

Worst-case evaluation in disjunctive scheduling with precedence constraints: a general formulation and a polynomial algorithm for the flow-shop case

Mohamed Ali Aloulou¹ and Christian Artigues^{2,3}

¹ LAMSADE – Université Paris Dauphine
Place du Maréchal de Lattre de Tassigny, 75775 Paris Cedex 16, France
tel: +33 1 44 05 46 53 - fax: +33 1 44 05 40 91
aloulou@lamsade.dauphine.fr

² LIA – Université d'Avignon
339 chemin meinajariés, Agroparc, BP 1228, 84911 Avignon Cedex 9, France

³ CRT – Université de Montréal
C.P. 6128, succursale Centre-ville Montréal, QC H3C 3J7 Canada
tel: +1 514 343 7370 - fax: +1 514 343 7121
christian.artigues@univ-avignon.fr

1 Introduction and related work

We consider a general non preemptive disjunctive problem in which a set of operations has to be scheduled on a set of machines, each operation requiring a single machine during its execution and each machine being able to process only one operation simultaneously. The operations are linked by simple precedence constraints that do not necessarily form chains. Such a model encompasses the standard flow-shop and job-shop models.

An important issue in scheduling concerns the support provided to the end-user(s) on line schedule execution after the off line scheduling phase which consists in providing an optimal or suboptimal schedule.

In disjunctive scheduling, as soon as a regular minmax objective function is considered, the support for on line scheduling often lies in providing for each machine the mandatory sequence of operations, and for each operation an earliest and a latest start time yielding operation slacks. The sequences and the time windows are such that scheduling the operations in the predetermined order and inside their time windows is feasible and keeps the objective function in a range of acceptable values. Such a flexibility provided to the end-user is referred to as temporal flexibility. This paper addresses the problem of providing more flexibility than the classical temporal one in a disjunctive scheduling problem where the objective is to minimize a regular minmax objective function. As already considered in previous studies [ER89, BR96, WBS99, AKP04, AP05, ABE05, EBS05], this can be achieved by defining only a partial order of the operations on each machines, leaving to the end-user the possibility to make the remaining sequencing decisions. This is the principle of the groups of permutable operations model that has been studied by several authors [ER89, BR96, WBS99, ABE05, EBS05]. The group model sets restrictions on the proposed partial orders that we relax in this paper. Indeed we represent the

partial orders through general precedence constraints between operations of the same machine, that have not to be distinguished from the structural precedence constraints.

Providing a partial solution through partial orders is useful in practice only if it can be assorted with an evaluation of the complete solutions that can be obtained by extension. More precisely, given a reasonable decision policy followed by the decision maker, the following questions have to be answered. Do there remain decisions (following the decision policy) leading to a feasible schedule? What is the worst objective function value reachable by the remaining set of decisions? Answering these questions provides a performance guarantee if the given on line policy is followed. We assume that the decision maker follows a semi-active schedule policy to extend the proposed partial orders. In case of a minmax objective function of the completion times, the worst objective function value can be determined by computing the worst-case completion time of each operation separately.

We show that the problem of computing the worst completion time of an operation in all feasible semi-active schedules of a disjunctive problem can be done by finding an elementary longest path in the disjunctive graph with additional constraints. This gives a general framework integrating previous studies [ER89, BR96, WBS99, AKP04, ABE05, EBS05].

In the special case of the flow-shop problem with release dates and additional precedence constraints, we give a polynomial algorithm that computes the maximal completion times of all operations in all feasible semi-active schedules. These results generalize the ones previously established for the single machine version of this problem [AKP04].

In section 2 we formulate the problem. In section 3, the polynomial algorithm for the flow-shop is given.

2 A general formulation for the worst case evaluation in disjunctive problems with a minmax regular objective function

$N = \{1, \dots, n\}$ is a set of operations. m_i, p_i and r_i denote the machine, processing time and release date of operation i , respectively. Each operation is associated with a non-decreasing cost function $f_i(C_i)$ of its completion time C_i . We introduce two dummy operations 0 and $n+1$ such that $p_0 = p_{n+1} = 0$. A disjunctive problem is represented by a disjunctive graph $\mathcal{G} = (V, C, D)$ [RS64] where $V = N \cup \{0, n+1\}$. C is the set of conjunctive arcs representing the precedence constraints between the operations. Each conjunctive arc (i, j) is valued by p_i . D is a set of pairs of disjunctive arcs defined as $D = \{(i, j), (j, i) \mid i \neq j \text{ and } m_i = m_j\}$. In the remaining a pair of disjunctive arcs $\{(i, j), (j, i)\}$ is called a disjunction and denoted by e_{ij} . Note that such a definition of a scheduling problem covers a wide variety of problems including single machine, flow shop and job shop problems with release dates and due dates. \mathcal{D} denotes the set of all disjunctive arcs, i.e. $\mathcal{D} = \cup_{e_{ij} \in D} e_{ij}$. A selection π is a (possibly empty) set of arcs such that $\pi \subseteq \mathcal{D}$ and $|\pi \cap e_{ij}| \leq 1$, for all $e_{ij} \in \mathcal{D}$. Let $D(\pi) = \{e_{ij} \in D \mid e_{ij} \cap \pi = \emptyset\}$. A selection is complete if $D(\pi) = \emptyset$, otherwise it is partial. The disjunctive graph issued from a complete or partial selection π is denoted by $\mathcal{G}(\pi) = (V, C \cup \pi, D(\pi))$. Given a set of arcs E , let $G(E)$ denote graph $(V, C \cup E)$. A complete selection π is feasible if the graph $G(\pi) = (V, C \cup \pi)$ is acyclic. The completion time $C_i(\pi)$ of any operation $i \in N$ in the semi-active schedule derived from the complete feasible selection π is equal to the length of the longest path in $G(\pi)$ from 0 to i plus p_i . Let Π denote the set of feasible complete selections. The objective of the classical scheduling problem is to find a complete feasible selection $\pi \in \Pi$ such that a *regular* objective function $F(C_1(\pi), \dots, C_n(\pi)) = \max_{i=1, \dots, n} f_i(C_i(\pi))$ is minimized.

Here, we assume the decision maker makes on line the remaining sequencing decisions on each machine following a semi-active policy until obtaining a complete selection π . A feasible

semi-active schedule can be obtained by a list scheduling algorithm as soon as C is acyclic : sort the operations in a non decreasing order of their level in $G = (V, C)$ then sequence as soon as possible on its machine each operation according to this order. The problem tackled in this paper is the following problem (SP) : Given a disjunctive graph $\mathcal{G} = (V, C, D)$, what is the worst case objective function value of the feasible semi-active schedules, i.e compute $\max_{\pi \in \Pi} F(C_1(\pi), \dots, C_n(\pi))$?

To illustrate this problem, consider the following 2-machine and 4-job flow shop problem. This gives 8 operations and the flow-shop context sets $m_1 = m_3 = m_5 = m_7 = 1$ and $m_2 = m_4 = m_6 = m_8 = 2$. Furthermore, we have $p_1 = 1, p_2 = 6, p_3 = 2, p_4 = 5, p_5 = 4, p_6 = 6, p_7 = 6$ and $p_8 = 1$. All release dates are equal to 0 except for $r_5 = 2$. All objective functions are the completion times of the activities. Let us consider additional precedence constraints $\{(1, 3), (1, 5), (1, 7), (3, 7), (2, 4), (2, 6), (6, 8)\}$. We obtain the disjunctive graph \mathcal{G} displayed in Figure 2.

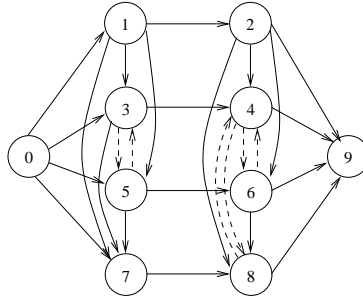


Figure 1: The disjunctive graph for a partial selection

The precedence constraints restrict the possible sequences to $(1, 3, 5, 7)$ and $(1, 5, 3, 7)$ on machine 1 and $(2, 4, 6, 8)$, $(2, 6, 4, 8)$ and $(2, 6, 8, 4)$ on machine 2. Note that such restriction cannot be modeled by the group of permutable operation representation used in [ER89, BR96, WBS99, AKP04, EBS05]. The solution of problem SP is 20 which is the worst-case makespan value of the 6 semi-active schedules. Note that the optimal makespan of the flow-shop problem is 19. Let \hat{C}_i denote the worst case completion time of i , i.e. $\hat{C}_i = \max_{\pi \in \Pi} C_i(\pi)$.

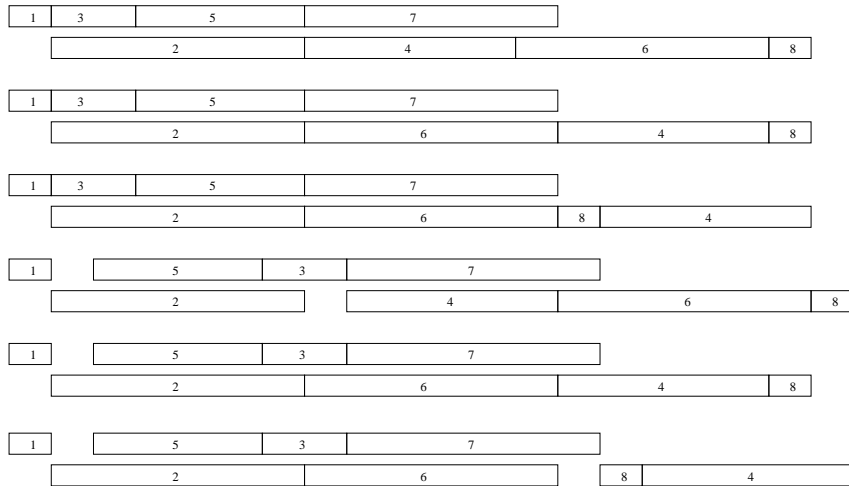


Figure 2: Sequences and semi-active schedules compatible with the partial selection

Computing \hat{C}_i , for each $i \in N$ solves problem SP since the objective function is a minmax function of non decreasing functions of the completion times. Recall that \mathcal{D} is the set of all disjunctive arcs and let us now consider the following Constrained Longest Path problem (CLP): Given a disjunctive graph $\mathcal{G} = (V, C, D)$ and an operation i , compute the longest elementary path $L(0, i)$ from 0 to i in $G(\mathcal{D}) = (V, C \cup \mathcal{D})$ such that $G(L) = (V, C \cup L(0, i))$ is acyclic.

We can show that \hat{C}_i is precisely the length of $L(0, i)$. The sketch of the proof is as follows: We can show that \hat{C}_i is the length of an elementary path l from 0 to i in $G(\mathcal{D})$ and that $G(l)$ is acyclic. We can also show that any elementary path L from 0 to i in $G(\mathcal{D})$ verifying $G(L)$ is acyclic is such that there exists a feasible complete selection $\pi \in \Pi$ verifying $C \cup L \subseteq C \cup \pi$.

Note that in the general case, problem CLP may be not easy to solve since it admits as a particular case the search for the longest elementary path in a graph with positive length cycles.

3 A polynomial algorithm for solving the worst-case evaluation problem in a flow shop with precedence constraints

We now consider a flow shop problem with operation release dates and additional precedence constraints appearing only between operations scheduled on the same machine, as in the considered example. For an operation j let j^- denote its predecessor on the job. We assume that if j is the first operation of its job, then j^- is a dummy operation denoted j^0 . Let P_j denote the set of operations that must be scheduled before j on machine m_j . Let I_j denote the set operations of machine m_j that are not linked to j with a precedence constraint. We have:

$$P_j = \{i | m_i = m_j \text{ and there is a path from } i \text{ to } j \text{ in } G=(V, C)\}$$

$$I_j = \{i \neq j | m_i = m_j, i \notin P_j \text{ and } j \notin P_i\}$$

Let us define $\hat{C}_{j^0} = r_j$. We prove that the worst case completion time of operation j is given by

$$\hat{C}_j = p_j + \max \begin{cases} r_j, & (a) \\ \hat{C}_{j^-}, & (b) \\ \max_{i \in I_j \cup P_j} \{ \max(r_i, \hat{C}_{i^-}) + \sum_{x \in I_j \cup P_j \setminus P_i} p_x \} & (c) \end{cases}$$

This states that the worst case completion time of an operation j is either set by (a) its release date, (b) by the worst case completion time of its job predecessor or (c) by the worst case start time of an operation i scheduled before j on m_j , plus the sum of processing time of all operations that can be scheduled between i and j (set $I_j \cup P_j \setminus P_i$). This can be shown by a recursion argument on the machine indices, starting with the operations scheduled on machine 1.

Once sets P_j and I_j are built for each operation j , all worst-case completion times can be computed trivially in $O(m\nu^2)$ where $\nu = n/m$ is the number of jobs.

In the illustrative example the worst case completion times are given (in the order of their computation) by $\hat{C}_1 = r_1 + p_1 = 1$ (a), $\hat{C}_3 = r_5 + p_5 + p_3 = 8$ (c), $\hat{C}_5 = r_1 + p_1 + p_3 + p_5 = 7$ (c), $\hat{C}_7 = r_5 + p_5 + p_3 + p_7 = 14$ (c), $\hat{C}_2 = \hat{C}_1 + p_2 = 7$ (b), $\hat{C}_4 = \hat{C}_7 + p_8 + p_4 = 20$ (c), $\hat{C}_6 = \hat{C}_3 + p_4 + p_6 = 19$ (c), $\hat{C}_8 = \hat{C}_3 + p_4 + p_6 + p_8 = 20$ (c)

We should now focus on extension of this approach to more general problems. Unfortunately, the extension this approach to the job shop is not trivial. A way to solve it is to study the complexity of the problem of finding the constrained longest elementary path in the disjunctive graph of the job-shop problem, which has been proven in this paper to be polynomially solvable in the case of flow-shop graph.

References

- [ABE05] C. Artigues, J.C. Billaut, and C. Esswein. Maximization of solution flexibility for robust shop scheduling. *European Journal of Operational Research*, 165(2):314–328, 2005.
- [AKP04] M. Aloulou, M. Kovalyov, and M.C. Portmann. Maximization in single machine scheduling. *Annals of Operations Research*, 129:21–32, 2004.
- [AP05] M.A. Aloulou and M.-C. Portmann. An efficient proactive-reactive scheduling approach to hedge against shop floor disturbance. In G. Kendall, E.K. Burke, S. Petrovic, and M. Gendreau, editors, *Multidisciplinary Scheduling: Theory and Applications 1st International Conference, MISTA '03 Nottingham, UK, 13-15 August 2003. Selected Papers*. Elsevier, 2005. to appear.
- [BR96] J.C. Billaut and F. Roubellat. A new method for workshop real time scheduling. *International Journal of Production Research*, 34(6):1555–1579, 1996.
- [EBS05] C. Esswein, J.C. Billaut, and V. Strusevich. Two-machine shop scheduling: Compromise between flexibility and makespan value. *European Journal of Operational Research*, 167(3):796–809, 2005.
- [ER89] J. Erschler and F. Roubellat. An approach for real time scheduling for activities with time and resource constraints. In R. Slowinski and J. Weglarz, editors, *Advances in project scheduling*. Elsevier, 1989.
- [RS64] B. Roy and B. Sussmann. Les problèmes d’ordonnancement avec contraintes disjonctives, 1964. D.S. vol. 9, SEMA, Paris, France.
- [WBS99] S.D. Wu, E.S. Byeon, and R.H. Storer. A graph-theoretic decomposition of the job-shop scheduling problem to achieve scheduling robustness. *Operations Research*, 47(1):113–124, 1999.