A Dynamic Algorithm Framework to Automatically Design
a Multi-Objective Local Search

Camille Pageau¹, Aymeric Blot¹, Holger H. Hoos²,
Marie-Éléonore Kessaci¹, Laetitia Jourdan¹

¹ Université de Lille, CNRS, UMR 9189 CRIStAL, 59650 Villeneuve d'Ascq, France
{camille.pageau,aymeric.blot,laetitia.jourdan,mkessaci}@univ-lille.fr
² LIACS, Leiden University, Leiden, 2333 CA, The Netherlands
hh@liacs.nl

Mots-clés: multi-objectif, configuration automatique d’algorithmes

Metaheuristics are parametrized algorithms designed to solve complex optimization problems. Their parameters highly affect their performance and have to be set for each class of instance. Both offline and on-line approaches can be used to configure algorithms to be efficient. Offline approaches, also called automatic algorithm configuration (AAC), are able to handle many parameters but provide static algorithms adapted to the training instances only. On the other hand, on-line approaches provide adaptive algorithms whose parameters are modified during its execution but generally handle very few parameters. In this work, we propose a new model, called Dynamic Algorithm Framework in order to benefit from the advantages of both approaches.

1 A Dynamic Algorithm Framework

Generally, in the combinatorial optimization fields, algorithms used to solve complex problems expose a large set of tunable parameters such as numerical values and strategy components (called categorical parameters) which heavily affect their performance. While off-line approaches try to find the best configuration of the algorithm among all the possibilities, on-line approaches usually deal with a single numerical parameter or very few categorical parameters only. Therefore, in order to keep the idea of modifying the algorithm during the execution, we propose a framework that will successively use several configurations. This idea enables the use of classical AAC tools instead of on-line mechanisms.

The purpose of this work is to evaluate the efficiency of switching from a configuration of an algorithm $A$ to another configuration of the same algorithm $A$ into a same run. This work is equivalent to the automatic configuration of the proposed dynamic algorithm framework.

2 Experiments

We investigate, as first experiments, the interest of the proposed approach in order to solve the classical bi-objective permutation flowshop problem (bPFSP) with a multi-objective local search (MOLS) algorithm [2], called D-MOLS.

We consider MO-ParamILS[1], a MO-AAC configurator with two performance indicators: the unary hypervolume, a volume-based convergence performance indicator (to maximize) and the Δ spread, a distance-based distribution metric (to minimize).

We use from the classical flowshop Taillard instances [3], the ten available instances of 50 jobs and 20 machines and we generated training instances. The performance indicators – hypervolume and spread – of a configuration are the average of the measures obtained on the 15 runs.
We test two scenarios where the number of allowed modifications equals to 2 or 3 and we limit the number of different time budgets to 3. When 2 successive configurations are possible, the different settings of \((T_1, T_2)\) are \((1/4, 3/4) \cdot T\), \((1/2, 1/2) \cdot T\) and \((3/4, 1/4) \cdot T\). When 3 successive configurations are possible, the different settings of \((T_1, T_2, T_3)\) are \((1/3, 1/3, 1/3) \cdot T\), \((1/4, 1/4, 1/4) \cdot T\) and \((1/2, 1/4, 1/4) \cdot T\). Therefore, while 60 configurations are available to parametrize our classical MOLS algorithm, \(K = 2\) involves about \(1.1 \cdot 10^4\) possible configurations \((60 + 3 \times 60^2)\) for the D-MOLS and \(K = 3\) a total of about \(6.6 \cdot 10^5\) configurations \((60 + 3 \times 60^2 + 3 \times 60^3)\). We fix to 50 seconds, the stopping criterion of the D-MOLS.

![Pareto front of the final configurations of the D-MOLS for K = 2 (left) and K = 3 (right).](image)

**FIG. 1 –** Pareto front of the final configurations of the D-MOLS for \(K = 2\) (left) and \(K = 3\) (right).

First we present the results for the two proposed scenarios applied to the dynamic MOLS (D-MOLS) when \(K\) is set to 2 or 3. Secondly, we compare results of the D-MOLS to ones of the static MOLS (S-MOLS).

Figure 1 (left) shows the mean hypervolume and spread values obtained by the 12 final configurations of the D-MOLS when \(K\) is set to 2. When a D-MOLS is configured with \(k\) set to 1 (denoted D-MOLS(1)), then it is equivalent to a static MOLS while with \(k = 2\), 2 MOLS are successively applied. Among the final configurations, we find only D-MOLS(2).

Figure 1 (middle) shows the mean hypervolume and spread values obtained by the 11 final configurations of the dynamic MOLS framework when \(K\) is set to 3. Here, only D-MOLS with \(K \geq 2\) appear in the final configurations of the framework. However, D-MOLS(3) is more represented (8 vs. 3).

Since only 60 configurations are considered for S-MOLS, it is possible to compute the average performance for all these possible configurations. Among the 60 possible S-MOLS, 11 of them are in the optimal Pareto set. Figure 1 (right) shows the three Pareto fronts of S-MOLS and the two versions of D-MOLS representing 11, 12 and 11 configurations respectively. Undoubtedly, D-MOLS with a maximal number of successive configurations set to 3, gives better performance since the configured D-MOLS(3) dominates most of the others. However, a single configuration of the S-MOLS dominates the others, this can be explained by the solution rarity.

**Références**

