par l'approche paramétrique. Le but est d'approcher l'histogramme à l'aide d'une combinaison de gaussiennes, dont il faut trouver les paramètres.

- **La méthode de recherche avec tabous :** La recherche taboue est une métaheuristique d'optimisation présentée par Fred Glover en 1986. On trouve souvent l'appellation recherche avec tabous en français. Cette méthode est une métaheuristique itérative qualifiée de recherche locale au sens large.
- **La recherche à voisinage variable : (RVV)** est une métaheuristique récente pour la résolution de problèmes d'optimisation combinatoire et globale, dont l'idée de base est le changement systématique de voisinage au sein d'une recherche locale.

3.2.2 Les métaheuristiques à population de solutions

Contrairement aux algorithmes partant d'une solution singulière, les métaheuristiques à population de solutions améliorent, au fur et à mesure des itérations, une population de solutions. On distingue dans cette catégorie, les algorithmes évolutionnaires, qui sont une famille d'algorithmes issus de la théorie de l'évolution par la sélection naturelle, énoncée par Charles Darwin en 1859, et les algorithmes d'intelligence en essaim qui, de la même manière que les algorithmes évolutionnaires, proviennent d'analogies avec des phénomènes biologiques naturels.

- **Les algorithmes génétiques :** Les algorithmes génétiques sont des algorithmes d'optimisation s'appuyant sur des techniques dérivées de la génétique et des mécanismes d'évolution de la nature : croisements, mutations, sélections, etc... Ils appartiennent à la classe des algorithmes évolutionnaires.
- **L'intelligence en essaim :** L'optimisation par essaim particulière a été appliquée à la segmentation d'images en 2005 grâce aux travaux de Du Feng. La métaheuristique a été utilisée pour maximiser l'entropie de Shannon à deux dimensions afin de segmenter des images infrarouges. L'algorithme OEP mis en œuvre est celui développé par Kennedy.
- **Les colonies de fourmis :** Les algorithmes de colonies de fourmis sont des algorithmes inspirés du comportement des fourmis, ou d'autres espèces formant un super organisme, et qui constituent une famille de métaheuristiques d'optimisation.

3.3 Optimisation à l'aide de l'algorithme génétique

Nous allons décrire brièvement les travaux d'application des métaheuristiques à population de solutions en choisira **Les algorithmes génétiques** au problème de la segmentation d'images.

3.4 Définition sur algorithmes génétiques

Les algorithmes génétiques appartiennent à la classe des algorithmes évolutionnaires. Ce sont des métaheuristiques inspirées de l'analogie entre un processus d'optimisation et l'évolution des êtres vivants. La simulation des mécanismes de variation et de sélection oeuvrant dans les processus évolutifs naturels est exploitée pour résoudre des problèmes artificiels d'optimisation [28].

Dans les algorithmes évolutionnaires, l'analogie avec l'optimisation consiste à considérer les solutions potentielles au problème comme des chromosomes. Ceux-ci sont manipulés par des opérateurs de *sélection* et *mutation*. Dans le cas des algorithmes génétiques, un troisième opérateur est utilisé, appelé opérateur de *croisement*. La qualité de la solution correspondant à chaque chromosome est quantifiée via sa propre *fitness*. Les procédures de croisement et de mutation ont pour but de créer de manière permanente de nouveaux chromosomes.

Les algorithmes génétiques gardent la même taille de la *population* des solutions potentielles, appelées aussi *individus*. La population initiale est, généralement, choisie de manière aléatoire.

Cependant d'autres mécanismes d'initialisation peuvent être utilisés suivant l'application. Le nombre de fois qu'un individu est sélectionné pour la recombinaison est fonction de sa *fitness* par rapport au reste de la population. La procédure Algorithme 2.3 présente les différentes étapes de base d'un algorithme génétique. Dans cet exemple, $P(t)$ représente une population de solutions candidates pour un problème donné, à l'itération t . Dans les paragraphes suivants, nous décrivons les différentes procédures qui composent un algorithme génétique [29].

Le cycle d'évolution est répété jusqu' à ce qu'un critère de terminaison est atteint. Dans le but de faciliter le cycle d'évolution de l'AG, deux opérateurs fondamentaux sont créés – croisement et mutation – aussi la routine de sélection peut être prise comme un autre opérateur. Pour simplifier la procédure on choisit, un mécanisme de croisement à un point aléatoire.

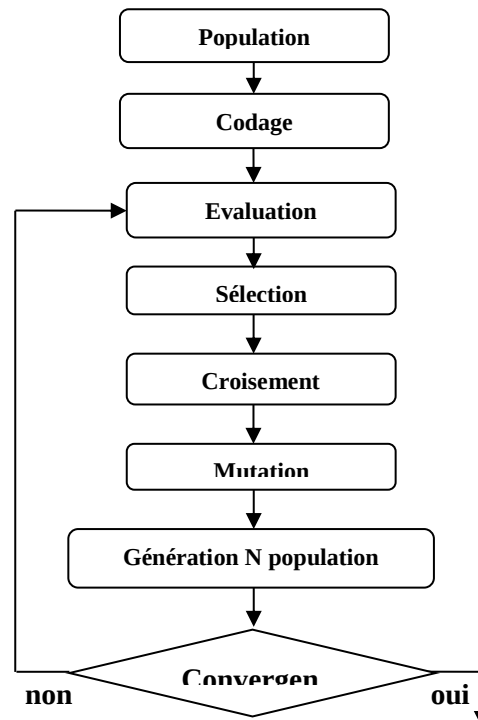


Figure 3.1 Organigramme d'un algorithme évolutionnaire

L'algorithme génétique utilise :

- Une population d'individus (où chacune représente une solution potentielle du problème à optimiser).
- Une procédure de sélection, pour déterminer les individus susceptibles de se reproduire sur la base du critère de sélection naturelle.
- ainsi que quelques opérateurs pour engendrer de nouveaux individus.

Un algorithme génétique générique à la forme suivante :

Algorithme original d'AG par contraste

Début

1. Générer une population aléatoire de n chromosomes.
2. Evaluer la fitness des chromosomes avec la fonction $f(x)$
3. Répéter
4. appliquer l'opération de sélection
5. Appliquer l'opération de croisement avec une probabilité pc
6. Appliquer l'opération de mutation avec une probabilité pm
7. Ajouter les nouveaux chromosomes à la nouvelle population
8. Calculer la fonction fitness $f(x)$, pour tout chromosome x
9. appliquer l'opération de remplacement
10. Jusqu'à la satisfaction des conditions de terminaison

FIN

Algo 3.1 Principe de l'algorithme AG.

Les critères d'évaluation pour tester la convergence sont divers :

- Taux minimum désiré pour l'adaptation de la population au problème.
- Temps de calcul à ne pas dépasser.
- Combinaison des points précédents.

3.5 Fonctionnement général des algorithmes génétiques :

Les algorithmes génétiques sont employés afin de répondre aux problèmes n'ayant pas de solutions calculables en temps raisonnable de façon analytique ou algorithmique.

Les algorithmes génétiques renvoient en résultat des solutions appelées génotypes, répondant plus ou moins au problème, qui sont analysées et éliminées en suivant les principes de l'évolution afin de ne laisser « survivre » que les solutions optimales. Ces procédés permettent de tendre rapidement vers une solution optimale.

L'évolution et la sélection sont indépendantes du problème à résoudre. Par contre il y a des fonctions qui se modifient pour pouvoir répondre au problème :

- La fonction qui s'occupe de générer les familles de départ.
- La fonction inverse qui à partir d'un chromosome permet d'obtenir une solution par décodage du génome

- La fonction qui évalue l'adaptation d'une solution à un problème, sa pertinence

L'avantage des algorithmes génétiques, est qu'il n'est pas nécessaire de connaître la façon dont le problème est résolu. Il suffit juste de fournir une fonction ayant des paramètres permettant de coder l'information sous forme de gène et une autre qui permet d'évaluer la pertinence de la solution.

En conséquence les algorithmes génétiques sont efficaces dans divers domaines :

- Optimisation (de fonction, planification, ...)
- Apprentissage (classification, prédiction, robotique, ...)
- Programmation automatique (programme LISP, automates cellulaires, ...)
- Etude du monde réel (marchés économiques, comportements sociaux, ...)

3.6 Le principe de l'algorithme génétique :

Un algorithme génétique permet trouver une solution en cherchant un extremum sur un ensemble de solutions donnés. Cet algorithme est construit selon :

- Le codage des éléments de la population. C'est à ce moment que l'on détermine les structures de données utiles. Bien entendu cette étape suit la modélisation du problème posé. La réussite des algorithmes génétiques est issue de la qualité de leur codage. Au début, un codage binaire était très utilisé mais maintenant, on travaille avec des codages réels plus pratiques pour des problèmes à variables réelles comme les problèmes d'optimisation.
- La génération de la population initiale. Ce mécanisme doit produire une population non homogène qui évoluera pour permettre de trouver une solution. La convergence ou la divergence peuvent être plus ou moins rapide en fonction de la qualité de cette population initiale.
- Une fonction qui retourne le fitness du brin d'ADN pour pouvoir le comparer.
- Des outils qui permettront de créer de l'aléatoire comme la mutation afin de garantir l'exploration des solutions.
- Des paramètres afin de stopper le programme comme un critère de convergence ou divergence ou encore un nombre d'itérations pour effectuer le programme.

Les algorithmes évolutionnaires constituent une approche originale : on ne cherche pas à trouver une solution exacte mais d'obtenir un résultat satisfaisant le plus possible les critères, souvent contradictoires, des problèmes. Les solutions trouvées ne sont peut-être pas optimales mais sont souvent meilleures que les solutions obtenues grâce à des algorithmes plus conventionnels qui donneront, pour un temps identique, un résultat plus éloigné des besoins.

Un individu est lié à un environnement par son code génétique contenu dans son ADN, Une solution est liée à un problème par son indice de qualité.

3.7 Le codage:

Les codages binaires ont été très utilisés à l'origine. Les codages réels sont désormais largement utilisés, notamment dans les domaines applicatifs pour l'optimisation de problèmes à variables réelles [30]. Chaque variable (information) représente alors un gène et toutes les valeurs que peuvent prendre cette caractéristique représentent les allèles possibles pour ce gène, et en concaténant tous ces gènes pour obtenir un chromosome qui lui représente une solution dans son intégralité.

Il y a trois principaux types de codage utilisables, et on peut passer de l'un à l'autre relativement facilement :

- **Le codage binaire** : c'est le plus utilisé. Chaque gène dispose du même alphabet binaire $\{0, 1\}$, Un gène est alors représenté par un entier long (32 bits), les chromosomes qui sont des suites de gènes sont représentés par des tableaux de gènes et les individus de notre espace de recherche sont représentés par des tableaux de chromosomes.
- **Le codage réel** : cela peut-être utile notamment dans le cas où l'on recherche le maximum d'une fonction réelle. *Exemple*: si le chromosome contient trois variables donc trois gènes et si le premier a la valeur $X_1=3.256$ donc son codage sera 10010011.
- **Le codage de Gray** : le codage de Gray est un codage qui a comme propriété qu'entre un élément n et un élément $n + 1$, donc voisin dans l'espace de recherche, un seul bit diffère.

3.8. Opérateurs de l'algorithme génétique

Les opérateurs utilisés dans l'AG emploient les paramètres suivant:

3.8.1 L'opérateur de sélection:

L'opérateur de sélection est chargé de définir quels seront les individus de P qui vont être dupliqués dans la nouvelle population P' et vont servir de parents (application de l'opérateur de croisement) [30]. Soit n le nombre d'individus de P , on doit en sélectionner $n/2$ (l'opérateur de croisement nous permet de repasser à n individus).

Cet opérateur est peut-être le plus important puisqu'il permet aux individus d'une population de survivre, de se reproduire ou de mourir. En règle générale, la probabilité de survie d'un individu sera directement reliée à son efficacité relative au sein de la population.

On trouve essentiellement quatre types de méthodes de sélection différentes :

- La méthode de la "loterie biaisée" (roulette wheel) de Goldberg [31],
- La méthode "élitiste",
- La sélection par tournois,
- La sélection universelle stochastique.

3.8.2 L'opérateur de croisement:

Une fois certains individus sélectionnés, on les fait se reproduire entre eux, pour cela on utilise l'opérateur croisement. C'est l'opérateur essentiel de recherche d'un algorithme génétique. Il combine les génotypes de deux individus pour en obtenir deux nouveaux. Avec cet opérateur, les génotypes sont vus comme une chaîne de nombres binaires. Plusieurs méthodes de croisement sont utilisées: [32].

- **Croisement en un point:** On choisit au hasard un point de coupure identique sur les deux génotypes et on échange les fragments situés après le point de coupure pour donner les deux nouveaux génotypes.
- **Croisement en deux points:** On choisit au hasard deux points de croisement et on échange les fragments situés entre ces deux points.
- **Croisement en k points:** Généralisation à k points de coupure des méthodes précédentes.

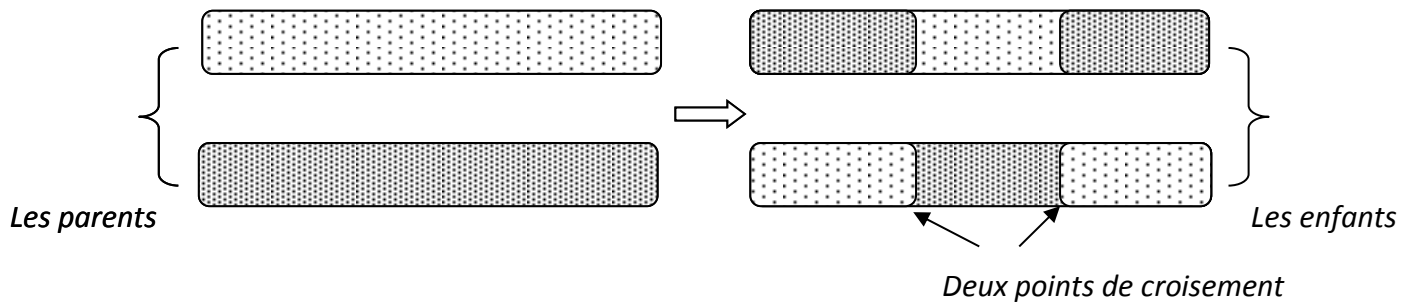


Figure 3.2 Opérateur croisement a deux points

3.8.3 L'opérateur de mutation:

Cet opérateur consiste à changer la valeur allélique d'un gène avec une probabilité pm très faible, généralement comprise entre 0.01 et 0.001. On peut aussi prendre $pm = 1 / lg$ où lg est la longueur de la chaîne de bits codant notre chromosome. Une mutation consiste simplement en l'inversion d'un bit (ou de plusieurs bits, mais vu la probabilité de mutation c'est extrêmement rare) se trouvant en un locus bien particulier et lui aussi déterminé de manière aléatoire; on peut donc résumer la mutation de la façon suivante [33]:

On utilise une fonction censée nous retourner *true* avec une probabilité pm .

Algorithme de mutation

```

Pour chaque locus faire
  Faire appel à la fonction
  | Si cette fonction nous renvoie true alors
  | | on inverse le bit se trouvant à ce locus
  | FinSi
  FinPour

```

Algo3.2 Principe de l'algorithme mutation d'AG.

L'opérateur de mutation modifie donc de manière complètement aléatoire les caractéristiques d'une solution, ce qui permet d'introduire et de maintenir la diversité au sein de notre population de solutions. Cet opérateur joue le rôle d'un "élément perturbateur", il introduit du "bruit" au sein de la population.

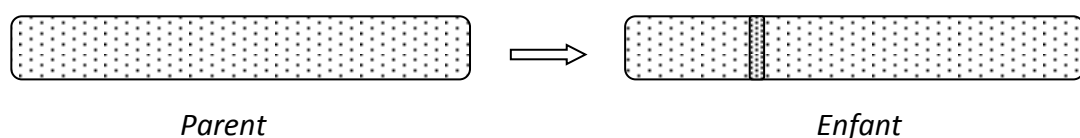


Figure 3.3 Opérateur mutation

3.8.4 Evaluation de la fonction fitness

On appelle adaptation d'une séquence A une valeur positive que nous noterons $f(A)$. f est la fonction objectif ou fitness du problème à résoudre.

Un critère permettant de déterminer l'adaptation d'un individu par rapport à l'environnement afin de classer les individus entre eux. Dans le vocabulaire des algorithmes génétiques, on dénomme par "fitness" (traduction anglaise du mot adaptation) le critère à optimiser.

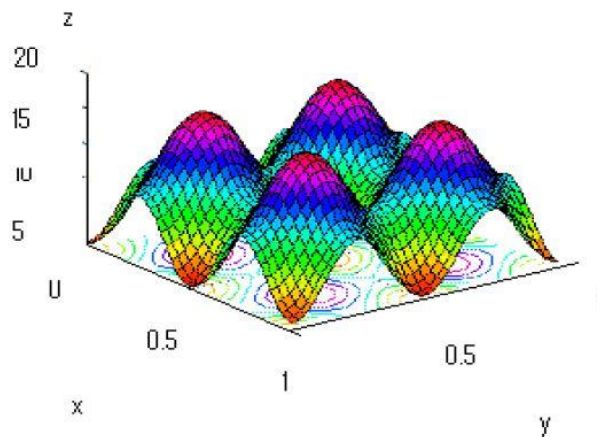


Figure 3.4 représentation de la fitness des individus.

La surface représente la fitness des individus dans un endroit de l'espace, les pics figurent les individus les mieux adaptés à leur environnement. L'objectif est de rapprocher les individus des pics.

3.8.5 Opérateur de remplacement :

Cet opérateur est le plus simple, il permet de générer la nouvelle famille avec les nouveaux brins d'ADN. On trouve essentiellement 2 méthodes de remplacement différentes :

- Le remplacement stationnaire : consiste à remplacer automatiquement les parents par les enfants. Ceci induit une grande diversité de population.
- Le remplacement élitiste : On ne garde que les individus performants en regardant sur la génération complète et non que sur les parents.

3.9 Avantages et Inconvénients des AGs :

3.9.1 Avantages :

- Élimination de solutions non valides.
- Permet de traiter des espaces de recherche important (beaucoup de solutions, pas de parcourt exhaustif envisagé).
- Nombre de solutions important.
- Relativité de la qualité de la solution selon le degré de précision demandé.

3.9.2 Inconvénients :

- Nécessitent plus de calculs que les autres algorithmes méta heuristiques (notamment la fonction évaluation).
- Paramètres difficiles à fixer (taille population, % mutation).
- Choix de la fonction d'évaluation délicat.
- Pas assuré que la solution trouvée est la meilleure, mais juste une approximation de la solution optimale.
- Problèmes des optimums locaux si paramètres mal évalués.

3.10 Conclusion :

Les algorithmes génétiques ne sont pas des algorithmes déterministes ce qui implique qu'ils ne fourniront pas nécessairement de solution optimale exacte. Cependant ces algorithmes, peuvent permettre de trouver une solution optimale et approchée très rapidement.