Paysages de fitness de CSP et stratégies d'évolution

Adrien Goëffon, Éric Monfroy, Luca Davis

Univ Angers, LERIA, SFR MathSTIC, 49000 Angers, France
{adrien.goeffon,eric.monfroy}@univ-angers.fr

Mots-clés : *CSP, stratégies d'évolution, paysages de fitness.*

1 Introduction : CSP, paysages de fitness

La résolution de problèmes de satisfaction de contraintes (CSP) par des algorithmes de recherche évolutionnaires reste encore une approche marginale, bien qu'elle fut explorée avec intérêt il y a une vingtaine d'années [1, 2, 3, 4]. Chercher une solution réalisable dans un espace de recherche combinatoire de solutions quasi-exclusivement composé de solutions non réalisables rend les critères de guidage difficiles à établir, et l'usage d'algorithmes approchés vain si une solution ne peut être atteinte. Nous proposons ici d'étudier la structure des CSP dans un contexte évolutionnaire, en les modélisant en paysages de fitness combinatoires, afin d'identifier les éventuelles possibilités offertes par des stratégies de recherche évolutionnaires.

Un paysage de fitness est donné par un espace de recherche de solutions (génotypes), une relation de voisinage entre solutions (définissant une mesure de distance mutationnelle entre génotypes), et une fonction de fitness affectant une valeur d'adaptation à chaque solution. Ce triplet (codage, voisinage, évaluation) est usuellement la donnée d'un algorithme de recherche à base de voisinage (de type recherche locale), dédié à explorer le paysage de solution voisine en solution voisine, guidé par un critère basé sur l'évaluation des solutions et cherchant à maximiser l'espérance de la meilleure solution atteinte dans le budget imparti (en termes de nombre de solutions évaluées). L'analyse des paysages de fitness au moyen d'indicateurs variés permet de renseigner sur son caractère déceptif, rugueux, et plus généralement sur la pertinence d'employer un mécanisme d'apprentissage basé sur cette correspondance entre distances mutationnelles et valeurs de fitness ainsi définies.

2 Paysages de CSP

Cette première étude est consacrée à l'analyse de problèmes à domaines finis et contraintes linéaires, c'est-à-dire des problèmes de sac à dos multidimensionnels et sans fonction objectif. Une instance de problème à n variables $x_j \in D_j$ est décrite par un ensemble de m contraintes de type $\sum_{j=1}^{n} a_{ij}x_j \ \{\leq, =\} \ c_i, \ \forall i \in [1..m]$, avec $a_{ij}, c_i \in \mathbb{Z}$; on pose $D = [v_{\min}..v_{\max}]^n$, c'est-à-dire que les domaines seront identiques pour toutes les variables. Nous avons développé un générateur d'instances réalisables, et utilisons un benchmark de 24 instances construites à partir de toutes les combinaisons de paramètres possibles parmi $n \in \{20, 50, 100\}$, $m \in \{20, 50, 100, 1000\}$ et $D \in \{[1..100]^n, [-100..100]^n\}$.

Caractériser un CSP au moyen d'un paysage de fitness nécessite de définir une fonction de voisinage et une fonction de fitness. Pour le voisinage, on considérera la fonction la plus naturelle concernant les problèmes d'affectation : deux solutions sont voisines si elles ne diffèrent que par une seule valeur. La définition de la fonction de fitness, déjà investiguée pour la conception d'algorithmes de recherche locale [4], est plus délicate. L'analyse des paysages de fitness que nous proposons ici permet justement d'identifier les fonctions fournissant une pression adaptative cohérente à un processus évolutionnaire. Notons $f_1 : D^n \to [0..|m|]$ la fonction décrite au moyen d'un CSP et associant le nombre de contraintes satisfaites d'une solution, et $f_2 : D^n \to \mathbb{N}$ associant la somme des écarts aux contraintes (erreurs $|\sum_{j=1}^{n} a_{ij}x_j - c_i|$ pour les contraintes non satisfaites). Les paysages respectifs définis au moyen de ces deux fonctions de

fitness (dans notre cas à minimiser) engendrent des difficultés particulières en termes de neutralité (f_1) et de déceptivité (f_2). Nous avons ainsi analysé différentes fonctions de fitness de type $\alpha \cdot f_1 + (1-\alpha) \cdot f_2$ afin d'identifier un compromis favorable. Les différentes caractéristiques des paysages (neutralité, longueur d'autocorrélation, longueur de marche adaptative, qualité des marches adaptatives) ont indiqué que le paramètre $\alpha = 0.25$ permettait dans le cas général d'obtenir les paysages de fitness les plus structurés, avec un coefficient d'autocorrélation distance-fitness toujours supérieur à 0.9.

3 Application de stratégies d'évolution

Les stratégies d'évolution [5] forment une famille d'algorithmes évolutionnaires génériques, très utilisés dans des contextes d'optimisation boite-noire, majoritairement continue mais également discrète. Une stratégie d'évolution consiste à faire évoluer un individu unique, ou une population d'individus, au moyen de mutations et le cas échéant de croisements. Contrairement à beaucoup de métaheuristiques sophistiquées en apparence, cet algorithme de recherche très épuré permet d'explorer un espace de recherche au moyen d'un nombre réduit de paramètres dédiés à gérer la pression de mutation et de sélection, et ainsi l'équilibre à apporter entre exploration et exploitation.

Nous avons appliqué à nos instances de problème les stratégies d'évolution $(1, \lambda)$ et $(1 + \lambda)$, qui considèrent une unique configuration courante (individu), génèrent à chaque pas de recherche λ mutants, et sélectionne celui ayant la valeur de fitness la plus adaptée (ici la plus petite). La stratégie *virgule* remplace systématiquement l'individu courant par le meilleur mutant, tandis que la stratégie *plus* conditionne le remplacement à une amélioration de l'individu courant. Lorsque le budget en termes d'individus évalués est atteint, le meilleur rencontré est retourné. Enfin, nous avons expérimenté plusieurs schémas de mutations : changer la valeur d'une unique variable (la stratégie d'évolution décrit dans ce cas un algorithme de recherche locale), augmentée ou non de la variation d'un nombre aléatoire de variables supplémentaires relativement à une distribution de Poisson de paramètre $\kappa \in \{0.5, 1, 1.5, 2\}$. Un second paramètre de mutation décrit le processus de génération de la valeur de la variable mutée, soit selon une distribution uniforme dans D, soit par ajout d'une valeur aléatoire non nulle selon une loi normale de paramètres 0 et $\sigma \in \{1, 5, 10, 15\}$.

Les stratégies (1,λ) et (1+λ) ont été appliquées pour $\lambda \in \{1, 10, 20, 50, 100\}$ sur chaque instance et avec les 25 schémas de mutation. À l'exception d'une instance de problème (50 variables, 100 contraintes), la stratégie d'évolution (1,λ) avec $\lambda = 20$ et combinée à certains schémas de mutation parvient à trouver systématiquement ou quasi-systématiquement une solution réalisable, dans un budget variant de quelques dizaines à quelques centaines de milliers d'évaluations en moyenne (en fonction des schémas de mutation). Ces résultats encourageants nous laissent à penser que les algorithmes de recherche approchés, et les stratégies d'évolution en particulier, peuvent être appliqués à la résolution de CSP. Une prochaine étape consiste à mener une étude similaire sur des problèmes binaires.

Références

[1] Philippe Codogner, Daniel Diaz. *An efficient library for solving CSP with local search.* MIC 2003.

[2] BGW Craenen, and Eiben, AE and Marchiori, Elena Paolo Toth and Daniele Vigo. *How to handle constraints with evolutionary algorithms.* Practical Handbook Of Genetic Algorithms : Applications, 341–361, 2001.

[3] AE Eiben, Jano van Hemert, Elena Marchiori, AG Steenbeek. *Solving binary constraint satisfaction problems using evolutionary algorithms with an adaptive fitness function.* International Conference on Parallel Problem Solving from Nature, 201–210, 1998.

[4] Philippe Galinier, Jin-Kao Hao. *A general approach for constraint solving by local search.* Journal of Mathematical Modelling and Algorithms 3(1), 73–88, 2003.

[5] Nikolaus Hansen, Dirk V Arnold, Anne Auger. *A general approach for constraint solving by local search.* Springer handbook of computational intelligence, 871–898, 2015.