Algorithmes Évolutionnaires — Master 2 MIAGE IA^2
Travaux dirigés N^o 2 : le problème de la partition

Andrea G. B. Tettamanzi
Université côte d'Azur
andrea.tettamanzi@univ-cotedazur.fr

Année universitaire 2020/2021

Résumé

On va adapter l'algorithme génétique simple développé lors de la séance précédente à un problème bien plus difficile que le « *MaxOne* », notamment le problème de la partition, qui est NP-complet.

1 Introduction

Le problème de la partition, en tant que problème de décision, peut être énoncé comme suit : étant donné un ensemble fini A et une taille $t(a) \in \mathbb{N} \setminus 0$ pour chaque $a \in A$, existe-t-il un sousensemble $A' \subseteq A$ tel que

$$\sum_{a \in A'} t(a) = \sum_{a \in A \setminus A'} t(a) \; ? \tag{1}$$

Ce problème a été démontré être NP-complet [2], même si il peut être résolu en temps pseudo-polynomial par programmation dynamique [1].

Une énonciation équivalente, en tant que problème d'optimization, consiste à minimiser la valeur absolue de la difference entre les deux sommes de l'équation 1 :

$$\underset{A' \subseteq A}{\textbf{minimiser}}\, z(A') = \left| \sum_{a \in A'} t(a) - \sum_{a \in A \setminus A'} t(a) \right|, \tag{2}$$

car si $\min_{A' \subseteq A} z(A') = 0$, alors la réponse au problème de décision est affirmative ; si, au contraire, $\min_{A' \subseteq A} z(A') > 0$, elle est négative.

Une solution candidate pour le problème de la partition est un sousensemble de A, qui peut être naturellement représenté par un vecteur de 0 et 1 de taille $N = \|A\|$. Cela s'adapte parfaitement à un algorithme génétique simple, qui représente une solution candidate avec une chaîne de chiffres binaires !

2 Consignes

1. Modifiez la fonction de *fitness* pour qu'elle calcule $f(A') = 1/(z(A') + 1)$, où $z(A')$ est défini comme dans l'équation 2. Pourquoi ne peut-on pas utiliser z directement comme fonction de *fitness* ?
2. Pour tester l'algorithme, vous pouvez utiliser une petite *instance* du problème, de taille $N = 19$, où $A = \{1, 2, \ldots, 19\}$ et, pour tout $a \in A$, $t(a) = a$. Pour ce problème on connaît une solution optimale, qui est

$$A' = \{1, 2, 4, 7, 8, 11, 12, 15, 16, 19\}, \quad z(A') = 0.$$

3. Pour étudier le comportement de l'algorithme, créez des autres instances du problème en générant les tailles des éléments $a \in A$ au hasard. Notez qu'il est permis d'avoir plusieurs éléments ayant la même taille.
4. Pour une instance donnée, exécutez plusieurs fois l'algorithme. Qu'est-ce que vous observez ?
5. S'il vous reste du temps, jouez avec les parametres pour étudier leur influence sur le comportement de l'algorithme. Vous pouvez aussi essayer des définitions alternative de la fonction de *fitness*, comme $f(A') = e^{-kz(A')}$, pour $k > 0$, ou $f(A') = \left[\sum_{a \in A} t(a)\right] - z(A')$.

Rendez votre code et vos observations dans un archive zippé par courriel.

Références

[1] M. R. Garey and D. S. Johnson. *Computers and Intractability : A guide to the Theory of NP-Completeness*. Freeman, New York, 1979.

[2] R. M. Karp. Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, New York, 1972.