Métaheuristique pour l'optimisation TP6 120 crédit

Deniz Sungurtekin

December 2020

1 Introduction

Le but de ce travail pratique est de nous familiariser avec le "Genetic Algorithms" en l'utilisant pour minimiser une fonction donnée. Cette méthode est une métaheuristique de population qui consiste à faire évoluer des individus à travers des générations qui subiront une phase de sélection où les meilleurs candidats seront sélectionnés, une phase de croisement consistant à concevoir la nouvelle génération dont les individus posséderont, selon une certaine probabilité, une partie de l'information de ses deux parents, puis finalement chaque individus aura une certaine chance de subir des mutations. Dans un premier temps, nous allons expliquer la méthodologie utilisée afin d'implémenter la simulation de cette évolution darwiniste puis nous analyserons et discuterons les résultats obtenus.

2 Méthodologie

L'objectif est donc de minimiser la fonction suivante:

$$f(x,y) = -|\frac{1}{2}xsin(\sqrt{|x|})| - |ysin(30\sqrt{|\frac{x}{y}|})|$$

où x,y $\in [10, 1000] \cap N$

Tout d'abord, il est nécessaire de définir un individu, celui-ci sera représenté par un objet possédant une séquence de 20 bits dont les 10 premiers bits seront utilisé pour encodé x et les 10 derniers y. En plus de cette séquence, chaque individus possédera une évaluation de leur séquence qui correspondra à leur énergie. Nous utilisons une représentation en bits car le "Genetic Algorithms" est spécialement efficace dans cette espace de recherche. Par conséquent, il est nécessaire d'effectuer un mapping sur x et y dans l'intervalle donné avant d'évaluer notre séquence. Voici donc la fonction de mapping utilisée:

$$map(x) = round(\frac{x}{2^{10}}(1000 - 10) + 10)$$

Avec cela, il nous ai donc possible d'implémenter notre algorithme. Celui-ci sera constitué de quatre fonctions:

1. generateIndividu(N): Prenant en argument une taille N et retournant une liste contenant N individus dont la séquence à été générée aléatoirement.

2. tournament(Individus): Cette fonction choisie aléatoirement 5 individus et retourne celui-ci possédant la meilleur énergie.

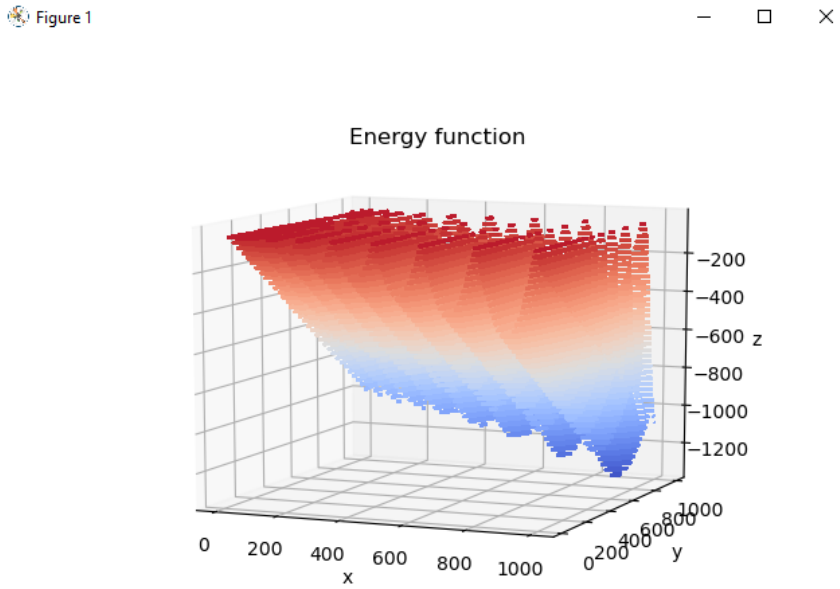
3. crossOver(Individu1,Individu2): Prenant deux individus en paramètre et retourne avec une probabilité de 60% deux individus possédant chaque moitié de séquence des deux parents. Sinon la fonction retourne les individus en paramètre sans aucun changement.

4. mutation(Individu,p_m): Cette fonction retourne un même individu dont les bits auront une chance p_m de subir une mutation (un simple switch entre 0 et 1).

La première génération sera donc initialisée grâce à l'appel de la première fonction. Ensuite, il suffit de d'appeler chacune des fonctions restantes dans l'ordre à chaque itération. La deuxième fonction sera appelée N fois pour conserver la même taille de population et le crossover sera appelé N/2 fois puisqu'elle parcours les individus par paire. Après la phase de mutation, il suffira de mettre à jour et d'évaluer les attributs de chaque individus puisque chacune des phases ne modifie que leur séquence. Après le nombre d'itération atteint, on renvoi simplement la meilleur énergie parmi tous les individus de la dernière génération.

3 Résultat

Dans un premier temps, affichons la fonction à minimiser pour visualiser la difficulté à trouver le minimum global:



On distingue clairement plusieurs minimums locaux très proche l'une de l'autre dont le minimum global se trouve en z = -1356.482683119401.

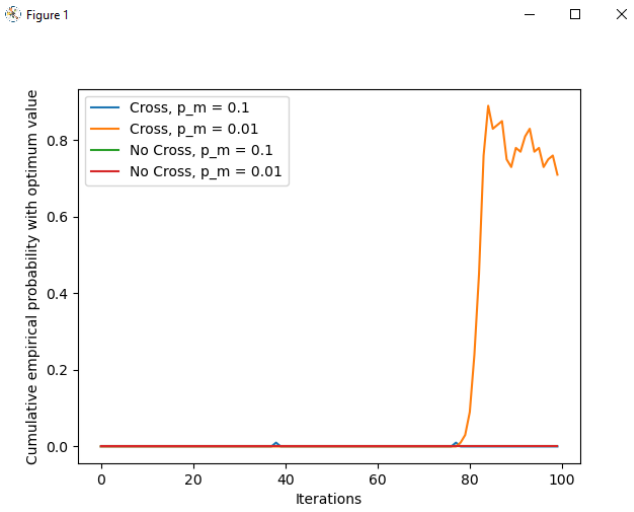
Analysons maintenant nos résultats obtenus dans quatre contextes différents avec 100 individus et 100 générations (itérations):

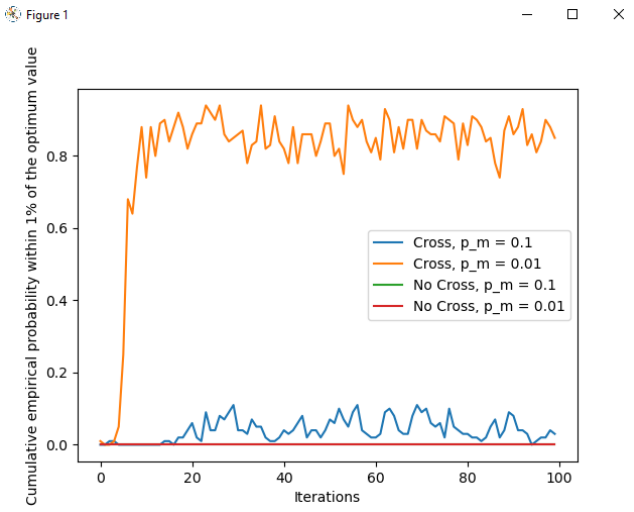
1. Avec une probabilité de mutation $p_m = 0.01$ et le crossOver() défini précédemment.
2. Avec une probabilité de mutation $p_m = 0.1$ et le crossOver() défini précédemment.
3. Avec une probabilité de mutation $p_m = 0.01$ sans le crossOver() défini précédemment.
4. Avec une probabilité de mutation $p_m = 0.1$ sans le crossOver() défini précédemment.

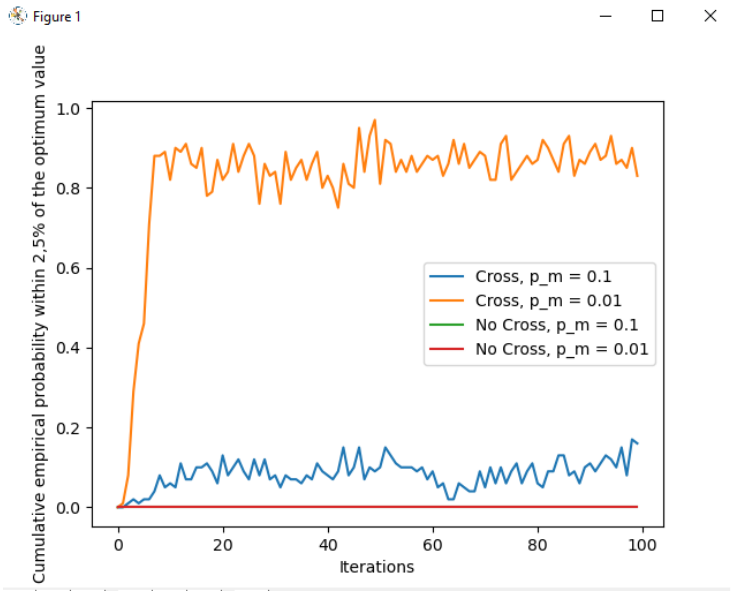
Cela nous permettra d'analyser l'impact de ces deux paramètres dans la résolution du problème.

D'abords analysons l'évolution de la probabilité empirique cumulative à travers chaque génération sous 3 configurations différentes. La première consiste à calculer le nombre d'individus possédant exactement la valeur du minimum global/N à chaque générations. La deuxième consiste à prendre en compte les individus se trouvant à 1% du résultat optimal puis finalement ceux se trouvant à 2,5% du résultat optimal. De plus, afin de changer le "seed" des valeurs aléatoires générées, nous prendrons en compte la moyenne des résultats sur 5 run de l'algorithme.

Voici les graphes obtenus pour les trois configurations:



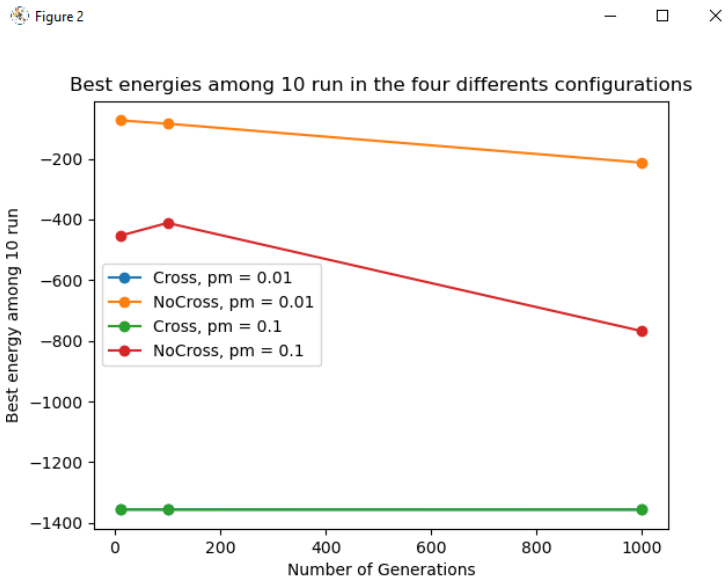


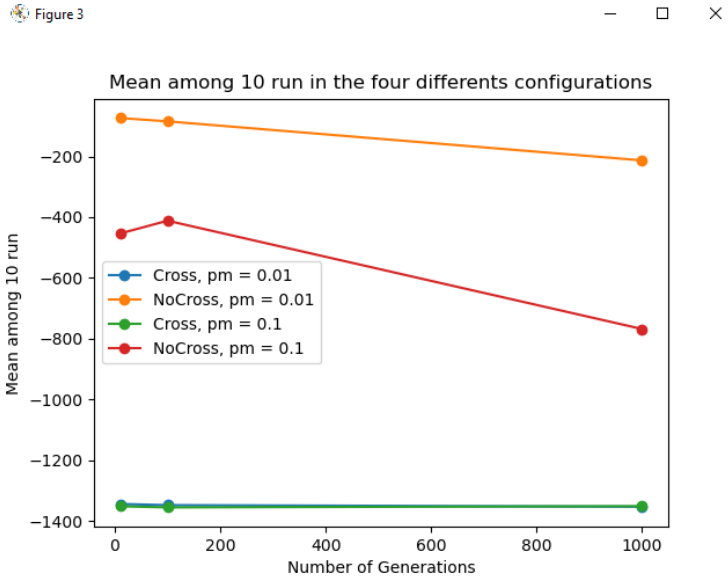


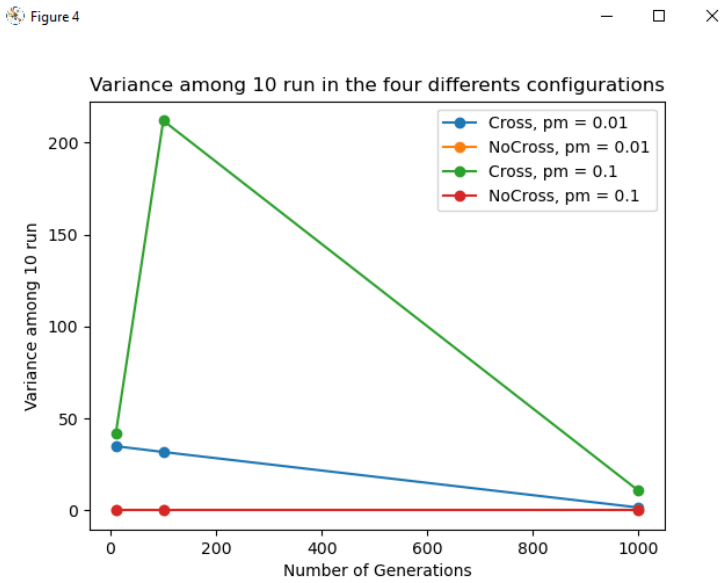
On observe ici que c'est uniquement avec un crossover et une valeur de $p_m = 0.01$ qu'on obtient un taux important de valeur optimal (ou proche de celle-ci) trouvée. Cela s'explique par le fait que lorsqu'une valeur optimal est trouvée, celle-ci à moins de chance de subir des mutations négatives qui nuiront à cette convergence accentuée par le phénomène de sélection. On observe aussi qu'avec un taux de mutations trop grand (0.1), on aura parfois tendance à trouver un individu optimal qui aura des chances très élevé de subir une mutation sur un de ces vingts bits et donc de disparaître.
Pour ce qui est de l'évolution lorsqu'il n'y a pas de crossover, on remarque que la mutation ne produit pas une diversification assez forte pour explorer le domaine de recherche et donc de trouver des solutions assez proche ou égal à la valeur optimal.
Forcément plus la tolérance d'acceptation de valeur est grande plus la probabilité empirique cumulative l'est, puisque si une valeur est optimale elle est aussi incluse dans les deux autres configurations.

Analysons maintenant les résultats obtenus pour une évaluation de la fitness de 10^3, 10^4 et 10^5. Cette valeur peut être estimé par le produit entre la taille de la population et le nombre de génération. Par conséquent, nous allons observer les meilleurs résultats obtenues, leur moyenne et leur variance à travers 10 runs pour chacun des quatre contextes définis avec 10, 100 et 1000 itérations (générations) puisque nous travaillons avec 100 individus.

Voici les graphes obtenus:







Premièrement, on observe que ce sont de loin les méthodes qui incluses le crossover qui possèdent les meilleurs résultats, puisque ceux si sont toujours extrêmement proches ou égales à la valeur optimal. Cependant, on note une amélioration quasi linéaire dans les méthodes sans crossover, car le seul élément de diversification provient des mutations, on observe donc que plus la probabilité correspondante est grande plus la pente de progression est améliorée. Cependant, cette amélioration est quasiment instantané avec le crossover puisqu'en regardant la valeur moyenne, on voit qu'au bout de dix générations tous les individus sont déjà très proche de la valeur optimal. De plus, on explique que les deux premiers graphes sont très semblable par le fait que les méthodes sans crossover possède une diversification si petite que les individus ont tendance à être sélectionnés de la même manière à travers tout les runs. Pour ce qui est des méthodes avec cross, tout les individus converge tellement vite que la valeur moyenne est très similaire au meilleur résultat.
En ce qui concerne la variance, on voit effectivement que le crossover est le facteur impactant le plus la diversité, car pour les méthodes sans crossover la variance est presque nulle. Cependant, on observe également que le taux de mutation impact la diversité en comparant les deux méthodes avec crossover. Effectivement avec $p_m = 0.1$, on voit une grande variance dans les 100 premiers runs qui se stabilisera de manière linéaire dû au phénomène de sélection qui aura tendance à choisir toujours les mêmes candidats plus le nombre de génération augmente. D'un autre côté, avec $p_m = 0.01$, il y a moins de diversité mais cela est compensé par le crossover et semble suffire à stabiliser le système et le faire converger, puisqu'au bout d'un certain nombre d'itération la variance devient nulle. De plus, on a observer avec la mesure de la probabilité empirique cumulative que c'est dans cette configuration que l'ensemble des individus se rapproche le plus de la valeur optimal.