Cours

Algorithme génétique (AG) :

Fondement, Développement et Application

par :

Pr TAIBI Soufiane

Département d'Electrotechnique

Faculté de Technologie

Université Batna 2

Sommaire

I. L'intelligence artificielle

I.1. Introduction

Comme son nom l'indique, l'intelligence artificielle notée (IA) désigne la science de simuler le raisonnement humain pour faire résoudre des problèmes complexes par un ordinateur (calculateur). Cette façon de procéder est fondée sur des algorithmes de nature très différente de la programmation traditionnelle (calcul scientifique). L'intelligence artificielle s'attaque à la manipulation des symboles plutôt que des nombres. Ces algorithmes sont basés sur la simulation des mécanismes de

raisonnement. L'analyse de divers modes de raisonnement humain permet de déduire des modèles sous forme d'algorithme (programmes).

Les processus et systèmes commandés par une technique d'intelligence artificielle nous donnent l'impression qu'ils se comportent comme un partenaire complètement autonome.

Si on veut être plus concret et donner une définition plus proche de la mise en œuvre de l'intelligence artificielle, on dira qu'on vise à construire un système formel. Ce système est composé de symboles, d'expressions construites à partir de ces symboles et de processus capables de détruire, modifier, créer de nouvelles expressions. On voit ici apparaître l'idée d'un langage informatique au sens traditionnel du terme.

Une approche plus récente de l'intelligence artificielle vise à simuler le traitement de l'information dans le cerveau par un circuit électronique élémentaire, modèle du neurone humain. Les circuits neuronaux sont simulés sous forme d'algorithme et on leur donne une réalisation logicielle du modèle de l'intelligence.

L'intelligence artificielle s'est développée en parallèle avec le calcul numérique. Elle s'attaque à des secteurs d'application réputés difficiles ou impossibles pour celui-ci (calcul numérique).

Nous pouvons citer les principaux secteurs d'application de l'intelligence artificielle :

- La calcul formel
- La vision, l'analyse de texte
- La robotique
- Les machines autonomes : perception, interprétation, décision, action
- Le langage naturel : traduction, compréhension, synthèse

I.2. Techniques de commandes intelligentes

Des techniques modernes sont utilisées pour doter les systèmes et processus industriels de l'intelligence artificielle. Ces techniques sont capables de rendre ces systèmes autonomes pour leur permettre d'analyser des informations et prendre ensuite des décisions convenables à la place de l'homme, et agir en conséquence pour assurer une ou plusieurs actions souhaitées. Parmi ces techniques, nous citons :

- Les réseaux de neurones

- La logique floue
- L'algorithme génétique

II. Réseau de neurones

Les premiers travaux sur les neurones artificiels ont débuté dans les années 1940 et ont été menés par Mc Culloch et Pitts. Ils décrivent les propriétés du système nerveux à partir de neurones idéalisés (neurones logiques 1 ou 0) [1].
Dans les années 1970, a eu lieu une remise en cause de l'intérêt des réseaux de neurones car les ordinateurs à cette époque apprenaient lentement, coûtaient très cher et leur performance n'était pas si impressionnante.
Dans les années 1980, suite au développement des micro-ordinateurs, les réseaux neuronaux ont pris un nouveau départ.

II.1. Bases biologiques

Le neurone biologique Figure (II.1) est une cellule vivante spécialisée dans le traitement des signaux électriques. Les neurones sont reliés entre eux par des liaisons appelées axones, ces axones vont conduire des signaux électriques de la sortie d'un neurone vers l'entrée (synapse) d'un autre neurone. Les neurones font une sommation des signaux reçus en entrée et en fonction du résultat obtenu vont fournir un courant en sortie.

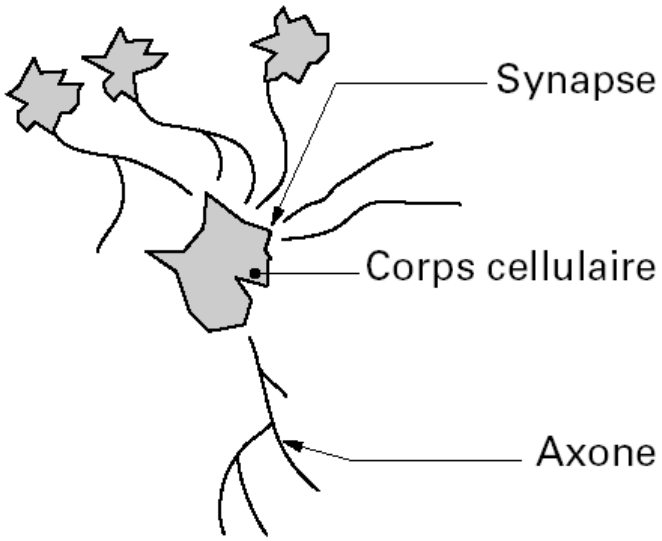

Figure (II.1) : Neurone biologique

II.2. Neurones artificiels

Le neurone artificiel, Figure (II.2), est processeur élémentaire. Il reçoit un nombre variable d'entrées en provenance de neurones appartenant à un niveau situé en amont. A chacune de ces entrées, est associé un poids (w) qui représente la force de connexion. Chaque processeur élémentaire (neurone) est doté d'une sortie unique. Cette sortie alimente un ou plusieurs neurones appartenant à un niveau situé en aval. 3

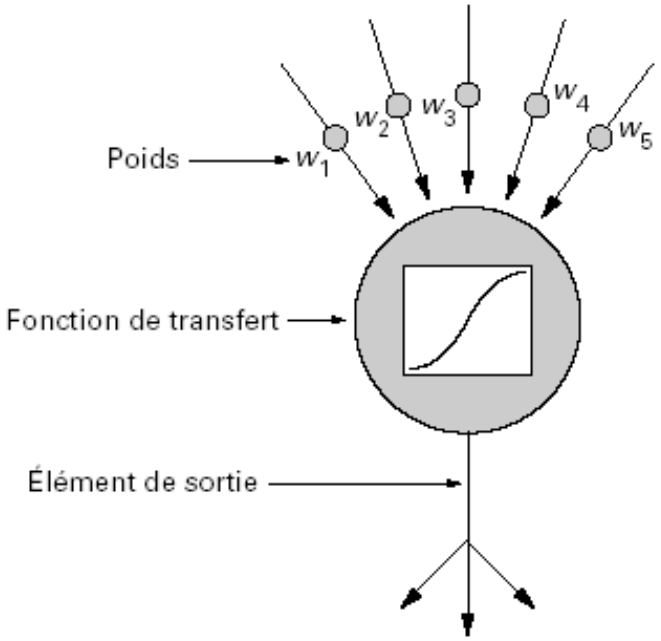

Figure (II.2) : Neurone artificiel

Il existe plusieurs structures d'inter-connexion. Ces structures décrivent la topologie du modèle [1]

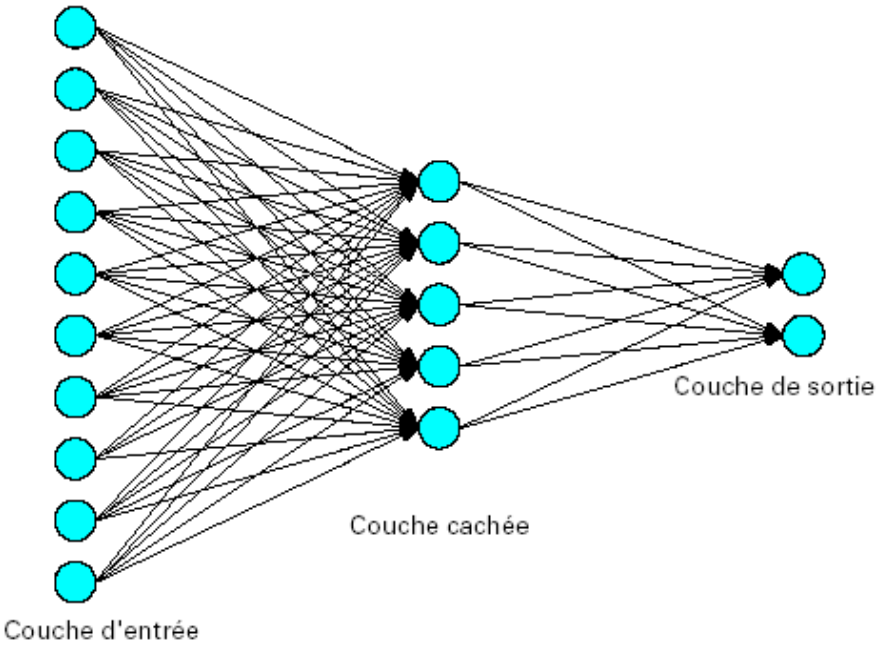

Figure (II.3) : Réseau multicouche classique

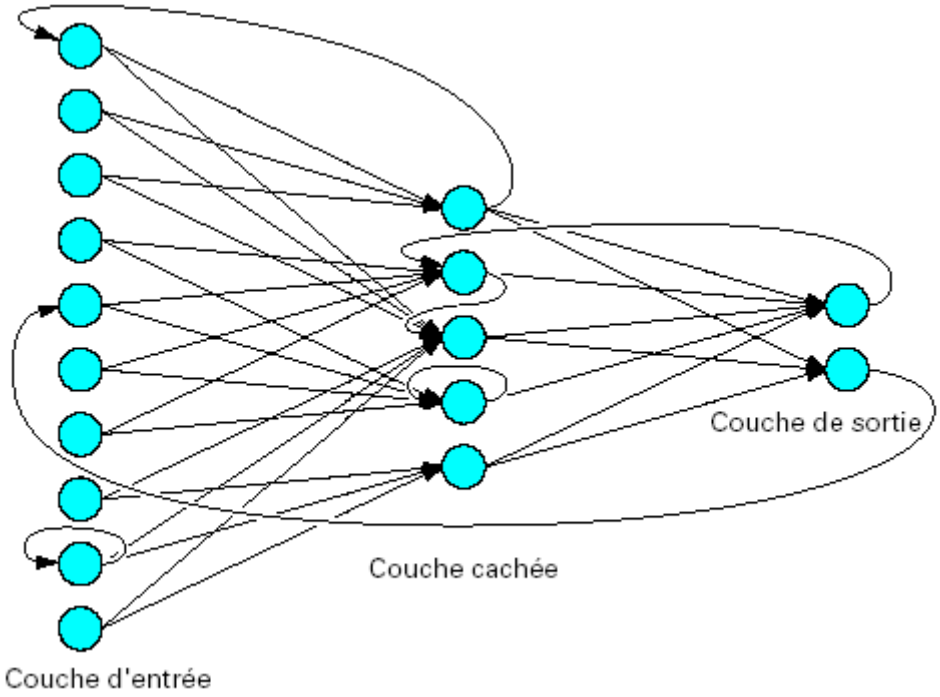

Figure (II.4) : Réseau multicouche à connexions récurrentes

II.3. Apprentissage

L'apprentissage est une phase du développement du réseau de neurones durant laquelle on calcule les poids des neurones (w) de telle manière que les sorties du réseau sont aussi proches que possible des sorties désirées. Cet apprentissage fait appel à des exemples de comportement du processus à modéliser : c'est à dire chaque réseau de neurones prêt à fonctionner ne peut être utilisé que pour l'application pour laquelle il est désigné.

III. La logique floue

Cette technique de commandes a été introduite en 1965. ses principes ont été appliqués pour la première fois en 1974 pour construire le premier contrôleur flou. La logique floue propose une approche beaucoup plus pragmatique que mathématique. Cette technique de contrôle exclut les solutions cartésiennes ou trop déterministes. Depuis quelques années, la commande floue a connu au Japon un essor remarquable. Elle a été appliquée à des problèmes divers : la marche automatisée d'une rame de métro, la purification d'eau, etc.

L'intérêt de cette méthode de commande (par logique floue) réside dans la possibilité de l'utiliser dans le contrôle des systèmes lorsque ceux-ci sont difficilement modélisables ou identifiables ou bien lorsque aucun modèle du système à piloter est disponible. Elle peut être également utilisée dans le cas où les paramètres du système à commander subissent des variations très brutales.

La commande par la logique floue s'appuie sur le savoir-faire humain. Son principe est basé sur la théorie des sous-ensembles flous. Cette théorie permet d'étendre la notion de nombre pour mieux représenter des données imprécises ou vagues. Le principe consiste à ne plus manipuler directement des grandeurs numériques mais à employer des valeurs linguistiques comme « grand », « assez grand », « très grand », etc. On parle alors de variables linguistiques dont les valeurs constituent un ensemble fini de terme linguistique appelé « univers de discours ».

III.1. Philosophie de la logique floue

Soit l'exemple suivant qui décrit le comportement d'un automobiliste à l'approche d'un carrefour contrôle par des feux tricolores. Le processus de conduite met en œuvre les quelques règles suivantes :

Si le feu est rouge	Si ma vitesse est élevée	Si le feu est proche	**Je freine fort**
Si le feu est rouge	Si ma vitesse est faible	Si le feu est loin	**Je maintiens ma vitesse**
Si le feu est orange	Si ma vitesse est moyenne	Si le feu est loin	**Je freine doucement**
Si le feu est vert	Si ma vitesse est faible	Si le feu est proche	**J'accélère**

Prenons la première des quatre règles, si nous essayons de la transposer dans un monde plus mathématique, « moins flou », cela donnerait :
si le feu est rouge, si ma vitesse dépasse 85 km/h, et si le feu est moins de 62m, alors je dirais, j'appuie sur la pédale de frein avec une force de 32 Neuton.

La logique floue formalise le monde exactement comme le fait notre cerveau. Elle apprécie les variables d'entrée de façon approximative (faible, élevé, loin, proche), et fait de même pour les variables de sortie (freinage léger ou fort). Elle emploie un ensemble de règles permettant de déterminer les sorties en fonction des entrées.

III.2. Rappels généraux

Derrière les termes « logique floue » sont regroupées plusieurs notions complémentaires dont la théorie des sous-ensembles flous, la théorie des possibilités et le raisonnement flou. Les bases mathématiques associées sont détaillées dans les documents [2] [3] référencés dans la bibliographie.

Pour l'application qui nous intéresse, la logique floue a été employée dans un but de commande en utilisant un ensemble de règles floues. Un tel système utilise, de manière pragmatique, la notion de sous-ensembles flous pour manipuler, à l'aide de termes linguistiques, les données du système, ainsi que les principes du raisonnement flou pour exploiter le jeu de règles.

III.3. Sous-ensembles flous

Introduite par L.A. Zadeh en 1965, la théorie des sous-ensembles flous permet d'étendre la notion de nombres pour mieux représenter des données imprécises ou vagues. Le principe consiste à ne plus manipuler directement des grandeurs numériques, mais à employer des « valeurs linguistiques » comme « Grand », « Assez-Grand », « TrèsGrand », etc. On parle alors de variables linguistiques dont les valeurs constituent un ensemble fini de termes linguistiques appelé « univers du discours ». Les domaines de valeurs numériques des variables sont partitionnés en classes. Chaque classe, identifiée par un terme linguistique, est caractérisée par une fonction définissant le degré d'appartenance des valeurs numériques à cette classe. Ces classes ne sont pas exclusives, mais présentent des domaines de valeurs communs ; une même valeur peut donc appartenir à plusieurs classes, son appartenance partielle à chaque classe étant caractérisée par un degré d'appartenance spécifique.

Exemple :

Si on considère la vitesse d'une automobile, on peut considérer une variable linguistique « Vitesse » caractérisée selon trois classes : « Faible », « Normale » et « Élevée ». Tout le monde s'accorde sur le fait que 180 km/h est une vitesse « Élevée » et 20 km/h une vitesse « Faible ». Mais comment qualifier une allure de 70 km/h ? Certains diront qu'il s'agit d'une vitesse « Normale » alors que d'autres la considéreront comme « Faible ». La théorie des sous-ensembles flous permet d'accorder les deux points de vue en considérant 70 km/h comme une vitesse à la fois « Faible » et « Normale » figure (III.**5**).

Dans l'exemple, 70 km/h est considérée comme une vitesse « Faible » avec un degré de 0,25 et comme une vitesse « Normale » avec un degré de 0,75.

Les fonctions en forme de trapèze qui définissent les degrés d'appartenance des vitesses aux classes sont arbitraires et leurs formes doivent être définies en fonction du problème (la forme trapézoïdale est présentée à titre d'illustration, toute autre fonction est possible).

Cette approche permet d'assurer une transition progressive entre les classes, ce qui n'est pas possible en logique booléenne où les classes ne se recouvrent pas.

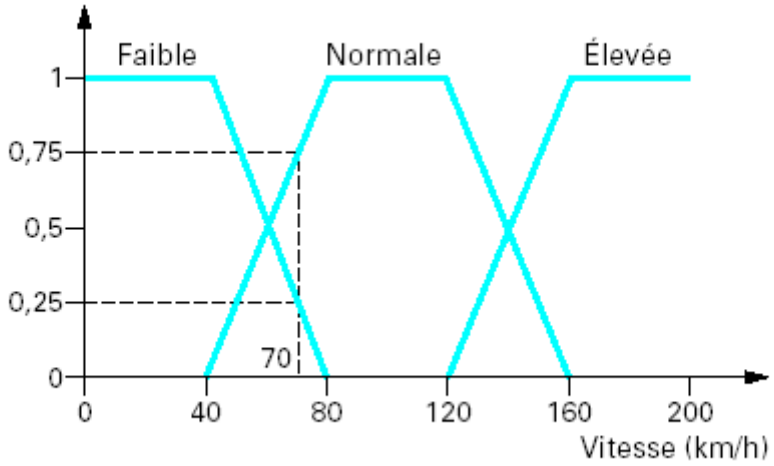

Figure (III.5) : Exemple de sous-ensembles flous. Vitesse

IV. ALGORITHME GENETIQUE

IV.1. Introduction

Depuis quelques années, les recherches s'orientent vers l'optimisation de dispositifs électromagnétiques par le biais de différentes approches. Ces dernières sont plus ou moins contraignantes et précises. En effet, les paramètres à optimiser sont souvent interdépendants et il est difficile de trouver la solution optimale prenant en compte les différentes interactions.

En fait, trouver la solution optimale d'un problème dans un espace complexe implique un compromis entre deux objectifs : l'exploitation des meilleures solutions et l'exploration robuste de l'espace de recherche. Les méthodes de type grimpeur procèdent itérativement en tentant, à chaque pas, de trouver localement une solution intermédiaire meilleure que la solution courante ; ce genre de méthode est pénalisée par son incapacité à traiter des problèmes représentant des reliefs de solutions multimodales (système possédant plusieurs optimas locaux).

Les algorithmes génétiques (AG) représentent une stratégie de recherche réalisant un compromis équilibré entre l'exploration de l'espace de recherche et l'exploitation des meilleures solutions. Des analyses théoriques ont montré que les algorithmes génétiques gèrent ce compromis de façon optimale [4].

Le but de ce polycop est de présenter le principe de la méthode d'optimisation basé sur l'AG, son fondement ainsi que les différentes étapes nécessaires à son développement, pour finir avec une application qui présente un problème nécessitant une procédure d'optimisation pour trouver la solution optimale.

IV.2. Définitions

L'optimisation par algorithme génétique prend son origine dans les mécanismes de la sélection naturelle et la génétique de l'évolution. Cette méthode a été mise en œuvre par

J.H. Holland dans les années 70 [5]. Comme son nom l'indique, elle est basée sur la traduction mathématique des phénomènes naturels qui sont la reproduction des espèces, la survie et l'adaptation des individus. Cette traduction est exploitée pour la résolution de problèmes nécessitant l'optimisation d'une fonction ou d'un système dépendant de plusieurs paramètres et qui ont besoin d'être calculés pour un critère bien défini (maximisation, minimisation, …).

Cette technique constitue une méthode d'optimisation robuste. L'AG peut résoudre, avec fiabilité, des fonctions représentant des reliefs de solution réputés très difficiles pour les méthodes d'optimisation classiques (Simplex, le plus fort gradient …). Les fonctions réputées difficiles sont des fonctions qui présentent plusieurs optima locaux, des fonctions discontinues ou des fonctions à plusieurs dimensions où les méthodes ordinaires ne peuvent pas prendre en compte l'effet d'interaction entre tous les paramètres.

IV.3. caractéristiques de l'AG

Les principales caractéristiques relatives à cette technique se concentrent autour des quatre points [5][6] :

- le parallélisme : l'algorithme génétique travaille en parallèle sur un certain nombre de candidats et non pas sur un candidat unique. La méthode de recherche est globale et couvre tout l'espace de recherche.

- l'algorithme génétique peut manipuler des entités qui ne sont pas forcément numériques, à condition que les points de cet espace soient toujours constitués d'un ensemble d'entités élémentaires.

- L'utilisation minimale d'informations : il n'a besoin que de la mesure d'adéquation (la qualité d'une solution), il ne repose sur aucune autre information, par exemple des dérivés ou hypothèses telles que la continuité et la différentiabilité. Il ne requiert qu'une capacité à classer les solutions entre elles.

- L'utilisation de règles probabilistes plutôt que déterministes dans l'exploration de l'espace de recherche. L'introduction du hasard est très bénéfique pour l'optimisation de fonctions présentant plusieurs optimas et aussi en cas de fonction non permanente (déplacement ou changement des optimas au cours du temps).

Ces caractéristiques donnent à la technique d'optimisation par algorithme génétique, une grande capacité pour localiser la niche de l'optimum global [6].

IV.4. Principe de fonctionnement

L'AG est une de recherche méthode itérative. Le but est d'optimiser une fonction définie par l'utilisateur appelée *fonction objectif* ou *fonction d'adéquation.*

Pour atteindre cet objectif, l'algorithme travaille en parallèle sur une population de points candidats appelés *individus* ou *chromosomes* (solution particulière) qu'on note **I**, chaque individu est constitué d'un ensemble d'éléments appelés *gènes* notés $\mathbf{x_n}$ figure (IV.1.a). Dans l'algorithme génétique de base, tel qu'il a été fondé par Holland, les gènes sont formés de **1** et **0**. Dans ce cas, chaque valeur réelle $\mathbf{x_n}$ (paramètres à optimiser) est codée par son équivalent en binaire et l'individu obtenu est représenté par une chaîne codée de plusieurs gènes représentant une solution particulière pour la fonction objectif figure (IV.1.b).

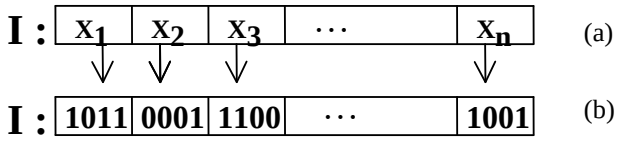

Figure (IV.1) : Représentation d'un individu; codage réel (a), codage binaire (b)

De nouvelles versions d'algorithme génétique sont apparues. Elles ne se basent plus sur le codage binaire mais elles travaillent directement sur les paramètres réels.

Ces versions, appelées algorithmes génétiques codés réels, offrent d'une part l'avantage d'accélérer la recherche et d'autre part de rendre plus facile le couplage avec d'autres méthodes d'optimisation. Ce codage est de plus en plus répandu [4][7]. C'est cette solution que nous allons utilisée dans les applications du présent polycop.

Le but de l'algorithme génétique est de chercher la combinaison optimale des gènes constituant l'individu, qui donne lieu à la meilleure adéquation. A chaque itération, appelée *génération,* est généré une nouvelle population avec, toujours, le même nombre d'individus au total (n). Cette population est mieux adaptée à l'environnement tel qu'il est représenté par la fonction objectif et les critères de l'optimisation (maximisation, minimisation, …). Plus on progresse dans les générations, plus les individus vont devoir tendre vers l'optimum de la fonction objectif.
Le passage d'une génération à l'autre s'effectue en trois étapes, évaluation puis sélection et enfin reproduction avec des opérateurs de *croisement* puis de *mutation*.

Ces différentes étapes vont permettre, à la fin de la procédure, de trouver la combinaison optimale de gènes constituent l'individu le mieux adapté.

IV.5. Evaluation

L'algorithme génétique évalue une population d'individus qui forme la génération courante qu'on appelle (G_t). Les individus les plus forts, au sens des critères de la fonction objectif, auront théoriquement plus de descendants, que les autres individus, dans la génération qui suit, donc (G_{t+1}). Si l'on transpose dans un sens mathématique, l'algorithme génétique évalue la fonction d'adéquation que nous notons (F) pour chaque individu (I) de la population courante $P(G_t)$. Suivant les critères de l'optimisation, un classement entre individus sera effectué et les meilleurs, qui répondent le mieux aux critères de la fonction objectif auront une probabilité de reproduction (clonage) plus importante que les autres. C'est cette information qui guidera l'algorithme génétique vers les meilleurs individus.

Nous devons faire très attention aux probabilités de reproduction attribuées aux différents individus, dans le sens où, si un individu est très supérieur à la moyenne, il constituera presque exclusivement la population suivante et on perdra toute diversité d'où un risque de convergence prématurée. S'il y a très peu de différence entre les qualités d'adéquation des individus (niveau d'adaptation), la recherche stagnera et se comportera comme une «promenade aléatoire ».
Pour éviter ces deux éventualités (convergence prématurée ou promenade aléatoire) une solution consiste à ranger les individus par ordre croissant ou décroissant : croissant si le critère est une minimisation de la fonction objectif (F) et décroissant dans le cas contraire (maximisation de la fonction objectif).

Après avoir rangé les individus on leur attribue une probabilité de sélection qu'on note p_i. Cette probabilité est attribuée pour chaque individu selon son rang dans le rangement par la relation [4]:

$$p_i = \frac{P_{sel}(n-1)-(r_i-1)(2P_{sel}-2)}{n(n-1)}$$

la quantité r_i représente le rang de l'individu *i*, n le nombre d'individus et P_{sel} la pression de sélection. P_{sel} appartient à l'intervalle [1 , 2] et représente le nombre moyen d'enfants du meilleur individu. La quantité $n\,p_i$ donne le nombre moyen d'enfants pour chaque individu du rang *i*, donc le moins bon des individus a nécessairement en moyenne $(2 - P_{sel})$ enfants.

$$\sum_{i=1}^{n} p_i = 1$$

$$\sum_{i=1}^{n} np_i = n$$

Cette stratégie de sélection, appelée sélection par rangement, permet d'assurer un changement d'échelle dynamique non linéaire. C'est cette méthode que nous allons utilisée, dans les différentes applications du présent polycop, pour sa robustesse et sa simplicité.

IV.6. Sélection

Suivant les probabilités de reproduction données aux individus, au moment de l'évaluation, on fera la sélection stochastique des individus pour construire une population intermédiaire, notée $P'(G_t)$, toujours constituée de *n* individus. On procèdera suivant le principe de « la roue de la fortune », la roue est lancée autant de fois qu'il y a d'individus (*n* fois).
Autrement dit, les meilleurs individus, qui répondent au mieux à la fonction d'adéquation, vont construire théoriquement la plus grande partie de la population intermédiaire $P'(G_t)$ toujours constituée de *n* individus.

IV.7. Reproduction avec les opérateurs de croisement et de mutation

Une fois l'étape de sélection achevée, l'algorithme génétique poursuit sa recherche par l'application des opérateurs de croisement et de mutation. L'opérateur de croisement joue le rôle de recombinaison et d'échange entre certains individus. Quant à l'opérateur de mutation, il modifie « localement » un individu en changeant sa composition.

IV.7.1. Croisement

Un pourcentage de la population intermédiaire sélectionnée $P'(G_t)$ sera soumis au croisement. Ainsi, l'opérateur de *croisement* choisit au hasard, et avec une probabilité fixée qu'on note p_c, deux individus (deux parents) parmi cette population intermédiaire. Il construit alors deux enfants en faisant l'échange de certains gènes

choisis aléatoirement d'un parent avec ceux de l'autre. Les deux enfants issus de ce croisement sont injectés dans la population que l'on note $P''(G_t)$. Cette dernière sera alors constituée d'un pourcentage d'enfants issus du croisement. Le reste est issu directement de la population $P'(G_t)$ sans aucune modification. Le nombre total d'individus dans $P''(G_t)$ est toujours égal à *n* individus.

Il existe plusieurs type d'opérateurs de croisement :

IV.7.1.1. Croisement un point ou discret

Pour ce type de croisement, on choisira au hasard "un site de coupe" entre les deux parents, figure (IV.2). Cela nous amène à prendre un nombre *k* (indice de site) entre 1 et (*L*-1) où *L* représente le nombre de gènes d'un individu. On obtient alors deux enfants en prenant les *k* premiers gènes du premier parent et les (*L-k*) derniers gènes du second parent. L'autre enfant est obtenu avec les *k* premiers gènes du second parent et les (*L-k*) derniers gènes du premier parent. Ces deux enfants sont, comme indiqué ci dessus, injectés dans la population $P''(G_t)$.

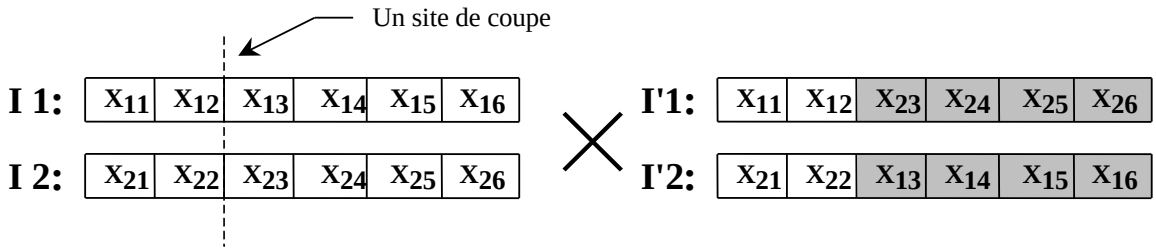

Figure (IV.2) : Croisement un point (exemple d'un individu à six gènes)

IV.7.1.2. Croisement deux points

Le croisement deux points est basé sur le même principe que le croisement un point. La différence réside dans le fait que deux sites de coupe sont introduits au hasard entre les deux parents. Comme le montre la figure (IV.3), les gènes se trouvant entre les deux sites de coupe sont échangés respectivement entre les deux individus (parents) pour former les deux enfants qui rejoindront la population $P''(G_t)$.

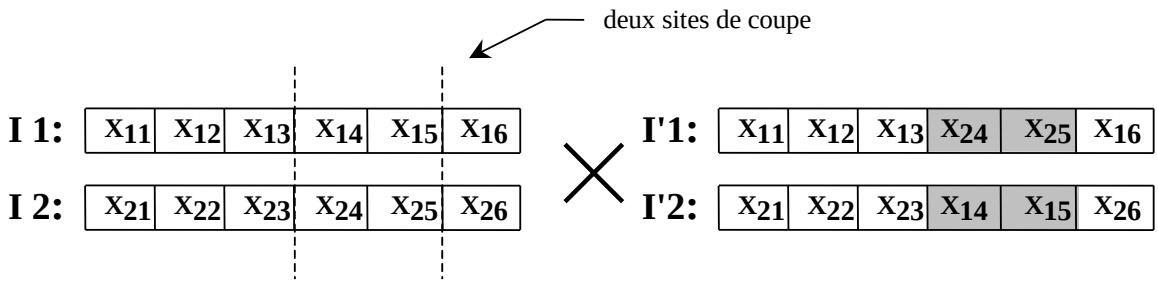

Figure (IV.3) : Croisement deux points

IV.7.1.3. Croisement continu (ou uniforme)

Ce croisement effectue une opération de type moyenne sur certains gènes des parents. Chaque gène a une chance sur deux d'être moyenné avec son homologue chez l'autre parent -voir figure (IV.4).

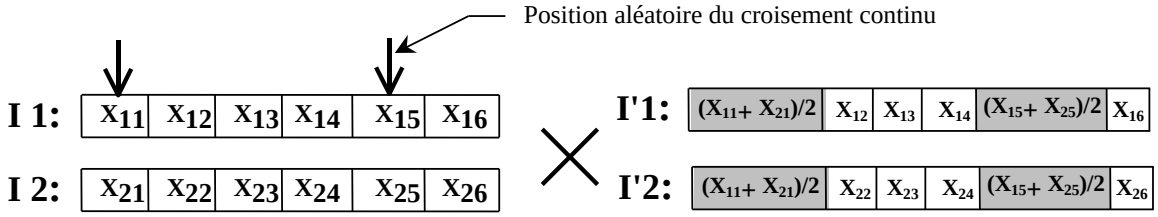

Figure (IV.4) : Croisement continu (uniforme)

IV.7.1.4. Croisement arithmétique

Ce type de croisement est proche du croisement continu. Comme précédemment, on choisit aléatoirement deux positions d'échange puis on effectue une moyenne arithmétique pondérée par un coefficient *a*. soit, pour la figure (IV.5) les deux nouveaux gènes :

$$Y_{13} = a\,X_{13} + (1-a)\,X_{23}$$

et

$$Y_{23} = (1-a)\,X_{13} + a\,X_{23}$$

qui remplacent X_{13} et X_{23}. Ils forment les enfants qui intègrent la population intermédiaire $P''(G_t)$. La valeur de *a* est générée aléatoirement dans l'intervalle $[0\,,1]$. On note que si $a = 0.5$, on retrouve le croisement continu.

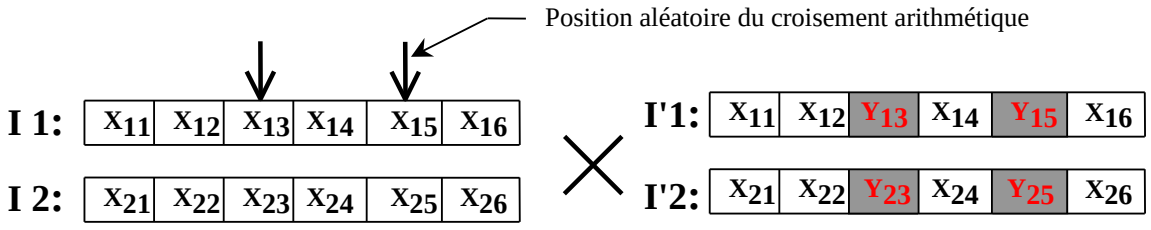

Figure (IV.5) : Croisement arithmétique

On peut aussi, appliquer une autre loi de croisement. Les gènes qui se trouvent par exemple dans les sites de croisement X_{11} et X_{21}, sont remplacés respectivement par :

$$X_{11} + a(X_{21} - X_{11})$$

et

$$X_{21} + a(X_{11} - X_{21})$$

Une fois l'étape de croisement achevée on applique la procédure de mutation.

IV.7.2. Mutation

Chaque gène des individus de la population $P''(G_t)$ peut subir une *mutation* avec une probabilité fixée notée p_m. L'opérateur de mutation agit donc en modifiant

aléatoirement un ou plusieurs gènes d'un individu. La valeur du gène muté est remplacée par une autre appartenant au même domaine de variation.

On considère les individus choisis aléatoirement, parmi les individus de la population $P''(G_t)$, qui sont soumis à l'opérateur de mutation. Pour chacun de ces individus, on choisit également aléatoirement des gènes qui vont subir une modification.

Il est à noter que les gènes concernés par la mutation vont subir une modification importante durant les premières générations. Par la suite, le niveau des modifications ira en diminuant au fur et à mesure que la recherche évoluera vers la solution.

Il existe plusieurs types d'opérateurs de mutation qui permettent d'assurer cette modification, nous citons ici :

IV.7.2.1. Mutation uniforme

Pour chaque gène qui mute, on prend deux nombre *s* et *r*. Le premier « s » peut prendre les valeur ± 1. Dans le cas où *s* est égal à 1, le changement est positif, dans l'autre cas le changement est négatif. Le second « r » détermine l'amplitude du changement. C'est un nombre généré aléatoirement dans l'intervalle [0 1]. Dans ces conditions, le gène X'_j qui remplace le gène qui mute X_j est calculé à partir de l'une des deux relations suivantes [4] :

$$X'_j = X_j + \left(X_{\max} - X_j\right)\left(1 - r^{(1-\frac{G_t}{G_F})^5}\right) \quad \textbf{si} \quad s = 1$$

$$X'_j = X_j - \left(X_j - X_{\min}\right)\left(1 - r^{(1-\frac{G_t}{G_F})^5}\right) \quad \textbf{si} \quad s = -1$$

Où $X_{\min}$ et $X_{\max}$ désignent respectivement les limites inférieure et supérieure de la valeur du paramètre X_j et $G_F \leq G_T$ représente la génération pour laquelle l'amplitude de la mutation s'annule. A partir de cette génération les individus ne subissent plus de mutation, et l'exploration du reste de l'espace de recherche est assurée uniquement par l'action de croisement.

IV.7.2.2. Mutation non uniforme

On remplace directement la valeur du gène qui mute, par exemple $\boldsymbol{x_j}$, par une autre valeur prise aléatoirement dans l'intervalle $[X_j^{\min} \quad X_j^{\max}]$.

Il existe un autre type de mutation non uniforme. Dans ce cas, la valeur mutée X'_j du gène X_j est donnée par la formule [6]:

$$X'_j = \begin{cases} X_j + \Delta(t, y) \\ \quad ou \\ X_j - \Delta(t, y) \end{cases}$$

$$\Delta(t, y) = y\, r \left(1 - \frac{G_t}{G_F}\right)^b$$

Dans cette expression, y peut prendre les valeurs $(X_j^{\max} - X_j)$ ou $(X_j - X_j^{\min})$ à condition que le résultat de l'opération ne sorte pas de l'intervalle $[X_j^{\min} \quad X_j^{\max}]$. Le paramètre r représente un nombre aléatoire avec $0 < r < 1$. b est un paramètre qui définit le degré de non uniformité. La fonction $\Delta(t, y)$ renvoie, d'une façon aléatoire, un nombre dans l'intervalle $[0 \quad y]$. Elle permet de réaliser une recherche uniforme dans les premières générations, et plus pointue au fur et à mesure que l'on avance.

D'autres lois peuvent être utilisées pour calculer l'évolution des gènes en mutation. On peut remplacer le gène qui mute X_j par X'_j telle que X'_j est calculé par :

$$X'_j = X_j \pm \Delta X_j$$

Avec :

$$\Delta X_j = (X_{\max} - X_{\min}).2^{-k\,r}$$

la quantité k est une constante souvent prise égale à 16 [8].

Après la mutation, les individus constitueront la nouvelle population $P(G_{t+1})$ de n individus qui donne naissance à la nouvelle génération. Le cycle continue : évaluation, sélection, reproduction (croisement et mutation), évaluation, etc. jusqu'à la dernière génération fixée.

Il y a donc quatre paramètres de base qui doivent être fixés pour assurer le fonctionnement d'un AG : le nombre d'individus dans la population *n*, la génération maximale G_T, les taux de croisement p_c et de mutation p_m.

Trouver de bonnes valeurs à ces paramètres est un problème parfois délicat. La valeur de *n* qui désigne le nombre d'individus de la population, dépend fortement du problème à optimiser (en particulier le nombre de gènes de chaque individu) [4]. En pratique, on arrive à une solution acceptable en prenant pour les valeurs de p_c et p_m respectivement 0.7 et 0.005.

IV.8. Récapitulatif de la procédure AG

Nous résumerons, par l'organigramme de la figure (IV.6), l'ensemble des étapes permettant d'effectuer une optimisation avec l'algorithme génétique.

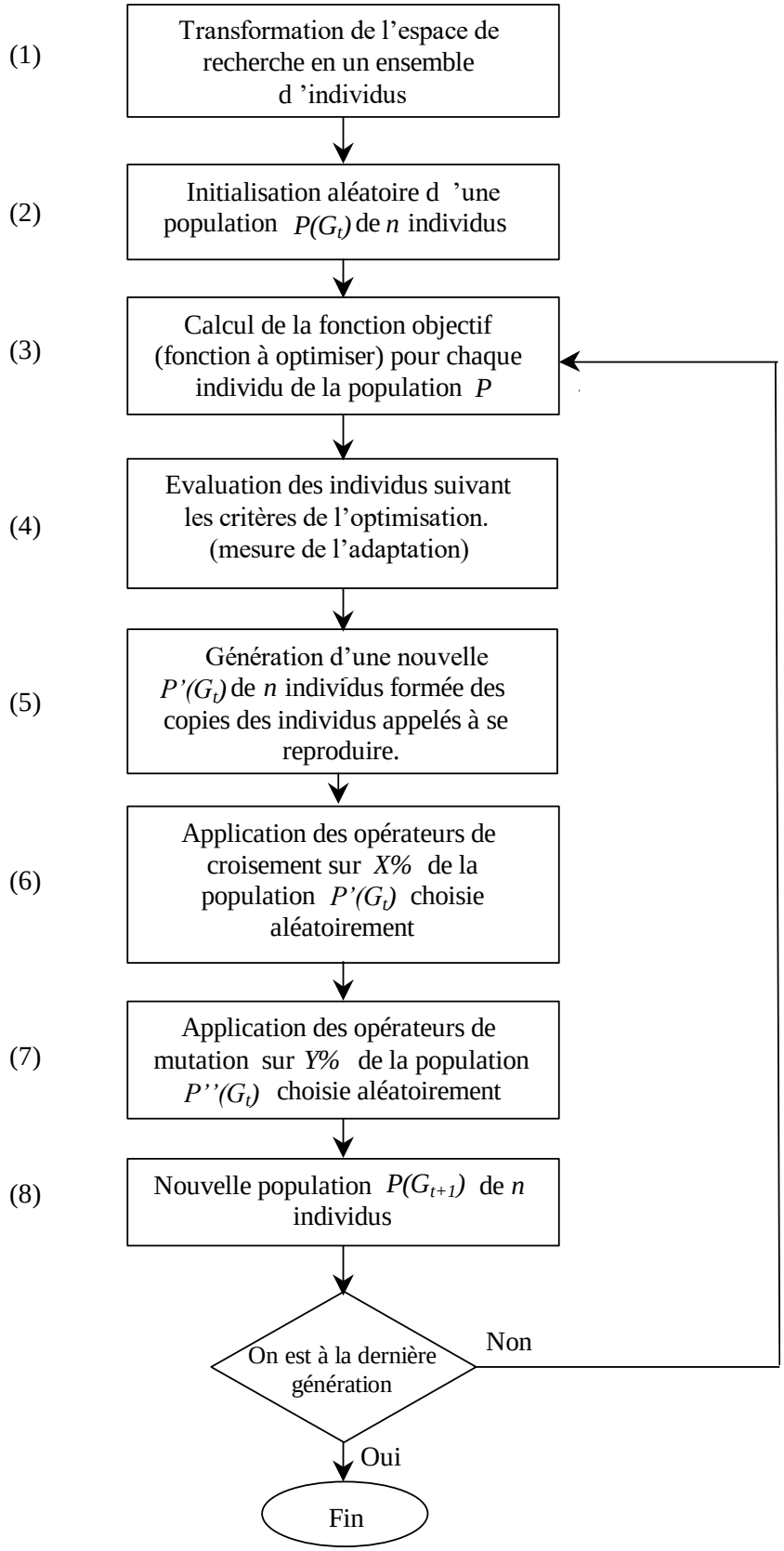

Figure (IV.6) : Les différentes étapes de l'algorithme génétique

V. Conclusion

Dans le présent polycop nous avons abordé le domaine de l'intelligence artificielle. Nous avons présenté à ce niveau trois techniques qui permettent de doter les systèmes et processus industriels de l'intelligence artificielle. Ces techniques sont capables de rendre ces systèmes autonomes pour leur permettre d'analyser des informations et prendre ensuite des décisions convenables à la place de l'homme, et agir en conséquence pour assurer une ou plusieurs actions.

Les trois techniques abordées sont :

- Les réseaux de neurones
- La logique floue
- L'algorithme génétique

En effet, nous avons présenté brièvement les deux premières techniques à savoir : les réseaux de neurones et la logique floue. Nous avons ensuite bien approfondi le développement sur la technique de l'algorithme génétique.

Dans ce polycop, nous avons fait une synthèse sur les différents opérateurs et fonctions (évaluation, sélection, croisement et mutation) nécessaires à l'écriture d'une procédure d'optimisation basée sur l'AG. Nous avons également présenté les résultats issus de l'application de la procédure développée sur plusieurs fonctions objectif.

Le choix des différents paramètres nécessaires à l'exécution d'une procédure d'optimisation AG est un problème très délicat et dépend essentiellement de la nature de la fonction objectif (problème à optimiser) et du temps de calcul. Le plus souvent, on teste plusieurs valeurs et on garde celles qui réalisent un bon compromis entre les temps de calcul et une meilleure exploration de l'espace de recherche.

Nous pouvons conclure, à travers l'ensemble des applications, que l'optimisation par l'algorithme génétique possède une très grande capacité à localiser la niche de l'optimum global. En effet, elle peut, à partir d'un espace de recherche initialement

inconnu, localiser les sous-espaces prometteurs, qui repondent au mieux à une fonction objectif.

Pour atteindre l'optimum global, il est possible d'utiliser des méthodes hybrides : Algorithme génétique-grimpeurs (Simplex, le plus fort gradient, …). Ce dernier fonctionnera en parallèle avec l'AG pour améliorer la solution obtenue.

V. Références bibliographiques :

[1] Fabrice SORIN, Lionel BROUSSARD, Pierre ROBLIN
Régulation d'un processus industriel par réseaux de neurones
Techniques de l'Ingénieur, traité Informatique industrielle
S 7 582

[2] KAUFMAN A.
Introduction à la logique floue. Techniques de l'Ingénieur, Mesures et Contrôle, R 7 032, 1992.

[3] BOUCHON-MEUNIER B.
La logique floue
Éditions Que sais-je ?

[4] J .M. Renders
Algorithmes génétiques et réseaux de neurones
Editions Hermes, Paris.

[5] J.H. Holland
Adaptation in natural and artificial systems
Ann Arbor, University of Michigan Press, 1975.

[6] L. Saludjian, J.L. Coulomb, A. Izabelle
Genetic algorithm and Taylor development of the Finite Element solution for Shape Optimization of Electromagnetic devices
IEEE Tans Mag., vol 34, n°5, Sept 1998, pp: 2841-2844.

[7] F. Zaoui
Méthodes d'optimisation associées à la modélisation numérique : application à la conception et au diagnostic des systèmes électromagnétiques.
Thèse de Doctorat, LGEP, Paris XI, octobre 1999.

[8] B. Sareni, L. Krähenbühl, A. Nicolas
Efficient genetic algorithms for solving hard constrained optimization problems
IEEE Transactions on magnetics,
Vol. 36, n°4, July 2000, pp :1027-1030.

[9] S. Taïbi :
Contribution à l'étude, la conception le dimensionnement et l'optimisation de Machines à réluctance variable de type Vernier.
Thèse de Doctorat, UST Lille 2002.

[10] S. Taïbi, A. Tounzi, F. Piriou
The use of Genetic Algorithm to reduce the torque ripples of a Vernier Reluctance Machine
Studies in Applied Electromagnetics and Mechanic., pp :217-222.

[11] A. Gallardo, D. A. Lowther
Some aspects of niching genetic algorithms applied to electromagnetic device optimisation
IEEE Transactions on magnetics, Vol. 36, n°4, July 2000, pp :1076-1079.

[12] Bachir Abdelhadi, Azeddine Benoudjit, and Nasreddine Nait-Said
Application of Genetic Algorithm With a Novel Adaptive Scheme for the Identification of Induction Machine Parameters
IEEE transactions on energy conversion, vol. 20, no. 2, june 2005

[13] Sylviane GENTIL
Intelligence artificielle appliquée à l'automatique
Techniques de l'Ingénieur, traité Mesures et Contrôle
R 7 215

[14] Arnold KAUFMANN
Introduction à la logique floue
Techniques de l'Ingénieur, traité Informatique industrielle
A 120 - R 7 032

[15] Jean-Pierre BARRAT, Mireille BARRAT, Yves LÉCLUSE
Application de la logique floue : commande de la température d'un four
Techniques de l'Ingénieur, traité Informatique industrielle
R 7 428

[16] Jean-Christophe RIAT, Jean-Pierre AURRAND-LIONS
Pilotage de direction automobile
par logique floue
Techniques de l'Ingénieur, traité Mesures et Contrôle
R 7 429

[17] Fabrice Rossi
Réseaux de neurones
http://apiacoa.org/contact.html