Job Shop Scheduling

(extended abstract)

J.K. Lenstra *

1. Introduction

The job shop scheduling problem is described as follows. Given are a set of jobs and a set of machines. Each machine can handle at most one job at a time. Each job consists of a chain of operations, each of which needs to be processed during an uninterrupted time period of a given length on a given machine. The purpose is to find a schedule, i.e., an allocation of the operations to time intervals on the machines, that has minimum length.

The problem allows a number of relatively straightforward mathematical formulations. In addition, it is extremely difficult to solve to optimality. This is witnessed by the fact that a problem instance with only ten jobs, ten machines and one hundred operations, published in 1963, remained unresolved until 1986.

We formulate mathematical models of the job shop scheduling problem in Section 2 and review results on its computational complexity in Section 3. Optimization and approximation algorithms are presented in Sections 4 and 5, respectively; their computational merits are discussed in Section 6.

2. Model Formulation

Given are a finite set $\mathcal{J}$ of jobs, a finite set $\mathcal{M}$ of machines, and a finite set $\mathcal{O}$ of operations. For each operation $i \in \mathcal{O}$, there is a job $J_i \in \mathcal{J}$ to which it belongs, a machine $M_i \in \mathcal{M}$ on which it requires processing, and a processing time $p_i \in \mathbb{N}$. There is a binary relation $\rightarrow$ on $\mathcal{O}$ that decomposes $\mathcal{O}$ into chains corresponding to the jobs; more specifically, if $i \rightarrow j$, then $J_i = J_j$ and there is no $k \notin \{i, j\}$ with $i \rightarrow k$ or $k \rightarrow j$. The problem is to find a starting time S_i for each operation $i \in \mathcal{O}$ such that

*Eindhoven University of Technology, Department of Mathematics and Computing Science, P.O. Box 513, 5600 MB Eindhoven, Netherlands.

NATO ASI Series, Vol. F 82
Combinatorial Optimization
Edited by M. Akgül et al.

$$\max_{i \in \mathcal{O}} S_i + p_i \tag{1}$$

is minimized subject to :

$$\begin{aligned} S_i &\geq 0 && \text{for } i \in \mathcal{O}, && (2)\\ S_j - S_i &\geq p_i && \text{whenever } i \rightarrow j, i, j \in \mathcal{O}, && (3)\\ S_j - S_i \geq p_i \ \bigvee\ & S_i - S_j \geq p_j && \text{whenever } M_i = M_j, i, j \in \mathcal{O}. && (4) \end{aligned}$$

The objective function (1) represents the schedule length, in view of (2). The conditions (3) are the job precedence constraints. The conditions (4) represent the machine capacity constraints, which make the problem NP-hard.

To obtain an *integer programming* formulation, we choose an upper bound T on the optimum and introduce a 0-1 variable y_{ij} for each ordered pair (i,j) with $M_i = M_j$, where $y_{ij} = 0$ ($y_{ij} = 1$) corresponds to $S_j - S_i \geq p_i$ ($S_i - S_j \geq p_j$). We now replace (4) by

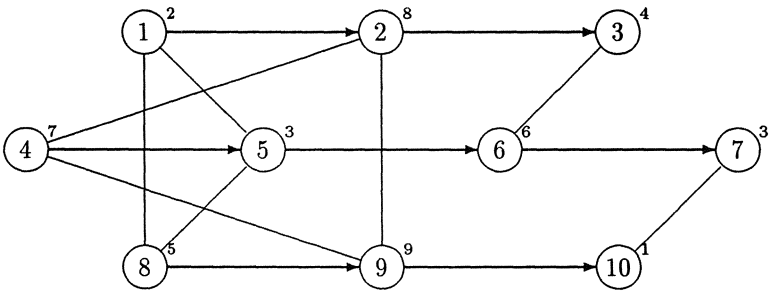
$$\left.\begin{aligned} y_{ij} &\in \{0,1\}\\ y_{ij} + y_{ji} &= 1\\ S_i + p_i - S_j - T y_{ij} &\leq 0 \end{aligned}\right\} \quad \text{whenever } M_i = M_j, i, j \in \mathcal{O}. \tag{5}$$

This formulation is closely related to the *disjunctive graph* model of Roy and Sussmann [28]. The disjunctive graph $G = (\mathcal{O}, A, E)$ has a node set $\mathcal{O}$, an arc set $A = \{(i,j) \mid i \rightarrow j\}$, and an edge set $E = \{\{i,j\} \mid M_i = M_j\}$; a weight p_i is associated with each node i. The (directed) arcs represent the job precedence constraints; the (undirected) edges represent the machine capacity constraints. The basic scheduling decision is to impose an ordering on a pair of operations $\{i,j\}$ on the same machine. In the integer program, this corresponds to setting $y_{ij} = 0$ and $y_{ji} = 1$ or $y_{ij} = 1$ and $y_{ji} = 0$; in the disjunctive graph, it corresponds to orienting the edge in question, in one way or the other. Thus, there exists a one-to-one correspondence between feasible values of the y_{ij} in {(1) , (2), (3), (5)} and orientations of the edges in E for which the resulting digraph is acyclic. Given any such orientation, we can determine feasible starting times by setting each S_i equal to the weight of a maximum-weight path in the digraph finishing at i minus p_i; the objective value is equal to the maximum path weight in the digraph. The problem is now to find an orientation of the edges in E that minimizes the maximum path weight. We refer to Figure 1 for an example.

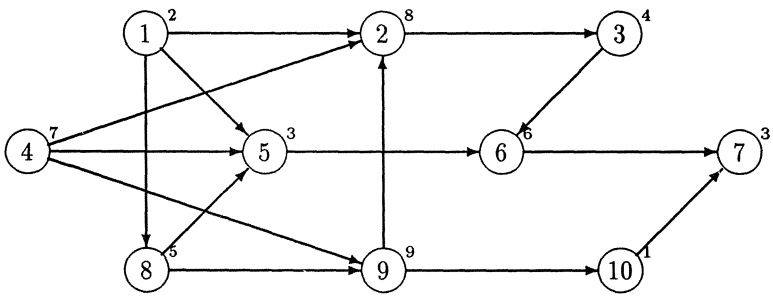
$$\mathcal{M} = \{M_1, M_2, M_3, M_4\}$$
$$\mathcal{J} = \{J_1, J_2, J_3\}$$
$$\mathcal{O} = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$$

u	1	2	3	4	5	6	7	8	9	10
ι_u	J_1	J_1	J_1	J_2	J_2	J_2	J_2	J_3	J_3	J_3
$\rightarrow$	$1 \rightarrow$	$2 \rightarrow$	3	$4 \rightarrow$	$5 \rightarrow$	$6 \rightarrow$	7	$8 \rightarrow$	$9 \rightarrow$	10
μ_u	M_1	M_2	M_3	M_2	M_1	M_3	M_4	M_1	M_2	M_4
p_u	2	8	4	7	3	6	3	5	9	1

(a) Instance.



(b) Instance, represented as a disjunctive graph.



(c) Feasible schedule, represented as an acyclic directed graph.

Figure 1: A job shop scheduling problem.

3. Computational Complexity

Two or three machines. – The two-machine job shop scheduling problem with no more than two operations per job is solvable in $O(n \log n)$ time by a simple extension of Johnson's algorithm for the two-machine flow shop [19]. The extensions to two machines and no more than three operations per job and to three machines and no more than two operations per job are NP-hard [23, 15].

In case of unit processing times, the two-machine job shop scheduling problem is solvable in time that is linear in the total number of operations [18, 5]. Again, the extension to three machines is NP-hard, and so is the case of two machines and processing times 1 or 2 [22].

Two or three jobs. – The job shop scheduling problem with two jobs is solvable in polynomial time by solving a shortest path problem in a network that is based on a geometric interpretation of the problem [31, 17]. The three-job problem has recently been shown to be NP-hard [30].

Deadline two or three. – It is trivial to decide if a schedule of length two exists for a given instance of the job shop scheduling problem. But, how does one decide if a deadline of value three can be met?

4. Optimization algorithms

Optimization algorithms for the job shop scheduling problem proceed by branch and bound. We will describe methods of that type in terms of the disjunctive graph $(\mathcal{O}, A, E)$, where $\mathcal{O}$ is the set of operations, A the arc set, and E the edge set.

A node in the search tree is usually characterized by an orientation of each edge in a certain subset $E' \subset E$. The question then is how to compute a lower bound on the value of all completions of this partial solution. Németi, Charlton & Death and Schrage [26, 8, 29] are among the researchers who obtain a lower bound by simply disregarding $E - E'$ and computing the maximum path weight in the directed graph $(\mathcal{O}, A \cup E')$. A more sophisticated bound, due to Bratley, Florian & Robillard [4], is based on the relaxation of the capacity constraints of all machines except one. They propose to select a machine M' and to solve the job shop scheduling problem on the disjunctive graph $(\mathcal{O}, A \cup E', \{\{i,j\} \mid M_i = M_j = M'\})$. This is a *single-machine problem* where the arcs in $A \cup E'$ define release times and delivery times for the operations that are to be scheduled on machine M' as well as precedence constraints among these operations. This observation has spawned much work on this single-machine problem. Although it is NP-hard, there

exist fairly efficient methods for its solution. Most other lower bounds appear as special cases of the single-machine bound, by relaxing the capacity constraint of M' (which gives Németi's longest path bound), by underestimating the contribution of the release and delivery times, by allowing preemption, or by ignoring the precedence constraints. These relaxations, with the exception of the last one, turn an $\mathcal{NP}$-hard single-machine problem into a problem that is solvable in polynomial time; see [21] for details.

Fisher, Lageweg, Lenstra & Rinnooy Kan [12] investigate surrogate duality relaxations in which either the capacity constraints of the machines or the precedence constraints among the operations of each job are weighted and aggregated into a single constraint. In theory, the resulting bounds dominate the above single-machine bound. Balas [2] describes a first attempt to obtain bounds by polyhedral techniques.

The usual enumeration scheme is due to Giffler & Thompson [14]. It generates all active schedules by constructing them from front to back. At each stage, the subset $\mathcal{O}'$ of operations all of whose predecessors have been scheduled is determined and their earliest possible completion times are calculated. It suffices to consider only a machine on which the minimum value of the earliest completion time is achieved and to branch by successively scheduling next on that machine all operations in $\mathcal{O}'$ for which the release time is strictly smaller than this minimum. In this scheme, several edges are oriented at each stage.

Lageweg, Lenstra & Rinnooy Kan and Carlier & Pinson [21, 7] describe alternative enumeration schemes whereby at each stage, a single edge is selected and oriented in either of two ways. Barker & McMahon [3] branch by rearranging the operations in a critical block that occurs on the maximum weight path.

We briefly outline three of the many implemented branch and bound algorithms for job shop scheduling. McMahon & Florian [25] combine the Giffler-Thompson enumeration scheme with the single-machine bound (disregarding precedence constraints), which is computed for all machines by their own algorithm. Lageweg [20] applies the same branching rule, computes the single-machine bound only for a few promising machines using Carlier's [6] algorithm, and obtains upper bounds with a heuristic due to Lageweg, Lenstra & Rinnooy Kan [21]. Carlier & Pinson [7] implement their novel enumeration schemes, the preemptive version of the single-machine bound (which can be computed in polynomial time), and a collection of powerful elimination rules for which we refer to their paper.

5. Approximation algorithms

Most approximation algorithms for job shop scheduling use a dispatch rule, which schedules the operations according to some priority function. A considerable effort has been invested in the empirical testing of rules of this type [13, 9, 10, 27, 16]

Adams, Balas & Zawack [1] develop a *sliding bottleneck heuristic*, which employs an ingenious combination of schedule construction and iterative improvement, guided by solutions to single-machine problems of the type described above. They also embed this method in a second heuristic that proceeds by partial enumeration of the solution space.

Matsuo, Suh & Sullivan [24] and Van Laarhoven, Aarts & Lenstra [32] apply the principle of *simulated annealing* to the job shop scheduling problem. This is a randomized variant of iterative improvement. It is based on local search, but it accepts deteriorations with a small and decreasing probability in the hope of avoiding bad local optima and getting settled in a global optimum. In the latter paper, the neighborhood of a schedule contains all schedules that can be obtained by interchanging two operations i and j for which $M_i = M_j$ and the arc (i, j) is on a longest path. In the former paper, the neighborhood structure is more complex.

6. $10 * 10 = 930$

The computational merits of all these algorithms are accurately reflected by their performance on the notorious 10-job 10-machine problem instance due to Fisher & Thompson [11].

The single-machine bound, maximized over all machines, has a value of 808. McMahon & Florian citemf-1975 found a schedule of length 972. Fisher, Lageweg, Lenstra & Rinnooy Kan [12] applied surrogate duality relaxation of the capacity constraints and of the precedence constraints to find lower bounds of 813 and 808, respectively; the computational effort involved did not encourage them to carry on the search beyond the root of the tree. Lageweg [20] found a schedule of length 930, without proving optimality; also, he computed a number of multi-machine lower bounds, ranging from a three-machine bound of 874 to a six-machine bound of 907. Carlier & Pinson [7] were the first to prove optimality of the value 930, after generating 22021 nodes and five hours of computing. The main drawback of all these enumerative methods, aside from the limited problem sizes that can be handled, is their sensitivity to particular problem instances and even to the initial value of the upper bound.

The computational experience with polyhedral techniques that has been reported until

now is slightly disappointing in view of what has been achieved for other hard problems. However, the investigations in this direction are still at an initial stage.

Dispatch rules show an erratic behavior. The rule proposed by Lageweg, Lenstra & Rinnooy Kan [21] constructs a schedule of length 1082, and most other priority functions do worse.

Adams, Balas & Zawack [1] report that their sliding bottleneck heuristic obtains a schedule of length 1015 in ten CPU seconds, solving 249 single-machine problems on the way. Their partial enumeration procedure succeeds in finding the optimum, after 851 seconds and 270 runs of the first heuristic.

Five runs of the simulated annealing algorithm of Van Laarhoven, Aarts & Lenstra [32], with a standard setting of the cooling parameters, take 6000 seconds on the average and produce an average schedule length of 942.4, with a minimum of 937. If 6000 seconds are spent on deterministic neighborhood search, which accepts only true improvements, more than 9000 local optima are found, the best one of which has a value of 1006. Five runs with a much slower cooling schedule take about 16 hours each and produce solution values of 930 (twice), 934, 935 and 938. In comparison to other approximative approaches, simulated annealing requires unusual computation times, but it yields consistently good solutions with a modest amount of human implementation effort and relatively little insight into the combinatorial structure of the problem type under consideration.

References

[1] Adams, J., E. Balas, and D. Zawack. The shifting bottleneck procedure for job shop scheduling. *Management Sci.*, 34:391–401, 1988.

[2] Balas, E. On the facial structure of scheduling polyhedra. *Math. Programming Stud.*, 24:179–218, 1985.

[3] Barker, J. R., and G. B. McMahon. Scheduling the general job-shop. *Management Sci.*, 31:594–598, 1985.

[4] Bratley, P., M. Florian, and P. Robillard. On sequencing with earliest starts and due dates with application to computing bounds for the $(n/m/G/F_{\max})$ problem. *Naval Res. Logist. Quart.*, 20:57–67, 1973.

[5] Brucker, P. A linear time algorithm to minimize maximum lateness for the two-machine, unit-time, job-shop, scheduling problem. In *System Modeling and Optimization*, Lecture Notes in Control and Information Sciences, R. F. Drenick and F. Kozin(Eds.), 38:566–571. Springer, Berlin, 1982.

[6] Carlier, J. The one-machine sequencing problem. *European J. Oper. Res.*, 11:42–47, 1982.

[7] Carlier, J., and E. Pinson. An algorithm for solving the job-shop problem. *Management Sci.*, 35:164–176, 1989.

[8] Charlton, J. M., and C. C. Death. A generalized machine scheduling algorithm. *Oper. Res. Quart.*, 21:127–134, 1970.

[9] Conway, R. W., W. L. Maxwell, and L. W. Miller. *Theory of Scheduling*, Addison-Wesley, Reading, MA, 1967.

[10] Day, J., and M. P. Hottenstein. Review of scheduling research. *Naval Res. Logist. Quart.*, 17:11–39, 1970.

[11] Fisher, H., and G. L. Thompson. Probabilistic learning combinations of local job-shop scheduling rules. In *Industrial Scheduling*, Muth and G. L. Thompson(Eds.), 225–251. Prentice Hall, Englewood Cliffs, NJ, 1963.

[12] Fisher, M. L., B. J. Lageweg, J. K. Lenstra, and A. H .G. Rinnooy Kan. Surrogate duality relaxation for job shop scheduling. *Discrete Appl. Math.*, 5:65–75, 1983.

[13] Gere, W. S. Heuristics in job shop scheduling. *Management Sci.*, 13:167–190, 1966.

[14] Giffler, B., and G.L. Thompson. Algorithms for solving production-scheduling problems. *Oper. Res.*, 8:487–503, 1960.

[15] Gonzalez, T., and S. Sahni. Flowshop and jobshop schedules: complexity and approximation. *Oper. Res.*, 26:36–52, 1978.

[16] Haupt, R. A survey of priority rule-based scheduling. *OR Spektrum*, 11:3–16, 1989.

[17] Hardgrave, W. W., and G. L. Nemhauser. A geometric model and a graphical algorithm for a sequencing problem. *Oper. Res.*, 11:898–900, 1963.

[18] Hefetz, N., and I. Adiri. An efficient optimal algorithm for the two-machines unit-time jobshop schedule-length problem. *Math. Oper. Res.*, 7:354–360, 1982.

[19] Jackson, J. R. An extension of Johnson's results on job lot scheduling. *Naval Res. Logist. Quart.*, 3:201–203, 1956.

[20] Lageweg, B. J. Private communication, 1984.

[21] Lageweg, B. J., J. K. Lenstra, and A. H. G. Rinnooy Kan. Job-shop scheduling by implicit enumeration. *Management Sci.*, 24:441–450, 1977.

[22] Lenstra, J. K, and A. H. G. Rinnooy Kan. Computational complexity of discrete optimization problems. *Ann. Discrete Math.*, 4:121–140, 1979.

[23] Lenstra, J. K., A. H. G. Rinnooy Kan, and P. Brucker. Complexity of machine scheduling problems. *Ann. Discrete Math.*, 1:343–362, 1977.

[24] Matsuo, H., C. J. Suh, and R. S. Sullivan. A controlled search simulated annealing method for the general jobshop scheduling problem. Working paper 03-44-88, Graduate School of Business, University of Texas, Austin, 1988.

[25] McMahon, G. B., and M. Florian. On scheduling with ready times and due dates to minimize maximum lateness. *Oper. Res.*, 23:475–482, 1975.

[26] Németi, L. Das Reihenfolgeproblem in der Fertigungsprogrammierung und Linearplanung mit logischen Bedingungen. *Mathematica (Cluj)*, 6:87–99, 1964.

[27] Panwalkar, S. S., and W. Iskander. A survey of scheduling rules. *Oper. Res.*, 25:45–61, 1977.

[28] Roy, B., and B. Sussmann. Les problèmes d'ordonnancement avec contraintes disjonctives. Note DS no. 9 bis, SEMA, Montrouge, 1964.

[29] Schrage, L. Solving resource-constrained network problems by implicit enumeration - nonpreemptive case. *Oper. Res.*, 18:263–278, 1970.

[30] Sotskov, Y. N. The complexity of shop-scheduling problems with two or three jobs. *European Journal Oper. Res.*, 53:326–336, 1991

[31] Szwarc, W. Solution of the Akers-Friedman scheduling problem. *Oper. Res.*, 8:782–788, 1960.

[32] Van Laarhoven, P. J. M., E. H. L. Aarts, and J. K. Lenstra. Job shop scheduling by simulated annealing. *Oper. Res.*, to appear.