Une approche évolutionnaire multiobjectif pour la planification des cours professionnels

Mounir Hafsa[1], Pamela Wattebled[1], Julie Jacques[2] , Laetitia Jourdan[3]

[1] Mandarine Academy, Univ. Lille, CNRS, Centrale Lille, UMR 9189 - CRIStAL, F-59000 Lille, France
{mounir.hafsa,pamela.wattebled}@mandarine.academy

[2] ICL - FGES, Univ. Lille, CNRS, Centrale Lille UMR 9189 - CRIStAL, F-59000 Lille, France
julie.jacques@univ-catholille.fr

[3] Univ. Lille, CNRS, Centrale Lille, UMR 9189 - CRIStAL, F-59000 Lille, France
laetitia.jourdan@univ-lille.fr

Mots-clés : *Algorithmes évolutionnaires multiobjectifs, problème d'emploi du temps.*

1 Introduction

Mandarine Academy est une entreprise Ed-Tech qui propose des solutions centrées sur les techniques de formation innovantes (Moocs, webconférences, etc.). Les programmes de formation proposés par la société ont un emploi du temps spécifique qui garantit (le jour, l'heure, les ressources réservées, lieu) de chaque événement. La création d'un emploi du temps initial couvrant une année de formation était établie à la main et prenait en moyenne 3 à 6 semaines et 30 travailleurs. Pour répondre à cette problématique, Mandarine Academy a développé l'outil Dileap Logistic, qui gère automatiquement la logistique de la planification des formations. L'approche gloutonne utilisée dans cet outil permet de réduire le temps de planification par rapport à l'approche traditionnelle. Mais ce dernier ne respecte pas les contraintes définies et ne supporte qu'un seul objectif.

2 Description et résolution du problème

Après avoir étudié le problème et identifié chaque entité impliquée dans le processus de planification, un total de 13 contraintes dures et 5 contraintes souples sont considérés. Mandarine Academy et ses clients souhaitent plus de flexibilité en proposant des objectifs supplémentaires : **(Objectif 1)** Maximiser le nombre de sessions de formation proposées. **(Objectif 2)** Minimiser le nombre de violations des contraintes souples. **(Objectif 3)** Maximiser la charge de travail entre les formateurs. **(Objectif 4)** Equilibrer la charge de travail entre les formateurs. **(Objectif 5)** Minimiser le nombre de salles mobilisées. Parmi les métaheuristiques utilisées pour résoudre des problèmes multi objectifs, NSGA II et NSGA III ont prouvé leur efficacité [2]. Les deux algorithmes et leurs composants sont implémentés en utilisant le framework JMetalPY [1]. Une représentation directe est utilisée pour décrire chaque solution, ce qui est dans notre cas un emploi du temps respectant toutes les contraintes dures. Afin de fournir une population initiale qui respecte la structure définie et les contraintes dures, une heuristique constructive était étudiée, conceptualisée et implémentée. Deux opérateurs génétiques personnalisés inspirés par [1] sont proposés pour être comparés aux opérateurs classiques. (1) $MAPT_{co}$ (opérateur de croisement) sera comparé avec PMX. (2) $MAPT_{mo}$ (opérateur de mutation) qui intègre des connaissances spécifiques au problème sera comparé avec l'opérateur classic Swap.

1. https ://github.com/jMetal/

3 Expérimentations

Mandarine nous a fourni des données historiques (2019-2020) pour tester notre approche. 3 instances de tests différentes de complexité variable sont utilisées, (1) Small : 100 sessions de formation que nous souhaitons planifier sur 2 mois. (2) Medium : 500 sessions sur 5 mois. (3) Large : 1050 sessions sur 9 mois. Afin de comparer équitablement NSGA II et NSGA III, nous utilisons le package i-race [2] qui sert à trouver la configuration la plus performante pour un algorithme. Les objectives (O1, O2, O4) sont considérés et l'indicateur de performance est l'hypervolume. Les tests initiaux ont été conduits en utilisant les paramètres trouvés par Irace pour NSGA II sur les instances Small. La configuration élite retournée par i-race est appliqué sur les deux algorithmes avec les 3 instances (Small, Medium et Large). Elle est composée de ces paramètres : *(1) Taille de population : 75*, (2) Générations : 750 (maximum défini), (3) Opérateur de croisment : PMX, (4) Probabilité de croisement : 0.1, (5) Opérateur de mutation : $MAPT_{Mo}$, (6) probabilité de mutation : 0.7. Le tableau 1 présente les résultats expérimentaux observés pour plus de 30 exécutions et avec les objectives (O1, O2, O4). On observe que NSGA II obtient des meilleurs résultats que NSGA III en matière d'hypervolume (HV), Distance générationnelle (GD), Distance générationnelle inversée (IGD) et l'indicateur epsilon. Le temps de calcul T (exprimé en secondes) est la seule métrique où NSGA III surpasse NSGA II. Cependant, dans ce problème, T est d'une importance mineure par rapport à la qualité de nos solutions. Bien que les deux algorithmes commencent par des performances similaires, les observations d'évolution d'hypervolume sur plusieurs générations révèlent que NSGA II a pu converger plus rapidement mais que les algorithmes sont encore en pleine convergence. Dans les prochaines expérimentations, des tests avec plus de générations seront réalisés. Une perspective est également d'ajouter des instances de test, d'observer les résultats sur les autres combinaisons d'objectifs et de tester des algorithmes supplémentaires (MOEA/D ou SPEA2) dans nos futures expérimentations, afin d'être en mesure de voir des changements concernant le choix des opérateurs génétiques, leurs probabilités et la façon dont les performances de chaque algorithme changent.

TAB. 1 – Performance comparison between NSGA II and NSGA III on 3 problem instances over 30 independent runs.

	Small		**Medium**		**Large**	
	NSGA II	NSGAIII	NSGAII	NSGAIII	NSGAII	NSGAIII
HV	**0.7629** 0.0136	0.7573 0.0119	**0.6751** 0.0149	0.6514 0.0063	**0.6481** 0.0092	0.6302 0.0071
GD	**0.9827** 0.0177	1.0001 0.0081	**0.8638** 0.0488	0.9519 0.0329	**0.8204** 0.0278	0.8621 0.0273
IGD	**0.8444** 0.0496	0.9056 0.0503	**0.6754** 0.0605	0.7546 0.0562	**0.6230** 0.0385	0.6874 0.0396
EP	**0.2071** 0.0155	0.2113 0.0098	**0.2648** 0.0227	0.2968 0.0068	**0.2659** 0.0148	0.2950 0.0117
T	6730 1620	**6151** 1508	25633 5634	**25303** 5627	58041 13090	**57054** 14071

Références

[1] Zlatko Bratković, Tomislav Herman, Vjera Omrčen, Marko Čupić, and Domagoj Jakobović. University course timetabling with genetic algorithm : A laboratory excercises case study. In *European Conference on Evolutionary Computation in Combinatorial Optimization*, pages 240–251. Springer, 2009.

[2] Hisao Ishibuchi, Ryo Imada, Yu Setoguchi, and Yusuke Nojima. Performance comparison of nsga-ii and nsga-iii on various many-objective test problems. In *2016 IEEE Congress on Evolutionary Computation (CEC)*, pages 3045–3052. IEEE, 2016.

2. https ://github.com/MLopez-Ibanez/irace