Two-stage stochastic/robust scheduling using permutable operation groups

Louis Rivière, Christian Artigues, Hélène Fargier

▶ **To cite this version:**

Louis Rivière, Christian Artigues, Hélène Fargier. Two-stage stochastic/robust scheduling using permutable operation groups. 23ème Congrès annuel de la Société Française de Recherche Opérationnelle et d'Aide à la Décision (ROADEF 2022), Institut National des Sciences Appliquées : INSA Lyon; Université de Lyon, Feb 2022, Villeurbanne - Lyon, France. hal-03595429

HAL Id: hal-03595429
https://hal.science/hal-03595429

Submitted on 3 Mar 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Two-stage stochastic/robust scheduling using permutable operation groups

Louis Riviere[1,2,3], Christian Artigues[1,2] and Hélène Fargier[1,3]

[1] Artificial and Natural Intelligence Toulouse Institute, Université de Toulouse, France
[2] LAAS-CNRS, Université de Toulouse, CNRS, UPS, Toulouse, France
[3] IRIT, Université de Toulouse, CNRS, UPS, Toulouse, France

Mots-clés : *Stochastic optimisation, permutable operation groups.*

1 Introduction

This paper considers a single machine scheduling problem with release dates, due dates, precedence constraints and aiming for the minimization of the maximum lateness or the sum of completion times over a sampled set of release dates scenarios, either in a stochastic or robust setting. The standard 2-stage approach for stochastic and robust scheduling on a single machine considers first-stage decisions that output a full ordering (sequence) of the jobs (J-SEQ) on the machine and second-stage decisions that set the job start times in accordance with the information of a given scenario. In this paper, we study the performances of an alternative 2-stage solution method based on groups of permutable jobs. The method, initially presented in [1], considers first-stage decisions that output only a partial ordering in the form of a sequence of groups of permutable jobs (G-SEQ). Given a scenario, the second stage policy orders the jobs inside each group according to the earliest realized release date first heuristic.

We introduce a constraint programming approach to compute an optimal G-SEQ solution but show that in a limited time, it provides poor quality solutions on the largest instances compared to a standard J-SEQ solution approach. We then design several heuristics that use a portion of the time limit to obtain good quality J-SEQ starting solutions and then switch to G-SEQ solutions via greedy or local search. The best G-SEQ heuristics outperform all of the standard J-SEQ approaches under the same time limit.

2 Problem definition

The problem consists in scheduling a set of jobs N given uncertain release dates described by a set of discrete scenarios S. We define a problem with $|N|$ jobs and $|S|$ scenarios as a tuple $\mathcal{P} = (P, R, D, E, \sigma, \gamma)$ where $p_i \in P$ is the duration of job i, $r_i^s \in R$ is the release date of job i in scenario s, $d_i \in D$ is the due date of job i, E is the set of precedence constraints. Parameter $\sigma \in \{max, avg\}$ is the objective aggregator amongst scenarios, and $\gamma \in \{L_{max}, \sum C_i\}$ is the objective type.

We consider two different first stage decision settings for this problem : respectively job sequences (J-SEQ) and sequences of groups of permutable jobs (G-SEQ). In the former setting are the usual sequences : a total ordering of the jobs. Given a J-SEQ and a scenario, the best associated start times are the left-shifted ones as semi-active schedules are dominant for our objectives. In the latter setting, sequences of groups of permutable jobs, are ordered partitions of the set of jobs. Because jobs within groups are not ordered, a G-SEQ represents a set of job sequences. Given a G-SEQ and a scenario, we use the "Earliest release date first" heuristic to compute a full job sequence, and then the corresponding left-shifted schedule. G-SEQs are in theory better than J-SEQs because they generalize them, but due to the much larger search space, in a limited time, it is not easy to predict which approach will give the best results.

		G-SEQ					J-SEQ			
obj	N	CP	Warm-CP	Taboo	GA	Greedy	CP	Taboo	GA	Greedy
avg	10	**1.0000**	**1.0000**	0.9961	0.9997	0.9998	0.9537	0.9537	0.9537	0.9537
L_{max}	20	0.9397	0.9993	0.9910	**0.9967**	0.9899	0.9479	0.9473	0.9476	0.9450
	50	0.5717	0.9783	0.9934	**0.9981**	0.7561	0.9648	0.9620	0.9627	0.7439
	100	0.3983	0.9639	0.9876	**0.9931**	0.5732	0.9853	0.9726	0.9720	0.5680
avg	10	0.9980	0.9997	0.9983	0.9997	**1.0000**	0.9793	0.9792	0.9793	0.9793
$\sum C_i$	20	0.9611	0.9923	0.9949	**0.9984**	0.9968	0.9748	0.9743	0.9749	0.9749
	50	0.8690	0.9813	0.9946	**0.9988**	0.9596	0.9786	0.9784	0.9798	0.9590
	100	0.8394	0.9575	0.9712	**0.9988**	0.8876	0.9726	0.9631	0.9725	0.9060
max	10	**1.0000**	**1.0000**	0.9934	0.9996	**1.0000**	0.9614	0.9614	0.9614	0.9614
L_{max}	20	0.9913	**1.0000**	0.9885	0.9971	0.9921	0.9624	0.9624	0.9624	0.9591
	50	0.6950	0.9958	0.9923	**0.9968**	0.7709	0.9786	0.9785	0.9785	0.7664
	100	0.4728	0.9896	0.9945	**0.9954**	0.6142	0.9934	0.9898	0.9893	0.6069
max	10	**1.0000**	**1.0000**	0.9968	0.9993	0.9999	0.9771	0.9771	0.9771	0.9771
$\sum C_i$	20	0.9600	0.9935	0.9911	**0.9969**	0.9955	0.9749	0.9741	0.9745	0.9748
	50	0.8496	0.9856	0.9921	**0.9970**	0.9543	0.9836	0.9814	0.9822	0.9604
	100	0.8154	0.9653	0.9711	**0.9943**	0.8796	0.9902	0.9660	0.9735	0.8971

TAB. 1 – Performance comparison for all solvers (best score/solver score)

3 Solvers

We compare on the one hand G-SEQ solvers (* denotes a starting J-SEQ solution) :

— CP : Constraint programming model for G-SEQ and J-SEQ running on CP Optimizer.

— Warm-CP* : G-SEQ CP model guided with a starting J-SEQ solution.

— Greedy : A simple greedy heuristic, after shuffling the jobs, at each step it inserts the jobs in an existing group or between existing groups to maximize the objective.

— Taboo* : A taboo list algorithm based on the neighborhood defined by moving one job from a group to another group or between existing groups.

— GA* : A genetic algorithm in which the crossover operator is an adaptation of the one-point crossover to G-SEQs.

And on the other hand we also propose GA*, Greedy and Taboo* heuristics producing standard J-SEQ solutions based on similar neighborhoods to their G-SEQ counterparts.

4 Experimental results and comparisons

The solvers have been tested on a custom benchmark of instances generated using several parameters : the number of jobs, the number of scenarios, the scenario variability, and the density of the precedence graph. The solvers were run with a 5 minutes timeout. For warm start solvers the output of the J-SEQ CP model after 1 minute was used as starting point for the solver for the remaining 4 minutes.

Table 1 shows, for different objectives and number of tasks, the average ratio of the best score reached by any solver to the score reached by the solver on the same instance. Results show that even though for small problem, the CP G-SEQ solver manages to solve most instances optimally, when the number of task increases, CP J-SEQ solver performs better than the CP G-SEQ solver, especially for L_{max} objective. However, we can see that several solvers (especially Taboo and AG heuristics) are able to improve the average performance of the input sequence more than J-SEQ solver is able to within the same time ; showcasing that it can be beneficial to compute G-SEQ solutions, even within limited time.

Références

[1] Artigues C., Billaut J.C., Cheref A., Mebarki N., Yahouni Z. Robust Machine Scheduling Based on Group of Permutable Jobs *Robustness analysis in decision aiding Optimization, and Analytics*, Springer, 191–220, 2016.