Limited discrepancy heuristic for scheduling problems with time lags

Wafa Karoui[1,2,3], Marie-José Huguet[1,2], Pierre Lopez[1,2] and Mohamed Haouari[3]

[1] CNRS ; LAAS ; 7 avenue du colonel Roche, F-31077 Toulouse, France
[2] Université de Toulouse ; UPS, INSA, INP, ISAE ; LAAS ; F-31077 Toulouse, France
[3] Unité de recherche ROI ; Ecole Polytechnique de Tunisie, 2078 La Marsa, Tunisie
{wakaroui,huguet,lopez}@laas.f, mohamed.haouari@ept.rnu.tn

Keywords: Scheduling, jobshop, flowshop, time lags, discrepancy.

1 Introduction

This paper addresses the jobshop and the flowshop scheduling problems with minimal and maximal time lags. The objective is to find a schedule that minimizes the makespan. Different definitions can be associated to time lags constraints. Initially, Mitten (Mitten 1958) proposes this concept. (Dell'amico 1996) defines it as time between the end of one operation and the start of another. In our case, we speak about two extra constraints added to the jobshop and flowshop problems, linking successive operations of a same job. Time between these operations is bounded by minimal and maximal time lags. These problems can be considered as a generalization of basic problems without time lags (NP-difficult in the strong sense). With time lags, problems become at least as difficult as the basic ones. Few methods have been used to solve this kind of problems. (Caumond *et al.* 2008) proposed a memetic algorithm which obtained good results on jobshop instances with null minimal and maximal time lags (no-wait problems). (Huguet *et al.* 2010) also studied this problem including generalized resource constraint propagation rules and branch-and-bound.

In this abstract, we propose adaptations of Climbing Discrepancy Search (CDS) (Milano and Roli 2002), to solve scheduling problems with time lags. The remainder of this extended abstract is organized as follows. Section 2 introduces the principle of CDS and proposed adaptations for the studied problems. Section 3 synthesizes carried out experiments to evaluate CDS performance. Finally, Section 4 provides conclusions and further works.

2 Discrepancy and learning for problems with time lags

To solve problems with time lags, we propose a variant of Climbing Discrepancy Search method, a tree search principle for optimization based on discrepancy. This method starts from an initial solution proposed by a given heuristic and tries to improve it by increasing step by step the number of times we do not follow this solution (discrepancy). It then builds a neighborhood around this initial solution. Nodes with discrepancy equal to 1 are first explored then those having a discrepancy equal to 2, and so on. When a leaf with improved value of the objective function is found, the reference solution is updated, the number of discrepancy is reset to 0, and the process for exploring the neighborhood is restarted. To limit the tree search expansion, we put a stop condition as a timeout on the CPU time. To problems under study, we propose various parameter settings: discrepancy position in the search tree, heuristics to generate the initial solution, learning mechanisms based on weights associated to jobs. The discrepancies counting is binary: the heuristic choice corresponds to zero discrepancies, all the other choices correspond to one discrepancy.

2.1 Discrepancy position

We experiment to diverge alternatively, first at the top of the search tree, or first at its bottom. We try also to diverge only in a part of the tree, for example at the top, and to visit the other part without discrepancies at all.

2.2 Heuristic to generate the initial solution

The heuristic selects a job and places all its operations in the order of their definition (routing). In the sequel, we call D the duration of a job equal to the sum of all its operation durations and DTL the sum of all its time lags durations. To obtain an initial solution, we can consider heuristics which sort jobs in the lexicographical order, as in (Huguet *et al.* 2010), or in the ascending or descending order of D, DTL, D+DTL and D/DTL. For a given problem, it is obvious that if we start the search from a good solution, we have more chance to get more improvements.

2.3 Learning based on weights on jobs

Learning can guide the method and improve it. To adjust the proposed method to problems under study, we associate a weight to each job. At the beginning, all weights are identical. We can increase a job weight in different ways. We studied three cases as shown in Figure 1 where operations of job J_3 cannot be placed in their first slack period on the associated machine.

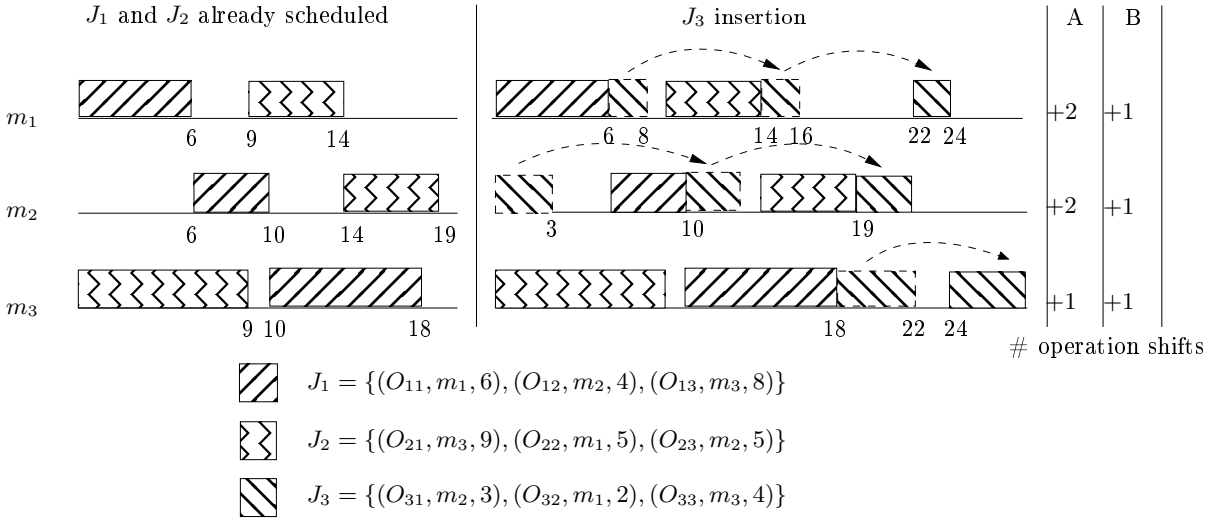

Fig. 1. Different ways to increment job weights

– Case A) The job weight is increased every time one of its operations is not inserted in some slack period. In the example, the weight of J_3 is then 5.

- Case B) The job weight is increased every time one of its operations is not inserted in the first slack period on one of its associated machine. As we can see in the example, we increment the weight at most one time per machine (or operation). The maximum factor to get on a considered operation is equal to the number of machines (or operations). In the example, the weight of J_3 is then 3.
- Case C) The job weight is increased every time one or more operations of this job are not placed in their first slack periods on the associated machine. In the example, the weight of J_3 is then 1. As we can see in the example, the weight is increased at most one time for the same job.

Anyway, in addition of the manner to increase weights, we have to choose a way to count them. For example, we can consider the sum of all weights obtained by the job during the iteration (denoted in the following by *Sum*). Or the maximum of all its weights (denoted by *Max*). In the next iterations, obtained weights are integrated in the basic heuristic to choose the job to schedule first. For example, if the heuristic is based on the duration D of jobs, the heuristic using weights can be based on $D/weights$.

3 Computational results

We experiment our propositions on the data set of classical instances of scheduling problems proposed in (Lacomme 2009). For jobshops, we consider the Lawrence's instances $\{laX\}_{X=1..20}$ in addition to Fisher and Thompson's instances, $ft06$ and $ft10$. For flowshops, we consider Carlier's instances $\{CarX\}_{X=5..8}$. The data set contains instances created from classical instances with minimal and maximal time lags generated with a minimal time lag ($TLmin$) equal to 0 and a maximal time lag ($TLmax$) equal to 0, 0.25, 0.5, 1, 2, 3, 5, and 10. Comparisons are done *vs.* results obtained by (Caumond *et al.* 2008) which consider only the $ft06$ with $TLmax$ of 0, 0.5, 1 and 2, $\{LaX\}_{X=1..20}$ with $TLmax$ equal to 0, $\{LaX\}_{X=1..5}$ with $TLmax$ equal to 0.5, 1 and 2, and $\{LaX\}_{X=6..8}$ with $TLmax$ equal to 0.5, 1, 2 and 10, in addition to $\{CarX\}_{X=5..8}$ with $TLmax$ equal to 0, 0.5, 1 and 2. Comparisons are also done *vs.* results obtained with ILOG-Scheduler for all considered instances.

The tests about the heuristic to generate the initial solution show that *descending D* gives the best results. In Table 1, we can observe the number of instances on which every heuristic is the best.

Table 1. Comparison of heuristics to generate an initial solution

Order	D	DTL	D+DTL	D/DTL	Lexicographical
Descending	**55**	51	**55**	29	20
Ascending	5	12	5	22	

For other tests, timeout is of 200 seconds. In general, the best known solutions (*BKSs*) are divided between (Caumond *et al.* 2008), ILOG-Scheduler and our propositions without any regularity. Nevertheless, we claim the following. For the no-wait problems, (Caumond *et al.* 2008) obtain the best results. For other instances, ILOG-Scheduler provides the best results except for cases referred in Table 2 where our propositions have the best results. In Table 2, *WF* refers to the version of CDS without weights which diverges at the top first. *B-Sum*, respectively *B-Max*, refers to *case B* for weighting jobs, as presented below,

Table 2. Obtained results on some instances

Instance	*TLmax*	*ILOG-Scheduler*	*WF*	*B-Sum*	*B-Max*
la11	0.25	2058	**1861**	1965	1965
	0.5	1945	**1874**	**1874**	**1874**
la12	0.25	1710	1682	1671	**1656**
la13	0.25	1906	**1897**	1892	1892
	0.5	1804	**1787**	1808	1808
la14	0.25	2143	**1823**	2042	2042
	0.5	2067	1964	**1953**	**1953**
	1	1976	1772	**1762**	**1762**
	2	1976	**1612**	1660	1660
	3	1695	1567	**1542**	**1542**
	5	1695	**1452**	1477	1477
la15	0.25	2371	2084	**2043**	**2043**
	0.5	2217	2118	**1910**	**1910**
la17	0.25	1455	**1410**	1427	1460

associated to the counting way *Sum*, respectively *Max*. Case B seems to be better than other cases of weighting jobs on considered instances.

4 Conclusions and further works

In this paper, a Climbing Discrepancy Search (CDS) method is proposed to solve job-shop and flowshop scheduling problems with time lags. We studied various parameter settings for the proposed method, such as discrepancy positions, heuristic to generate the initial solution, and learning mechanisms based on weights associated to jobs. Proposed variants were tested on known benchmarks in the literature. The obtained results show that we have to study variants of CDS associated to classical scheduling techniques as heuristic insertion to determine upper bounds, and ressource constraint propagation rules adaptation for lower bounds.

References

Caumond, A., Lacomme, P., and Tchernev, N., 2008. "A memetic algorithm for the job-shop with time-lags". *Computers and Operations Research 35*, p. 2331-2356.

Dell'amico, M., 1996, "Shop problems with two machines and time lags". *Operations Research 44(5)*, p. 777-787.

Huguet, M.-J., Artigues, C., Dugas, M. and Lopez, P., 2010. "Generalized Constraint Propagation for Solving Job Shop Problems with time lags". *PMS'10 (submitted)*.

Lacomme, P., 2009. http://www.isima.fr/~lacomme/Job_Shop_TL.html

Mitten, L.G., 1958. "Sequencing n jobs on two machines with arbitrary time lags". *Management Science 5*, p. 293-298.

Milano, M. and Roli, A., 2002. "On the relation between complete and incomplete search: an informal discussion". *Proceedings CPAIOR'02*, Le Croisic, France, p. 237-250.