UNIVERSITE DE HAUTE-ALSACE

Algorithme de l'Artificial Bee Colony

Olivier DELIERRE, Ilyasse TISSAFI, Michael GERWILL, Antonin LECLERC

14 janvier 2018

1 Introduction

Dans la nature, différentes espèces vivent en groupe : les bancs de poissons, les nuées d'oiseaux, les troupeaux, les abeilles... Ces dernières sont très bien organisées et très rigoureuses dans leur travail. Des chercheurs se sont donc intéressé au comportement des abeilles pour créer une métaheuristique.

Une métaheuristique est un algorithme d'optimisation permettant de résoudre des problèmes d'optimisation difficile pour lesquels on ne connaît pas de méthode classique plus efficaces. Nous allons donc dans ce rapport parler de l'algorithme ABC (Artificial Bee Colony)[3].

Dans cet algorithme on distingue trois types d'abeilles : les employées, les éclaireuses et les spectatrices qui ont chacune une tâche bien définie.

Table des matières

2 L'algorithme

2.1 Explication humaine

L'algorithme simule le comportement d'une colonie d'abeilles, il y a ainsi trois types d'abeilles :

— Les **employées** cherchent de la nourriture autour de la source de nourriture qu'elles ont en mémoire et partagent ces informations avec les abeilles spectatrices.

— Les **spectatrices**, elles, choisissent les meilleures sources de nourriture parmi celles que les abeilles employées leur ont transmises.

— Les **éclaireuses**, anciennes abeilles employées dont la source de nourriture n'a pas été retenue, cherchent aléatoirement une nouvelle source de nourriture.

2.2 Explication machine

L'algorithme procède en réalité de la manière suivante :

1. On commence à générer un nombre défini de solutions, possédant chacune une dimension précisée.
2. Pour chaque étape d'évolution :
   (a) On évalue la fitness des différentes solutions
   (b) Pour chaque solution, on génère une nouvelle solution, et on vérifie si celle-ci est meilleure que la précédente. Si elle l'est, la nouvelle solution remplace l'ancienne. Sinon, un compteur d'essai spécifique à la solution est incrémenté
   (c) Pour chaque solution, nous décidons aléatoirement, selon la fitness de la solution, si nous lui ré-appliquons la même chose que précédemment.
   (d) Puis, chaque solution ayant dépassé un nombre d'essais défini est régénéré.

2.2.1 Formules

Afin d'arriver à nos fins, certaines formules ont été choisis[2] spécialement pour cette algorithme. Les voici :

Génération des solutions :

$$x_{mi} = l_i + rand(0,1) * (u_i - l_i) \tag{1}$$

l_i : limite haute du problème
u_i : limite basse du problème

Génération des nouvelles solutions pour les abeilles employées :

$$\nu_{mi} = x_{mi} + rand(0,1) * (x_{mi} - x_{ki}) \tag{2}$$

x_k : une solution aléatoire
i : une dimension aléatoire

Transformation de la fitness pour calculer une minimisation :

$$fit(\vec{x_m}) = \begin{cases} \dfrac{1}{1 + f_m(\vec{x_m})} & \text{si} f_m(\vec{x_m}) >= 0 \\ 1 + abs(f_m(\vec{x_m}) & \text{si} f_m(\vec{x_m}) < 0 \end{cases} \quad (3)$$

3 Contraintes

Voici les différentes contraines qui nous ont été imposées :

1. Nombre maximum d'exécutions : 30
2. Dimension du problème : 30
3. Nombre d'individus par population : 30
4. Nombre total d'appel de la fonction objectif : 2×10^3
5. Nous devions calculer la moyenne ainsi que l'écart type
6. Nous devions comparer nos résultats avec un algorithme de la littérature.

4 Conception de l'algorithme

4.1 Les classes

Cinq classes sont présentes dans le code de cet algorithme. Ceux-ci ont été imposés dans un but de clarté. Les voici :

1. La classe **Benchmark** qui contient les différentes fonctions permettant de calculer f(x) de chaque Benchmark.
2. La classe **SetUpParams** contenant les différents paramètres du problème, défini dans la fonction de démarrage "main".
3. La classe **Problem** composé des différents éléments représentant un problème (c'est à dire, les bornes haute et basse du problème, sa dimension, ainsi que le code du Benchmark qui sera utilisé).
4. La classe **Solution** qui contient les valeurs de la solution pour chaque dimension, la fitness, la fitness remaniée (qui permet de générer des probabilités pour les abeilles spectatrices), ainsi que le problème à résoudre.
5. Pour finir, la classe **MyAlgorithm** qui permet de faire dérouler l'algorithme. Elle contient la totalité des fitness des différentes solutions, les différentes probabilités pour chaque solution d'être choisie, ainsi que le nombre d'essais pour chaque solution.

   La principale fonction de cette classe est `"evolution()"`, qui envoie les différentes abeilles en fonction des paramètres qui ont été indiqués.

4.2 Schémas

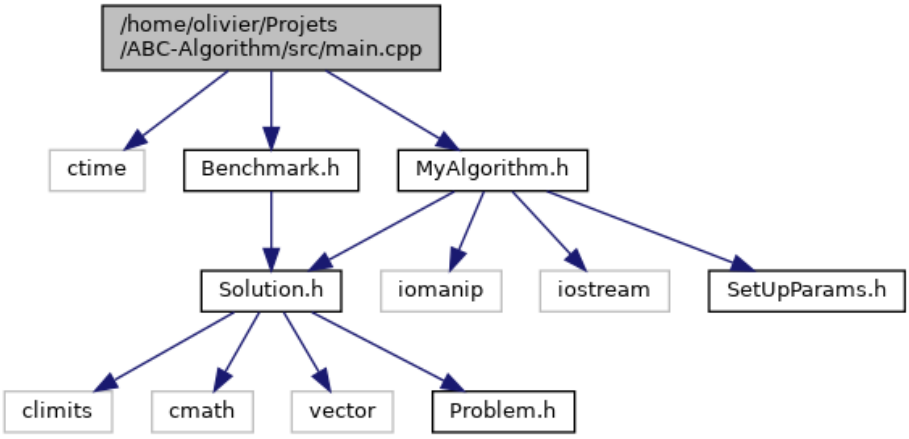

FIGURE 1 – Diagramme des dépendances des classes.

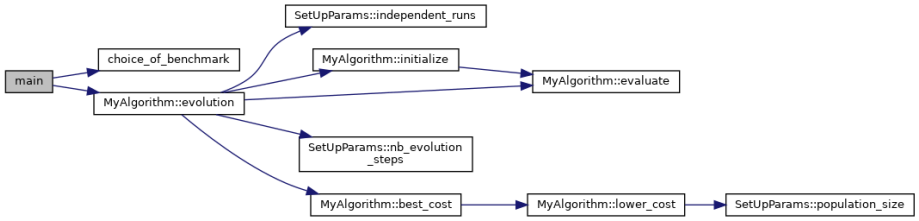

FIGURE 2 – Diagramme des appels fonction depuis le main.

5 Benchmarks

Voici les différents Benchmarks qui ont été utilisés afin de vérifier l'efficacité de notre algorithme. Ceux-ci ont été trouvés à l'aide du site de l'**Université Simon Fraser du Canada**[1].

5.1 Ackley

$$f(x) = -a \times exp\left(-b\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}\right) - exp\left(\frac{1}{d}\sum_{i=1}^{d} cos(cx_i)\right) + a + exp(1) \quad (4)$$

5.2 Rastrigin

$$f(x) = 10d + \sum_{i=1}^{d} \left[x_i^2 - 10cos(2\pi x_i)\right] \quad (5)$$

5.3 Rosenbrock

$$f(x) = \sum_{i=1}^{d-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right] \quad (6)$$

5.4 Schaffer

$$f(x) = 0.5 + \frac{sin^2(x_1^2 - x_2^2) - 0.5}{[1 + 0.001(x_1^2 + x_2^2)]^2} \quad (7)$$

5.5 Schwefel

$$f(x) = 418.9829d - \sum_{i=1}^{d} x_i sin(\sqrt{|x_i|}) \quad (8)$$

5.6 Weierstrass

$$f(x) = \sum_{k=0}^{29}\sum_{n=0}^{20} a^n cos(b^n \pi x_k) \quad (9)$$

où $a = 0.5$ et $b = 12$

6 Conclusions

Après réalisation de cet algorithme, la simplicité de celui-ci saute rapidement aux yeux. En plus d'être plutôt performant, celui-ci est plus ou moins simple d'approche pour quelqu'un ayant déjà des connaissances en méta-heuristique.

6.1 Olivier DELIERRE

En plus de m'avoir appris énormément de chance sur le réel fonctionnement d'une intelligence artificielle, ce projet m'a permis d'apprendre à travailler en équipe sur un sujet encore inconnu pour nous. Ce fût une expérience intéressante, et très instructive, bien que complexe, au vu des différents problèmes que nous avons pu rencontrer pendant la réalisation de cet algorithme.

6.2 Ilyasse TISSAFI

Ce projet portant sur l'intelligence artificielle a été très instructif pour moi, d'une part car l'on a découvert à travers ce projet la science qu'est l'intelligence artificielle, son fonctionnement, son but etc. Et aussi d'autre part car on avait carte blanche afin de mener à bien ce projet. Aussi, l'algorithme ABC, qui est un algorithme très récent a été vraiment intéressant à étudier.

6.3 Michael GERWILL

La mission qui nous a été confiée a sûrement été la plus complexe que nous ayons eu à traiter depuis le début de notre cursus universitaire. De plus, le fait que ce projet ait démarré de façon rapide sans réelles connaissances nous a efforcé à réaliser de multiples recherches avant de pouvoir réellement commencer la phase de réalisation. Selon moi, ce projet a été mené à bien notamment grâce à un esprit d'équipe et d'entraide, mais aussi à notre persévérance et à notre détermination sans faille. Nous avons été confrontés à de multiples reprises à des difficultés en terme de compréhension et de conception, donc par ce fait, été contraint de chercher des solutions, d'être autonome ainsi que de se concerter entre membres du groupe pour favoriser au mieux l'aboutissement du projet. En résumé, ce projet aura été l'occasion de découvrir l'intelligence artificielle, ainsi que d'approfondir nos compétences de développement en C++.

6.4 Antonin LECLERC

Ce projet a été pour moi une expérience enrichissante car il m'a permis de découvrir l'intelligence artificiel à travers un problème concret, l'algorithme ABC. Je pense que notre groupe s'est bien organisé, en effet nous avons commencé par de nombreuses recherches afin de constituer un document à suivre pour ensuite coder l'algorithme ce qui nous a grandement aidé. J'ai rencontré des difficultés pour implémenter le code de l'algorithme dans la structure imposée mais mes collègues ont pu m'éclairer. Je suis satisfait de ce projet car il m'aura permis de progresser en C++ et dans l'utilisation de GitHub

Références

[1] Derek Bingham. Virtual library of simulation experiments. http://www.sfu.ca/~ssurjano/index.html, 2017.

[2] Dervis Karaboga. Artificial bee colony algorithm. http://scholarpedia.org/article/Artificial_bee_colony_algorithm, 2010.

[3] Dervis Karaboga. Artificial bee colony algorithm homepage. http://mf.erciyes.edu.tr/abc/, 2010.

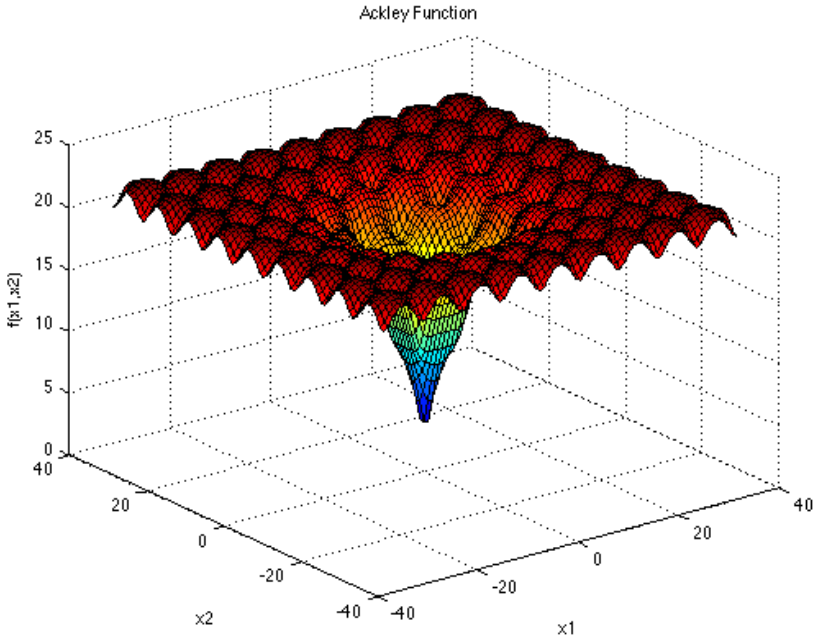

FIGURE 3 – Fonction d'Ackley.

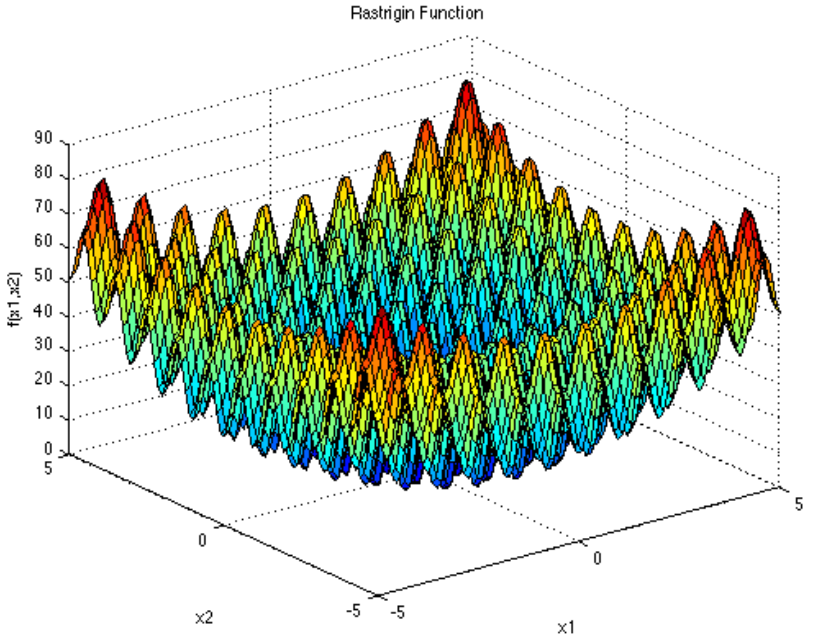

FIGURE 4 – Fonction de Rastrigin.

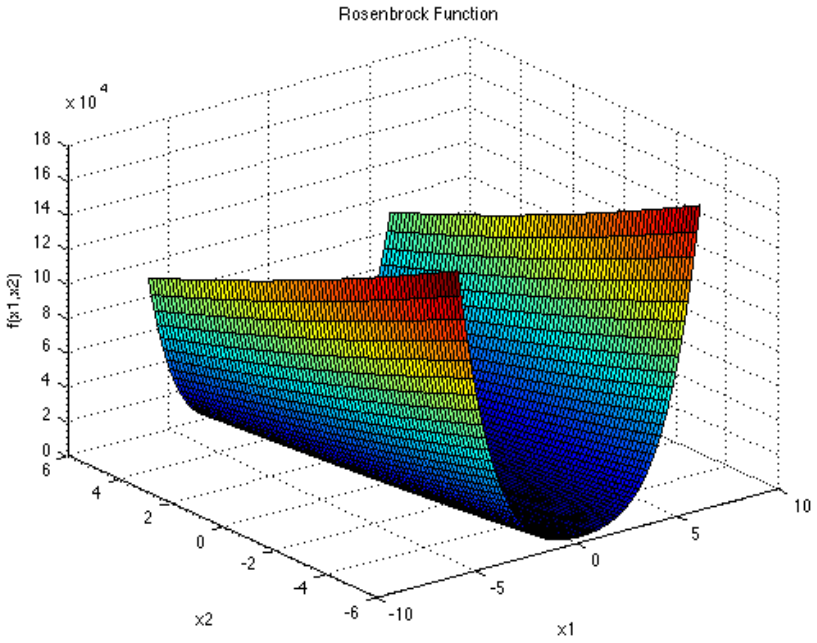

FIGURE 5 – Fonction de Rosenbrock.

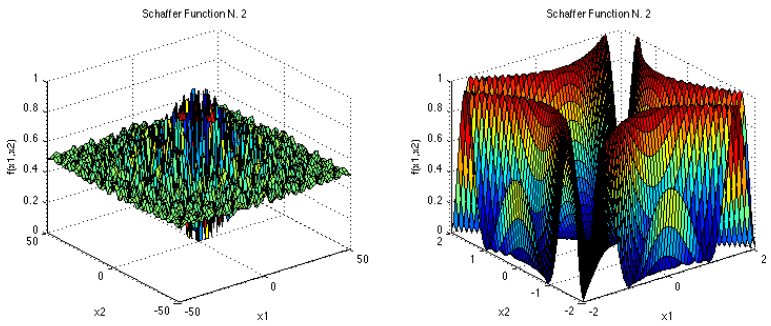

FIGURE 6 – Fonction de Schaffer version 2.

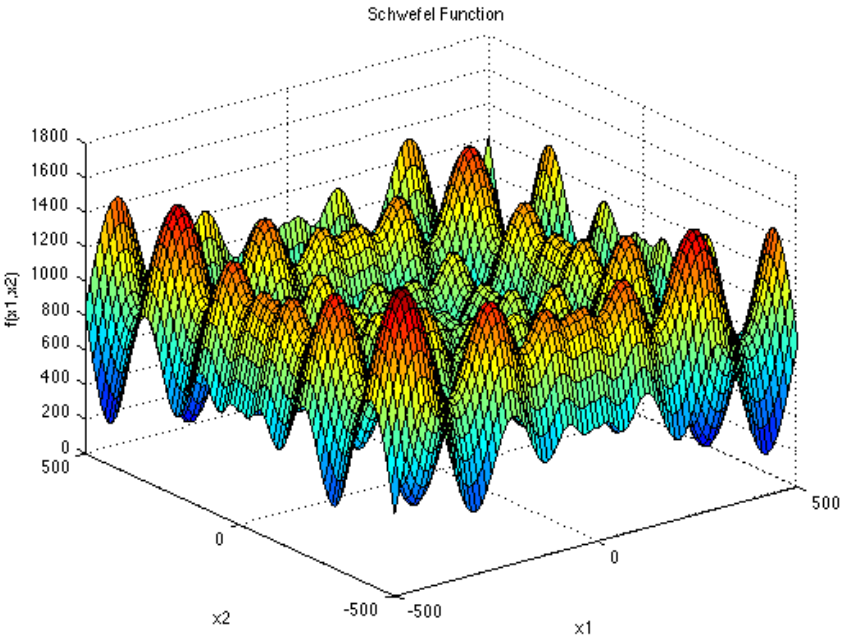

FIGURE 7 – Fonction de Schwefel.

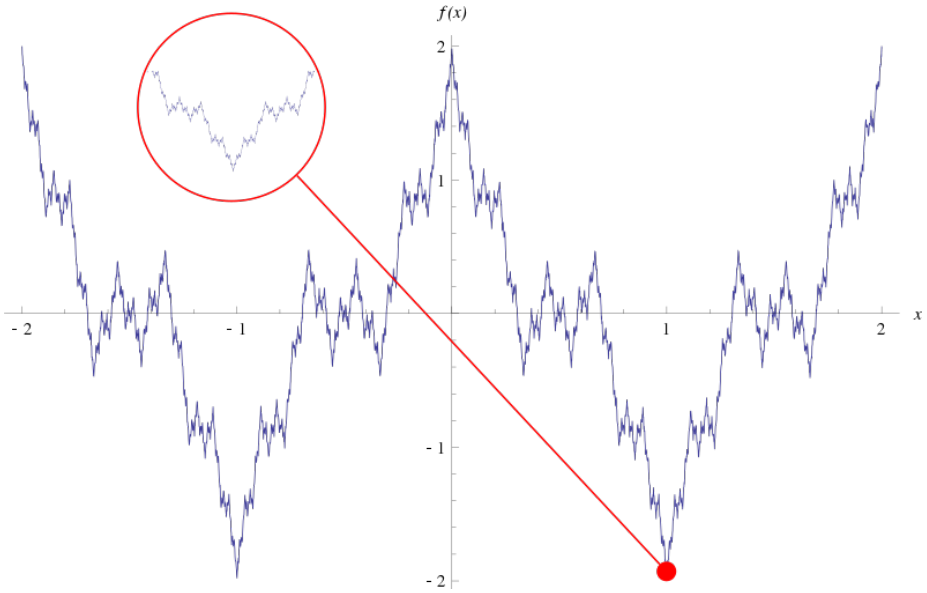

FIGURE 8 – Fonction de Weierstrass.