11

Sequencing and Scheduling

J.K. Lenstra
Centre for Mathematics and Computer Science, Amsterdam

A.H.G. Rinnooy Kan
Erasmus University, Rotterdam

CONTENTS

The theory of scheduling is concerned with the *optimal allocation* of scarce *resources* to *activities* over time. Of obvious practical importance, it has been

the subject of extensive research over the past decades. In view of the fact that the above description allows for a wide variety of problem types, it should come as no surprise that the development of the theory has gone hand in hand with the refinement of a detailed *problem classification,* for which the ultimate foundation was laid in the classic book *Theory of Scheduling* [Conway, Maxwell & Miller 1967] (see §1.1).

Partly under the influence of their work, the emphasis has been on the investigation of *machine scheduling problems,* in which the activities are represented by *jobs* and the resources by *machines*, each of which can process at most one job at a time. Typically, the number of feasible allocations or *schedules* will be finite, but very large. If all the relevant information on jobs, machines and optimality criterion is known in advance, the scheduling problem becomes an example of a *combinatorial optimization* problem, and indeed most of the techniques developed for such problems have at some point been applied to scheduling problems.

One of the techniques that has been especially successful is the *complexity classification* that results from the theory of $\mathcal{NP}$-completeness (see [Garey & Johnson 1979], §1.1). This theory allows for a formal interpretation of the empirical difference between *easy* and *difficult* combinatorial optimization problems, by equating the former group with the problems that are *well solvable* in the sense that their solution requires only time bounded by a polynomial function of problem size, and the latter group with the *$\mathcal{NP}$-hard* problems for which a polynomial algorithm is very unlikely to exist.

The application of $\mathcal{NP}$-completeness theory in conjunction with various algorithmic techniques has succeeded in settling the complexity status ('well-solvable' or '$\mathcal{NP}$-hard') of the large majority of the scheduling problems that occur in a detailed problem classification first published in [Graham, Lawler, Lenstra & Rinnooy Kan 1979] (see §1.2). We trust that we will be excused for adhering closely to their classification in this bibliography.

Thus, we first classify scheduling problems according to the type of *machine environment* in which they are situated. The simplest such environment is a *single machine,* on which job j has to spend an (integral) *processing time* p_j ($j = 1,\ldots,n$). An obvious generalization is to assume that each job has to be executed on any one of m *parallel machines,* which may be *identical, uniform* (machine i processes its jobs at *speed* s_i) or *unrelated* (machine i is able to process job j at speed s_{ij}). Another generalization is to assume that each job may have to visit more than one machine: if each job requires processing on all m machines in arbitrary order, the system is called an *open shop*; if each job has to visit all machines in a fixed order which is the same for each job, we have a *flow shop*; if the orders are fixed but possibly different for each job, we have a *job shop.* As the bibliography will partly reveal, flow shops and job shops have been the traditional domain of operations researchers and industrial engineers, whereas the study of parallel machine systems has been strongly influenced by their applicability in computer science.

Within each of these subclasses, we may further classify problem types by

specifying certain *job characteristics.* First of all, it is important to distinguish between the case that *preemption* (job splitting) is allowed at zero cost and the case that a job, once started on a machine, must be processed without interruption until its completion on that machine. Secondly, various types of *precedence constraints* may be defined on the job set, that must be respected in each feasible schedule. Another way to generalize the model is to assume that job j becomes available for processing at an (integral) *release date* r_j and has to be completed no later than an (integral) *deadline* d_j; in the basic model, all $r_j = 0$ and all $d_j = \infty$. Further, it is often fruitful to consider the special case of *unit processing times,* in which all $p_j = 1$.

The final component of the classification scheme is the *optimality criterion* adopted. This is usually defined in terms of cost functions f_j of the job *completion times* C_j in a particular schedule; f_j may also depend on a given (integral) *due date* d_j and *weight* $w_j (j = 1,...,n)$. We distinguish between *minmax* criteria, i.e., the minimization of *maximum cost* $\max_j \{f_j(C_j)\}$, and *minsum* criteria, i.e., the minimization of *total cost* $\Sigma_j f_j(C_j)$. Important minmax criteria are *maximum completion time* $\max_j \{C_j\}$ and *maximum lateness* $\max_j \{C_j - d_j\}$; important minsum criteria are *total completion time* $\Sigma_j C_j$, *total tardiness* $\Sigma_j \max\{0, C_j - d_j\}$, the *number of late jobs* Σ_j (*if* $C_j \leqslant d_j$ *then* 0 *else* 1), and the *weighted* versions of these in which the j-th term is multiplied by w_j $(j = 1,...,n)$.

It should be apparent that the number of problems in the above class is huge. Still, as we shall see below, many interesting problem types are not included and require special introduction in the bibliography.

In drawing up this bibliography, we have concentrated on publications that appeared in 1981 or later. For the literature prior to 1981, we refer to the books and surveys listed in §§1.1,2. We have exercised some judgement in determining which publications to include; if any reader feels we have overlooked an important contribution, we would be pleased to hear from him or her. We have not included papers on *parallel* scheduling algorithms, as those are dealt with in one of the other contributions to this volume.

1. Books and Surveys

In this section, we list some books and surveys that will serve as a general introduction to the area and to the less than recent literature, as well as some papers that bear on the problem classification.

1.1. *Books*

R.W. Conway, W.L. Maxwell, L.W. Miller (1967). *Theory of Scheduling,* Addison-Wesley, Reading, MA.

As the first serious book on scheduling theory, this text is now rather outdated but still remarkable for the way it mixes deterministic scheduling with queueing and simulation - a mix that has recently become fashionable again

(see §11).

K.R. Baker (1974). *Introduction to Sequencing and Scheduling,* Wiley, New York.

Although this textbook largely ignores recent issues such as computational complexity and analysis of heuristics, it does provide a very readable introduction to the basic results in the area.

E.G. Coffman, Jr. (ed.) (1976). *Computer & Job/Shop Scheduling Theory,* Wiley, New York.

This edited collection of papers contains some careful reviews of the state of the art around 1975, with particularly nice contributions by R. Sethi on minimizing maximum completion time and by R.L. Graham on the worst case analysis of heuristics.

A.H.G. Rinnooy Kan (1976). *Machine Scheduling Problems: Classification, Complexity and Computations,* Nijhoff, The Hague.

J.K. Lenstra (1977). *Sequencing by Enumerative Methods,* Mathematical Centre Tract 69, Centre for Mathematics and Computer Science, Amsterdam.

These Ph.D. theses contain surveys of optimization algorithms and complexity results. For some single machine, flow shop and job shop problems, branch-and-bound algorithms are developed and evaluated.

M.R. Garey, D.S. Johnson (1979). *Computers and Intractability: a Guide to the Theory of NP-Completeness,* Freeman, San Francisco.

The first textbook on computational complexity offers a well-written introduction to the tools and techniques in this area, with an extremely useful survey of $\mathcal{NP}$-completeness results at the end.

S. French (1982). *Sequencing and Scheduling: an Introduction to the Mathematics of the Job-Shop,* Horwood, Chichester.

Aimed at the same audience as [Baker 1974] (see above), this text covers most of the classical scheduling theory, including computational complexity and analysis of heuristics but with less emphasis on parallel machine models.

M.A.H. Dempster, J.K. Lenstra, A.H.G. Rinnooy Kan (eds.) (1982). *Deterministic and Stochastic Scheduling,* Reidel, Dordrecht.

The proceedings of the NATO Advanced Study and Research Institute on Theoretical Approaches to Scheduling Problems, held in Durham, England in 1981, provide some up-to-date surveys. Particular attention is paid to the interfaces between deterministic and stochastic scheduling. Among the contributors are E.G. Coffman, Jr., M.L. Fisher, J.C. Gittins, E.L. Lawler, M.L. Pinedo, S. Ross, L.E. Schrage, K. Sevcik and G. Weiss.

1.2. *Surveys*

M.J. Gonzalez, Jr. (1977). Deterministic processor scheduling. *Comput. Surveys 9,* 173-204.

A less than complete selection from the scheduling results available in 1977, aimed at a computer science audience.

R.L. Graham, E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan (1979). Optimization and approximation in deterministic sequencing and scheduling: a survey. *Ann. Discrete Math. 5,* 287-326.

Written on the occasion of the DO77 conference in Vancouver in 1977, this survey provides a comprehensive review of optimization and approximation algorithms, including complexity results and worst case performance bounds, based on the problem classification sketched above. More than 150 references to the literature are listed. Need we say more?

E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan (1982). Recent developments in deterministic sequencing and scheduling: a survey. M.A.H. Dempster, J.K. Lenstra, A.H.G. Rinnooy Kan (eds.). *Deterministic and Stochastic Scheduling,* Reidel, Dordrecht, 35-73.

On the occasion of the summer school on deterministic and stochastic scheduling in Durham, England in 1981, the preceding survey was revised to contain information on results up to 1981.

E.L. Lawler, J.K. Lenstra (1982). Machine scheduling with precedence constraints. I. Rival (ed.). *Ordered Sets,* Reidel, Dordrecht, 655-675.

Presented at the Symposium on Ordered Sets in Banff in 1981, this survey provides an exposition of the basic results in precedence constrained scheduling, including a treatment of the influence of so-called series-parallel constraints.

J.K. Lenstra, A.H.G. Rinnooy Kan (1984). Two open problems in precedence constrained scheduling. *Ann. Discrete Math.*

A contribution to the sequel of the Banff meeting, held in Lyon in 1982, this small survey deals with two open questions concerning scheduling unit-time jobs subject to precedence constraints, as well as a few new $\mathcal{NP}$-hardness results.

J. Carlier, P. Chrétienne (1982). Un domaine très ouvert: les problèmes d'ordonnancement. *RAIRO Rech. Opér. 16,* 175-217.

Written in French, this survey is in a sense an update of [Coffman 1976] (see §1.1), with emphasis on contributions by the authors. It is not so much a complete treatment as an attempt to focus on some of the main problem types and solution techniques.

E.L. Lawler (1983). Recent results in the theory of machine scheduling. A. Bachem, M. Grötschel, B. Korte (eds.). *Mathematical Programming: the State of the Art - Bonn 1982,* Springer, Berlin, 202-234.

A tutorial at the 11th International Symposium on Mathematical Programming in Bonn in 1982, this paper emphasizes polynomial algorithms and reviews a number of open problems.

E.G. Coffman, Jr., M.R. Garey, D.S. Johnson (1983). *Approximation Algorithms for Bin-Packing - an Updated Survey,* Bell Laboratories, Murray Hill, NJ.

This survey provides an overview of the analysis of approximation algorithms for the minimization of maximum completion time on identical parallel machines and for the related *bin packing* problem of minimizing the number of machines subject to a given bound on the maximum completion time.

D.S. Johnson (1983). The NP-completeness column: an ongoing guide. *J. Algorithms 4,* 189-203.

The seventh in a series of updates on [Garey & Johnson 1979] (see §1.1), this column surveys two types of complexity issues around parallel machine models: first the parallelization of algorithms and secondly the design and scheduling of multiprocessor systems.

S.C. Graves (1981). A review of production scheduling. *Oper. Res. 29,* 646-675.

This well-written survey deals with a wide range of sequencing and lot-sizing problems. A review of the practice of production scheduling leads to various challenging research questions. More than 100 references are given.

1.3. *Classification*

The problem classification sketched above was introduced in [Conway, Maxwell & Miller 1967] (see §1.1) and refined in [Graham, Lawler, Lenstra & Rinnooy Kan 1979] (see §1.2). In addition, the following papers are relevant.

B.J. Lageweg, J.K. Lenstra, E.L. Lawler, A.H.G. Rinnooy Kan (1982). Computer-aided complexity classification of combinatorial problems. *Comm. ACM 25,* 817-822.

A computer program is described that maintains a record of the known complexity results for a structured class of combinatorial problems. Given listings of well-solvable and $\mathcal{NP}$-hard problems, the program employs a reducibility relation defined on the class to classify each problem as easy, hard or open and to produce listings of the hardest easy problems, the easiest open ones, the hardest open ones and the easiest hard ones. The application of the program to a class of 120 single machine problems is demonstrated.

B.J. Lageweg, E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan (1981). *Computer Aided Complexity Classification of Deterministic Scheduling Problems,* Report BW 138, Centre for Mathematics and Computer Science, Amsterdam.

This documents the results obtained by application of the above-mentioned program to a class of 4536 scheduling problems.

N. Hefetz, I. Adiri (1982). A note on the influence of missing operations on scheduling problems. *Naval Res. Logist. Quart. 29,* 535-539.

If the processing time of an operation is equal to zero, this can be interpreted to mean that the processing time is infinitesimally small, but also that the operation does not exist. These interpretations are by no means equivalent, as is demonstrated by various examples.

2. Single Machine Scheduling: Minmax Criteria

2.1. *Maximum lateness*

J. Carlier (1982). The one-machine sequencing problem. *European J. Oper. Res. 11,* 42-47.

Although the problem of minimizing maximum lateness on a single machine subject to release dates is $\mathcal{NP}$-hard, it possesses sufficient structure to make it reasonably well solvable in practical terms. A very efficient branch-and-bound algorithm was developed by McMahon & Florian (*Management Sci. 17* (1975), 782-792) and refined by Lageweg, Lenstra & Rinnooy Kan (*Statist. Neerlandica 30* (1976), 25-41). The author improves over this method by proposing a different branching rule.

J. Erschler, G. Fontan, C. Merce, F. Roubellat (1982). Applying new dominance concepts to job schedule optimization. *European J. Oper. Res. 11,* 60-66.
J. Erschler, G. Fontan, C. Merce, F. Roubellat (1983). A new dominance concept in scheduling n jobs on a single machine with ready times and due dates. *Oper. Res. 31,* 114-127.

Dominance results among schedules may be used in the obvious way to speed up enumerative procedures. These two papers introduce dominance based on the (r_j,d_j)-intervals, assuming that the objective is simply to meet all due dates. The jobs for which the (r_j,d_j)-interval is minimal under the partial order defined by inclusion, turn out to play an important role: they may be assumed to appear in order of nondecreasing r_j, and the jobs dominated by them in the partial order are, roughly speaking, spread around them.

M.R. Garey, D.S. Johnson, B.B. Simons, R.E. Tarjan (1981). Scheduling unit-time tasks with arbitrary release times and deadlines. *SIAM J. Comput. 10,* 256-269.

The special case in which all processing times are equal (or, equivalently, all $p_j = 1$ and the r_j and d_j need not be integral) has been open for a long

time. In this situation, feasibility of the r_j and d_j can be tested in $O(n \log n)$ time by what amounts to repeated application of a dynamic version of Jackson's rule, which gives priority to the available jobs with the smallest d_j.

G.N. Frederickson (1983). Scheduling unit-time tasks with integer release times and deadlines. *Inform. Process. Lett. 16,* 171-173.

If, in the above problem, all $p_j = 1$ and the r_j and d_j are integral, then Jackson's rule solves the problem in $O(n \log n)$ time. Here, it is shown how an optimal schedule can actually be constructed in $O(n)$ time.

2.2. *Maximum cost*

K.R. Baker, E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan (1983). Preemptive scheduling of a single machine to minimize maximum cost subject to release dates and precedence constraints. *Oper. Res. 31,* 381-386.

The problem described in the title is solved in $O(n^2)$ time by generalizing Lawler's algorithm for the case of equal release dates.

C.L. Monma (1980). Scheduling to minimize the maximum job cost. *Oper. Res. 28,* 942-951.

Let c_j indicate the amount of resource consumed (or, if $c_j < 0$, contributed) by job j. The problem is to find a job permutation π minimizing the maximum cumulative cost $\max_j \{f_{\pi(j)}(\sum_{i=1}^{j-1} c_{\pi(i)})\}$. This problem is shown to generalize various scheduling problems. An $\mathcal{NP}$-hardness proof and polynomial algorithms for special cases are presented.

C.L. Monma (1981). Sequencing with general precedence constraints. *Discrete Appl. Math. 3,* 137-150.

J.B. Sidney (1981). A decomposition algorithm for sequencing with general precedence constraints. *Math. Oper. Res. 6,* 190-204.

These papers study under what conditions certain job interchange techniques can cope with general precedence constraints. This typically results in polynomial solvability for series-parallel constraints and in less complete characterizations of optimality for more complicated structures.

3. Single Machine Scheduling: Minsum Criteria

3.1. *Optimization algorithms: dynamic programming*

In the dynamic programming approach to minimizing total cost on a single machine subject to precedence constraints, the minimum cost of scheduling a set of jobs is related to the minimum cost of all its subsets that are feasible with respect to the precedence constraints. The implementation by Baker & Schrage (*Oper. Res. 26* (1978), 111-120, 444-449), in which each feasible subset receives an integer label within a certain range, produced impressive

computional results. The labeling is, however, not compact in the sense that, conversely, not every integer in the range corresponds to a feasible subset. Thus, there appeared to be room for further improvement.

E.L. Lawler (1979). *Efficient Implementation of Dynamic Programming Algorithms for Sequencing Problems*, Report BW 106, Centre for Mathematics and Computer Science, Amsterdam.

An alternative to the implementation scheme of Baker & Schrage is proposed. Time is proportional to n times the number of feasible sets generated, and space is proportional to n plus the maximum number of feasible sets of given size.

E.P.C. Kao, M. Queyranne (1982). On dynamic programming methods for assembly line balancing. *Oper. Res. 30,* 375-390.

Carefully designed experiments confirm that Lawler's scheme is computationally superior to the Baker-Schrage scheme.

R.N. Burns, G. Steiner (1981). Single machine scheduling with series-parallel precedence constraints. *Oper. Res. 29,* 1195-1207.

A compact labeling scheme is developed for the case that the precedence constraints are series-parallel.

G. Steiner (1984). Single machine scheduling with precedence constraints of dimension 2. *Math. Oper. Res. 9,* 248-259.

The compact labeling scheme from the previous paper is generalized to the case that the precedence constraints have dimension 2.

E.L. Lawler (1982). *Scheduling a Single Machine to Minimize the Number of Late Jobs*, Preprint, Computer Science Division, University of California, Berkeley.

Three results are presented. One is an $O(n \log n)$ algorithm that improves on an $O(n^2)$ method of Kise, Ibaraki & Mine (*Oper. Res. 26* (1978), 121-126). Another is an $O(n^6)$ dynamic programming algorithm for finding an optimal preemptive schedule subject to arbitrary release dates. Finally, the problem (with equal release dates) is shown to be $\mathcal{NP}$-hard when there are deadlines in addition to due dates.

3.2. *Optimization algorithms: branch-and-bound*

C.N. Potts, L.N. Van Wassenhove (1982). A decomposition algorithm for the single machine total tardiness problem. *Oper. Res. Lett. 1,* 177-181.

The problem of minimizing total tardiness on a single machine can be solved in $O(n^4 \Sigma p_j)$ (i.e., pseudopolynomial) time by a dynamic programming algorithm due to Lawler (*Ann. Discrete Math. 1* (1977), 331-342), which decomposes each problem into subproblems. The authors use a similar

decomposition approach, but apply the Baker-Schrage method as soon as the subproblems get sufficiently small. Supported by additional dominance rules, the algorithm solves problems of up to 100 jobs.

C.N. Potts, L.N. Van Wassenhove (1983). An algorithm for single machine sequencing with deadlines to minimize total weighted completion time. *European J. Oper. Res. 12,* 379-387.

Lagrangean relaxation of the constraints $C_j \leqslant d_j$ is applied. The multipliers are constrained so that a simple heuristic for the original problem provides an optimal solution to the relaxed one.

A.M.A. Hariri, C.N. Potts (1983). An algorithm for single machine sequencing with release dates to minimize total weighted completion time. *Discrete Appl. Math. 5,* 99-109.

In the same spirit as the previous paper, the constraints $C_j \geqslant r_j + p_j$ are dualized. A dynamic version of Smith's rule (order the jobs in order of nonincreasing w_j/p_j) solves the relaxed problem.

L. Bianco, S. Ricciardelli (1982). Scheduling of a single machine to minimize total weighted completion time subject to release dates. *Naval Res. Logist. Quart. 29,* 151-167.

The same problem, a simpler lower bound, and more elaborate dominance conditions.

3.3. *Approximation algorithms*

Theoretically, the best possible heuristics are *fully polynomial approximation schemes,* which produce an ϵ-optimal schedule in time polynomial in problem size and $1/\epsilon$.

E.L. Lawler (1982). A fully polynomial approximation scheme for the total tardiness problem. *Oper. Res. Lett. 1,* 207-208.

This scheme applies the author's pseudopolynomial algorithm (*Ann. Discrete Math. 1* (1977), 331-342) to a problem with rescaled processing times. The running time is $O(n^7/\epsilon)$.

G.V. Gens, E.V. Levner (1981). Fast approximation algorithms for job sequencing with deadlines. *Discrete Appl. Math. 3,* 313-318.

This fully polynomial approximation scheme for the problem of minimizing the weighted number of late jobs requires $O(n^2 \log n + n^2/\epsilon)$ time. The emphasis is on the derivation of a tight lower bound so that ideas for the related knapsack problem can be fruitfully employed.

3.4. *Related models: due date selection*

Rather than taking due dates as given, the two papers below treat them as decision variables.

K.R. Baker, J.W.M. Bertrand (1981). A comparison of due-date selection rules. *AIIE Trans. 13,* 123-131.

The problem of minimizing the average due date $\Sigma d_j / n$ is investigated under the assumption that no job may be late and that d_j only depends on job parameters such as r_j and p_j that are known in advance.

S.S. Panwalkar, M.L. Smith, A. Seidmann (1982). Common due date assignment to minimize total penalty for the one machine scheduling problem. *Oper. Res. 30,* 391-399.

A polynomial algorithm is given for the minimization of a weighted sum of a common due date d, total tardiness, and total earliness $\Sigma_j \max\{0, d - C_j\}$.

3.5. *Related models: minimization of variance*

J.J. Kanet (1981). Minimizing variation of flow time in single machine systems. *Management Sci. 27,* 1453-1459.

This paper presents a simple algorithm for minimizing the total absolute difference of job completion times on a single machine, and a heuristic for the more difficult problem of minimizing variance of completion times.

J.J. Kanet (1981). Minimizing the average deviation of job completion times about a common due date. *Naval Res. Logist. Quart. 28,* 643-651.

The total absolute difference between job completion times and a common due date on a single machine can be minimized by a minor modification of the first method from the previous paper.

3.6. *Related models: minimization of the number of setups*

Given a schedule of precedence constrained jobs, a *setup* is said to occur when a job does not directly follow one of its immediate predecessors. To the large literature on this problem, the following papers have been added.

W.R. Pulleyblank (1984). On minimizing setups in precedence constrained scheduling. *Discrete Appl. Math.*

$\mathcal{NP}$-Hardness for the case of a bipartite precedence graph is established, and a polynomial algorithm for (again!) series-parallel constraints is given.

M.M. Syslo (1984). Minimizing the jump number for partially ordered sets: a graph-theoretic approach. *Order 1,* 7-19.

The results for the case of series-parallel constraints are derived in a

different manner.

D. Duffus, I. Rival, P. Winkler (1982). Minimizing setups for cycle-free ordered sets. *Proc. Amer. Math. Soc. 85,* 509-513.

The obvious lower bound on the minimum number of setups, given by the Dilworth width minus 1, is shown to be tight for the case that the precedence graph contains no alternating cycles.

G. Gierz, W. Poguntke (1983). Minimizing setups for ordered sets: a linear algebraic approach. *SIAM J. Algebraic Discrete Meth. 4,* 132-144.

A lower bound that dominates the previous one is presented and shown to be tight for a class slightly more general than series-parallel constraints.

3.7. *Related models: two criteria*

Given two optimality criteria, the following papers deal with the determination of the set of Pareto-optimal points.

L.N. Van Wassenhove, L.F. Gelders (1980). Solving a bicriterion scheduling problem. *European J. Oper. Res. 4,* 42-48.

A pseudopolynomial algorithm is given for the total completion time and maximum lateness criteria.

L.N. Van Wassenhove, K.R. Baker (1982). A bicriterion approach to time/cost trade-offs in sequencing. *European J. Oper. Res. 11,* 48-54.

A procedure is developed for the maximum completion cost and total crashing cost criteria; the crashing cost of job j is given by $c_j(b_j - p_j)$, where p_j $(a_j \leqslant p_j \leqslant b_j)$ is a decision variable and a_j, b_j and c_j are known. The procedure is polynomial under additional assumptions on the completion cost functions.

K.S. Lin (1983). Hybrid algorithm for sequencing with bicriteria. *J. Optimization Theory Appl. 39,* 105-124.

A dynamic programming approach is presented for the total completion time and total tardiness criteria.

4. Nonpreemptive Parallel Machine Scheduling: Independent Jobs

4.1. *Optimization algorithms*

J.Y.-T. Leung (1982). On scheduling independent tasks with restricted execution times. *Oper. Res. 30,* 163-171.

The problem of minimizing maximum completion time on m identical parallel machines can be solved by dynamic programming in $O(\log p_{\max} \cdot \log m \cdot n^{2(k-1)})$ time, if the p_j can take on at most k different values.

B. Simons (1983). Multiprocessor scheduling of unit-time jobs with arbitrary release times and deadlines. *SIAM J. Comput. 12,* 294-299.

The m-machine generalization of the single machine problem solved in [Garey, Johnson, Simons & Tarjan 1981] (see §2.1) is considered. An algorithm with running time $O(n^3 \log \log n)$ is developed.

B. Simons (1982). On scheduling with release times and deadlines. M.A.H. Dempster, J.K. Lenstra, A.H.G. Rinnooy Kan (eds.). *Deterministic and Stochastic Scheduling,* Reidel, Dordrecht, 75-88.

This paper surveys polynomial algorithms and $\mathcal{NP}$-completeness results for scheduling equal-length jobs on one or more identical parallel machines subject to release dates and deadlines.

I. Meilijson, A. Tamir (1984). Minimizing flow time on parallel identical processors with variable unit processing time. *Oper. Res. 32,* 440-446.

A classical result states that the problem of minimizing total completion time on identical parallel machines can be solved by the SPT rule, assigning the jobs to machines in order of nondecreasing p_j. If the machines have a speed that increases over time, the SPT rule remains optimal; if the speed decreases, the problem becomes $\mathcal{NP}$-hard.

4.2. *Approximation algorithms: identical machines*

Unless stated otherwise, the papers in §§4.2−4 consider the minimization of maximum completion time.

In a *list scheduling* heuristic, the jobs are placed in a fixed list and, at each step, the earliest available machine is selected to process the first available job on the list.

J.O. Achugbue, F.Y. Chin (1981). Bounds on schedules for independent tasks with similar execution times. *J. Assoc. Comput. Mach. 28,* 81-99.

For arbitrary list scheduling, tight worst case relative error bounds as a function of $\rho = p_{\max}/p_{\min}$ are obtained. E.g., if $\rho \leqslant 3$, then the bound is equal to $2-1/3\lfloor m/3 \rfloor$ if $m \geqslant 6$, 17/10 if $m = 5$ and 5/3 if $m = 3, 4$.

B.L. Deuermeyer, D.K. Friesen, M.A. Langston (1982). Scheduling to maximize the minimum processor finish time in a multiprocessor system. *SIAM J. Algebraic Discrete Meth. 3,* 190-196.

For the unusual criterion of maximizing the minimum machine completion time, the LPT list scheduling heuristic, in which the jobs are listed in order of nonincreasing p_j, is shown to have a worst case ratio of 4/3. While the result is similar to Graham's result for minimizing maximum completion time (see, e.g., [Coffman 1976] in §1.1), the proof technique is quite different.

4.3. *Approximation algorithms: uniform machines*

Y. Cho, S. Sahni (1980). Bounds for list schedules of uniform processors. *SIAM J. Comput. 9*, 91-103.

It is known that both arbitrary list scheduling on identical machines and LPT list scheduling on uniform machines have a worst case ratio tending to 2 if m goes to infinity. Here, it is shown that for arbitrary list scheduling on uniform machines, the ratio is not bounded by a constant but increases not faster than $O(\sqrt{m})$.

D.K. Friesen, M.A. Langston (1983). Bounds for multifit scheduling on uniform processors. *SIAM J. Comput. 12*, 60-70.

The multifit heuristic, which involves repeated application of the first-fit-decreasing heuristic for bin packing to the packing of jobs in m intervals $[0,C_{\max}]$, is extended to uniform machines and shown to have a worst case ratio between 1.341 and 1.4. This is the best ratio found so far for this model.

4.4. *Approximation algorithms: unrelated machines*

E. Davis, J.M. Jaffe (1981). Algorithms for scheduling tasks on unrelated processors. *J. Assoc. Comput. Mach. 28*, 721-736.

An adaptation of list scheduling is considered that incorporates a search for a relatively fast machine for each job. The worst case ratio is shown to be $O(\sqrt{m})$.

5. Preemptive Parallel Machine Scheduling: Independent Jobs

5.1. *Optimization algorithms*

G. Schmidt (1983). *Preemptive Scheduling on Identical Processors with Time Dependent Availabilities*, Bericht 83-4, Fachbereich 20 Informatik, Technische Universität Berlin.

In case the machines are available only in certain given time intervals, the existence of a feasible preemptive schedule can be tested in polynomial time.

C. Martel (1982). Preemptive scheduling with release times, deadlines and due times. *J. Assoc. Comput. Mach. 29*, 812-829.

Polymatroidal network flow techniques are used to construct a preemptive schedule on uniform machines respecting release dates and meeting deadlines (if it exists) in $O(m^2n^4+n^5)$ time. The algorithm is combined with search techniques to minimize maximum lateness in polynomial time as well.

6. Parallel Machine Scheduling: Precedence Constrained Jobs

6.1. *Optimization algorithms: unit-time jobs*

The fundamental algorithmic results for scheduling precedence constrained unit-time jobs on m identical parallel machines so as to minimize maximum completion time are Hu's algorithm (*Oper. Res. 9* (1961), 841-848) for the case of tree-type constraints and various polynomial algorithms for the case of two machines. The complexity of the problem is open for every fixed number of machines greater than two. There are persistent rumors that these problems have turned out to be well solvable.

O. Marcotte, L.E. Trotter, Jr. (1984). An application of matroid polyhedral theory to unit-execution time, tree-precedence constrained job scheduling. W.R. Pulleyblank (ed.). *Progress in Combinatorial Optimization,* Academic Press, New York, 263-271.

Hu's algorithm is rederived from a minmax result due to Edmonds on covering the elements of a matroid (here, a transversal matroid on the jobs) by its bases (here, so-called feasible machine histories).

C.L. Monma (1982). Linear-time algorithms for scheduling on parallel processors. *Oper. Res. 30,* 116-124.

The generalization of Hu's algorithm to the problem of minimizing maximum lateness subject to intree constraints and some other scheduling problems are implemented to run in linear time by an adapted version of bucket sorting.

H.N. Gabow (1982). An almost-linear algorithm for two-processor scheduling. *J. Assoc. Comput. Mach. 29,* 766-780.

The two-machine problem with arbitrary precedence constraints is solved by an adaptation of Hu's algorithm in almost linear time ...

H.N. Gabow, R.E. Tarjan (1983). A linear-time algorithm for a special case of disjoint set union. *Proc. 15th Annual ACM Symp. Theory of Computing,* 246-251.

... and in strictly linear time.

K. Nakajima, J.Y.-T. Leung, S.L. Hakimi (1981). Optimal two processor scheduling of tree precedence constrained tasks with two execution times. *Performance Evaluation 1,* 320-330.

The two-machine problem with tree-type constraints and processing times equal to 1 or 2 is solved by a complicated $O(n \log n)$ algorithm. (For practical purposes, a heuristic due to Kaufman (*IEEE Trans. Comput. 23* (1974), 1169-1174) which has a worst case absolute error of 1, may be more attractive.)

M.R. Garey, D.S. Johnson, R.E. Tarjan, M. Yannakakis (1983). Scheduling opposing forests. *SIAM J. Algebraic Discrete Meth. 4,* 72-93.

The m-machine problem in which the precedence graph is the disjoint union of an inforest and an outforest is considered. If m is arbitrary, the problem is $\mathcal{NP}$-hard; if m is fixed, it is solvable in polynomial time; if $m = 2$, there is a linear time algorithm.

D. Dolev, M.K. Warmuth (1982). *Profile Scheduling of Opposing Forests and Level Orders*, Research report RJ 3553, IBM, San Jose, CA.

Opposing forests can be scheduled in $O(n^{2m-2}\log n)$ time; this improves over the above algorithm. Level orders, in which any two incomparable jobs with a common predecessor or successor have identical sets of predecessors and successors, can be scheduled in $O(n^{m-1})$ time; the case of arbitrary m is $\mathcal{NP}$-hard.

D. Dolev, M.K. Warmuth (1982). *Scheduling Flat Graphs*, Research report RJ 3398, IBM, San Jose, CA.

The theorems and background of the results in the above paper are presented.

D. Dolev, M.K. Warmuth (1984). Scheduling precedence graphs of bounded height. *J. Algorithms 5,* 48-59.

Precedence graphs in which the longest path has at most h arcs can be scheduled in $O(n^{h(m-1)+1})$ time. The case $h = 2$ is already $\mathcal{NP}$-hard.

E. Mayr (1981). *Well Structured Programs Are Not Easier to Schedule*, Report STAN-CS-81-880, Department of Computer Science, Stanford University.

The m-machine problem remains $\mathcal{NP}$-hard if the graph has a so-called hierarchical parallel structure.

6.2. *Optimization algorithms: preemptive scheduling*

E.L. Lawler (1982). Preemptive scheduling of precedence-constrained jobs on parallel machines. M.A.H. Dempster, J.K. Lenstra, A.H.G. Rinnooy Kan (eds.). *Deterministic and Stochastic Scheduling,* Reidel, Dordrecht, 101-123.

Some well-solvable problems involving the nonpreemptive scheduling of unit-time jobs turn out to have well-solvable counterparts involving the preemptive scheduling of jobs with arbitrary processing times. The latter problems include the minimization of maximum lateness on m identical machines subject to intree constraints and on two uniform machines subject to release dates and arbitrary precedence constraints. These results suggest a strong relationship between the two models.

6.3. *Approximation algorithms*

M. Kunde (1981). Nonpreemptive LP-scheduling on homogeneous multiprocessor systems. *SIAM J. Comput. 10,* 151-173.

In critical path list scheduling, the jobs are listed in order of nonincreasing total processing time of all jobs on the longest path starting at the job in question. This rule is investigated for the case of finding nonpreemptive schedules on identical machines subject to tree-type and chain-type precedence constraints. In the former case, the worst case ratio is $2-2/(m+1)$; in the latter case, the ratio tends to 5/3 as m goes to infinity.

7. Parallel Machine Scheduling: Related Models

7.1. *Additional resource constraints*

The class of scheduling models known as *resource constrained project scheduling,* in which resources are of a more general nature than machines, has generated an impressive literature of its own. Virtually all these problems are $\mathcal{NP}$-hard in a very strong sense. Below, we list a few publications that appear to be on the borderline between the general class and the restricted class considered here.

The first four papers deal with unit-time jobs, arbitrary precedence constraints and the maximum completion time criterion.

E.L. Lloyd (1980). List scheduling bounds for UET systems with resources. *Inform. Process. Lett. 10,* 28-31.

There are m identical machines and l additional resources h of size R_h; job j requires r_{hj} units of resource h during its execution ($j = 1,\ldots,n$; $h = 1,\ldots,l$). Arbitrary list scheduling is shown to have a tight worst case ratio of $\min\{m, 2-1/m+\Sigma R_h(1-1/m)\}$.

E.L. Lloyd (1981). Coffman-Graham scheduling of UET task systems with 0-1 resources. *Inform. Process. Lett. 12,* 40-45.

Here, all $R_h = 1$ and all $r_{hj} \in \{0,1\}$. A generalization of the Coffman-Graham labeling algorithm (*Acta Inform. 1* (1972), 200-213) turns out to have a similar worst case behavior as arbitrary list scheduling.

E.L. Lloyd (1982). Critical path scheduling with resource and processor constraints. *J. Assoc. Comput. Mach. 29,* 781-811.

A complicated analysis shows that, for the model of [Lloyd 1980] (see above), the worst case ratio of a generalization of Hu's algorithm is bounded by a piecewise linear function of l and m.

J. Blazewicz, J.K. Lenstra, A.H.G. Rinnooy Kan (1983). Scheduling subject to resource constraints: classification and complexity. *Discrete Appl. Math. 5,*

11-24.

A detailed complexity classification is given for problems with identical or uniform machines and various types of resource constraints, parametrized according to l, R_h and $\max\{r_{hj}\}$, each of which is taken to be equal to 1 or to an arbitrary integral value.

In another common model, each machine i has its own resource (say, memory) of size R_i and can only process jobs whose resource requirements are no larger than R_i.

T.-H. Lai, S. Sahni (1981). *Preemptive Scheduling of a Multiprocessor System with Memories to Minimize* $L_{\max}$, Technical report 81-20, Computer Science Department, University of Minnesota, Minneapolis.

A network representation yields a preemptive schedule on identical machines minimizing maximum lateness in $O(n^3)$ time.

T.-H. Lai, S. Sahni (1982). *Preemptive Scheduling of Uniform Processors with Memory*, Technical report 82-5, Computer Science Department, University of Minnesota, Minneapolis.

Linear programming formulations are given for finding preemptive schedules on uniform machines minimizing maximum completion time and maximum lateness.

Two yet different models conclude this subsection.

E.L. Lloyd (1981). Concurrent task systems. *Oper. Res. 29,* 189-201.

Again unit-time jobs, arbitrary precedence constraints and the maximum completion time criterion. Job j requires q_j identical machines during its execution. The problem is well solvable for $m = 2$ and $\mathcal{NP}$-hard for $m \geqslant 3$. Arbitrary list scheduling has a worst case ratio $(2m - q_{\max})/(m - q_{\max} + 1)$.

J. Carlier, A.H.G. Rinnooy Kan (1982). Scheduling subject to nonrenewable-resource constraints. *Oper. Res. Lett. 1,* 52-55.

If the resources are actually consumed by the jobs (take, e.g., money) and the machine capacity is not binding ($m \geqslant n$), then minmax problems are well solvable, even if the amount of resource becomes available gradually over time.

7.2. *Periodic scheduling*

In (preemptive) periodic scheduling, each job j has a *period* ρ_j and is to be executed in each interval $(r_j + k\rho_j, d_j + k\rho_j)$ $(k = 0,1,2,...)$. On a single machine, the rule that gives priority to the available job with the closest deadline is known to construct a feasible schedule, if one exists.

J.Y.-T. Leung, M.L. Merrill (1980). A note on preemptive scheduling of

periodic, real-time tasks. *Inform. Process. Lett. 11,* 115-118.

The problem of deciding feasibility is shown to be $\mathcal{NP}$-complete for each $m \geqslant 1$. The above priority rule for $m = 1$ turns out to provide an exponential method, in the sense that it is sufficient to verify whether feasibility has been achieved in a period equal to the least common multiple of the ρ_j; after which the schedule repeats itself.

E.L. Lawler, C.U. Martel (1981). Scheduling periodically occurring tasks on multiple processors. *Inform. Process. Lett. 12,* 9-12.

The last mentioned result is extended to the case of unrelated machines.

A.A. Bertossi, M.A. Bonuccelli (1983). Preemptive scheduling of periodic jobs in uniform multiprocessor systems. *Inform. Process. Lett. 16,* 3-6.

The Lawler-Martel algorithm above allows a more efficient implementation in the case of uniform machines.

J.Y.-T. Leung, J. Whitehead (1982). On the complexity of fixed-priority scheduling of periodic, real-time tasks. *Performance Evaluation 2,* 237-250.

The case of identical machines and equal release dates is solved in pseudo-polynomial time. The complexity of this problem is still open.

7.3. *Restricted starting times*

K. Nakajima, S.L. Hakimi (1982). Complexity results for scheduling tasks with discrete starting times. *J. Algorithms 3,* 344-361.

A detailed complexity analysis is given for the problem of finding a feasible nonpreemptive schedule on m identical machines in which each job j may start at any one of k_j given starting times. Even if the processing times can assume only two different values, the problem turns out to be $\mathcal{NP}$-complete in the case that $m = 1$ and all $k_j \leqslant 3$ and in the case that m is arbitrary and all $k_j = 2$. For some more restricted cases, polynomial algorithms are developed.

K. Nakajima, S.L. Hakimi, J.K. Lenstra (1982). Complexity results for scheduling tasks in fixed intervals on two types of machines. *SIAM J. Comput. 11,* 512-520.

The problem is to find a nonpreemptive schedule on two types of parallel machines: inexpensive slow machines and expensive fast ones. Job j requires a processing time p_j on a slow machine or $q_j < p_j$ on a fast one. Two models are considered: (a) each job j must be processed in an interval $(r_j, r_j + p_j]$; (b) each job j must start at time r_j. The objective is to minimize total machine cost. Both problems turn out to be $\mathcal{NP}$-hard. For some special cases, in which all $q_j = 1$, polynomial algorithms are presented.

8. Open Shop Scheduling

8.1. *Optimization algorithms*

E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan (1981). Minimizing maximum lateness in a two-machine open shop. *Math. Oper. Res. 6,* 153-158.
E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan (1982). Erratum. *Math. Oper. Res. 7,* 635.

The problem of finding a preemptive schedule minimizing maximum lateness in a two-machine open shop is solved by a linear time algorithm. The nonpreemptive case is shown to be $\mathcal{NP}$-hard.

Y. Cho, S. Sahni (1981). Preemptive scheduling of independent jobs with release and due times on open, flow and job shops. *Oper. Res. 29,* 511-522.

The existence of a preemptive schedule respecting release dates and deadlines in an m-machine open shop can be determined by linear programming. The analogous problems for two-machine flow and job shops are $\mathcal{NP}$-hard.

T. Fiala (1983). An algorithm for the open-shop problem. *Math. Oper. Res. 8,* 100-109.

In a very original contribution, results from graph theory are invoked to show that the problem of finding a nonpreemptive schedule minimizing maximum completion time in an m-machine open shop can be solved in $O(m^3n^2)$ time if $\max_i \{\Sigma_j p_{ij}\} \geqslant (16m'\log m' + 5m')p_{\max}$, where m' is the roundup of m to the closest power of 2.

E.L. Lawler, M.G. Luby, V.V. Vazirani (1982). Scheduling open shops with parallel machines. *Oper. Res. Lett. 1,* 161-164.

For a generalization of the preemptive open shop problem, in which there are given speeds s_{ijk} at which machine i can process the kth operation of job j, a linear programming formulation minimizes maximum completion time.

8.2. *$\mathcal{NP}$-hardness results*

J.O. Achugbue, F.Y. Chin (1982). Scheduling the open shop to minimize mean flow time. *SIAM J. Comput. 11,* 709-720.

The problem of finding a nonpreemptive schedule minimizing total completion time in a two-machine open shop, so far a prominent open problem, is shown to be $\mathcal{NP}$-hard through a reduction starting form 3-PARTITION. Further, tight bounds are derived on the quality of arbitrary schedules and shortest-processing-time-first schedules for an m-machine open shop.

T. Gonzalez (1982). Unit execution time shop problems. *Math. Oper. Res. 7,* 57-66.

The problem of finding a nonpreemptive or preemptive schedule

minimizing total completion time in an m-machine open shop is shown to be $\mathcal{NP}$-hard, even if $p_{ij} \in \{0,1\}$ for all (i,j). Similar results hold for the problems of minimizing maximum or total completion time in flow and job shops.

9. Flow Shop Scheduling

9.1. *Optimization algorithms and $\mathcal{NP}$-hardness results*

F.Y. Chin, L.-L. Tsai (1981). On J-maximal and J-minimal flow-shop schedules. *J. Assoc. Comput. Mach. 28,* 462-476.

For special cases of the problem of minimizing maximum completion time in an m-machine flow shop in which, for some machine h, $p_{hj} = \max_i \{p_{ij}\}$ for all j or $p_{hj} = \min_i \{p_{ij}\}$ for all j, $\mathcal{NP}$-hardness results complemented by polynomial algorithms are derived. In addition, bounds on the length of arbitrary permutation schedules are derived.

J.O. Achugbue, F.Y. Chin (1982). Complexity and solution of some three-stage flow shop scheduling problems. *Math. Oper. Res. 7,* 532-544.

A detailed analysis of the three-machine flow shop problem, in which each machine may be maximal or minimal in the above sense, leads to an exhaustive complexity classification.

W. Szwarc (1981). Precedence relations of the flow-shop problem. *Oper. Res. 29,* 400-411.

Conditions are provided under which Johnson's algorithm for the two-machine flow shop can be extended to the m-machine case.

W. Szwarc (1983). Flow shop problems with time lags. *Management Sci. 29,* 477-481.

An extension of the flow shop model is shown to cover many flow shop problems with time lags. Application of Johnson's algorithm yields lower and upper bounds.

J. Grabowski (1982). A new algorithm of solving the flow-shop problem. G. Feichtinger, P. Kall (eds.). *Operations Research in Progress,* Reidel, Dordrecht, 57-75.

A new branching scheme is proposed for the permutation flow shop problem based on an analysis of the transformations required to shorten the critical path corresponding to the feasible schedule at the current node of the search tree, and as such related to earlier work by Balas (*Oper. Res. 17* (1969), 941-957). The algorithm uses the bounding scheme developed by Lageweg, Lenstra & Rinnooy Kan (*Oper. Res. 26,* (1978), 53-67). Grabowski's method requires less time and generates smaller search trees than the method of Lageweg *et al.*

J. Grabowski, E. Skubalska, C. Smutnicki (1983). On flow shop scheduling

with release and due dates to minimize maximum lateness. *J. Oper. Res. Soc. 34,* 615-620.

The above approach is extended to the minimization of maximum lateness subject to release dates.

9.2. *Approximation algorithms*

I. Bárány (1981). A vector-sum theorem and its application to improving flow shop guarantees. *Math. Oper. Res. 6,* 445-452.

A surprising geometrical argument leads to a flow shop heuristic that requires $O(m^3n^2+m^4n)$ time and whose absolute error is bounded by $(m-1)(3m-1)p_{\max}/2$. A remarkable feature of this result is that the error does not depend on n.

H. Röck, G. Schmidt (1982). *Machine Aggregation Heuristics in Shop Scheduling,* Bericht 82-11, Fachbereich 20 Informatik, Technische Universität Berlin.

Aggregation heuristics proceed by replacing m machines by two machines, on which the job processing times are given by the appropriate sums of the original processing times. The worst case ratios of such heuristics are proportional to m.

C.N. Potts (1981). *Analysis of Heuristics for Two-Machine Flow-Shop Sequencing Subject to Release Dates,* Report BW 150, Centre for Mathematics and Computer Science, Amsterdam.

For the problem of minimizing maximum completion time in a two-machine flow shop subject to release dates, three heuristics with worst case ratio 2 are presented. Repeated application of one of them, that is inspired by a dynamic application of Johnson's algorithm to a modified version of the problem, reduces the worst case ratio to 5/3.

9.3. *Related models: no wait in process*

There has always been a special interest in the flow shop model in which all operations of a job must be performed without interruption. The problem of minimizing maximum completion time under this restriction is a special case of the traveling salesman problem. The case of two machines is solvable in $O(n \log n)$ time by the Gilmore-Gomory algorithm for a special TSP; the case of four machines was proved $\mathcal{NP}$-hard by Papadimitriou & Kanellakis (*J. Assoc. Comput. Mach. 27* (1980), 533-549).

H. Röck (1984). The three-machine no-wait flow shop is NP-complete. *J. Assoc. Comput. Mach. 31,* 336-345

This settles the open question implied by the paragraph above.

H. Röck (1984). Some new results in flow shop scheduling. *Z. Oper. Res. 28,*

1-16.

The problems of minimizing maximum lateness and total completion time in a two-machine no wait flow shop are shown to be $\mathcal{NP}$-hard. For the case of unit processing times and a single additional resource of unit size, an $O(n \log n)$ time algorithm is presented.

10. Job Shop Scheduling

10.1. *Optimization algorithms*

The problem of minimizing maximum completion time in a job shop is $\mathcal{NP}$-hard, even in the case of three machines and unit processing times and in the case of two machines and processing times equal to 1 or 2 (Lenstra & Rinnooy Kan, *Ann. Discrete Math. 4* (1979), 121-140). Below, N denotes the total number of operations of all jobs.

N. Hefetz, I. Adiri (1982). An efficient optimal algorithm for the two-machines unit-time jobshop schedule-length problem. *Math. Oper. Res. 7,* 354-360.

The above problem with two machines and unit processing times is shown to be solvable in $O(N)$ time, through a rule that gives priority to the longest remaining job.

P. Brucker (1981). Minimizing maximum lateness in a two-machine unit-time job shop. *Computing 27,* 367-370.

In the same model, maximum lateness can be minimized in $O(N \log N)$ time; the priority of a job now depends on the difference between its due date and its number of operations.

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

The previous algorithm can be implemented to run in $O(N)$ time.

M.L. Fisher, B.J. Lageweg, J.K. Lenstra, A.H.G. Rinnooy Kan (1983). Surrogate duality relaxation for job shop scheduling. *Discrete Appl. Math. 5,* 65-75.

As part of the continuing (and, so far, rather fruitless) attack on the general job shop problem, computational experience is reported with surrogate duality relaxations of capacity and precedence constraints. Although the lower bounds dominate the classical ones and also those obtained by Lagrangean relaxation, a lot of time is required for their computation. The notorious 10-job 10-machine problem remains unsolved.

11. Probabilistic Scheduling Models

Probability theory finds application in scheduling in two ways. The first one is through a *probabilistic analysis* of the performance of scheduling rules: given a probability distribution over the class of problem instances, the behavior of a random variable representing the performance is investigated. The second way arises when certain job data are no longer assumed to be known in advance; for example, the processing time of a job may be a random variable, whose realization becomes known at the job's completion. The term *stochastic scheduling* is usually reserved for the latter interpretation. We list a few typical references in both areas. Ch.6, §7.3, gives more references on the probabilistic analysis of scheduling algorithms.

11.1. *Probabilistic analysis*

P.G. Gazmuri (1981). *Probabilistic Analysis of a Machine Scheduling Problem,* Unpublished manuscript.

The problem of minimizing total completion time on a single machine subject to release dates is studied under the assumption that processing times as well as release dates are independent and identically distributed. For each of two cases characterized by the relation between expected processing time and expected interarrival time, a heuristic is developed whose relative error tends to 0 in probability.

E.G. Coffman, Jr., G.N. Frederickson, G.S. Lueker (1982). Probabilistic analysis of the LPT processor scheduling heuristic. M.A.H. Dempster, J.K. Lenstra, A.H.G. Rinnooy Kan (eds.). *Deterministic and Stochastic Scheduling,* Reidel, Dordrecht, 319-331.

The average performance of the longest-processing-time-first rule, used to minimize maximum completion time on m identical parallel machines, is studied under the assumption that processing times are uniformly distributed on (0,1]. The ratio of expected LPT schedule length to expected optimal length is bounded by $1+O(m^2/n^2)$.

11.2. *Stochastic scheduling*

G. Weiss (1982). Multiserver stochastic scheduling. M.A.H. Dempster, J.K. Lenstra, A.H.G. Rinnooy Kan (eds.). *Deterministic and Stochastic Scheduling,* Reidel, Dordrecht, 157-179.

This is a survey of stochastic scheduling results for parallel machine models. Typical examples are the optimality of the longest(shortest)-expected-processing-time rule for minimizing maximum (total) completion time on uniform machines, under a variety of assumptions on the distribution of processing times.

M. Pinedo, L. Schrage (1982). Stochastic shop scheduling: a survey. M.A.H. Dempster, J.K. Lenstra, A.H.G. Rinnooy Kan (eds.). *Deterministic and Stochastic Scheduling*, Reidel, Dordrecht, 181-196.

This survey deals with stochastic scheduling results for open shop, flow shop (including the no wait case) and job shop models. Most of the stronger results are for two-machine shops. Much work remains to be done.

12. Related Scheduling Models

This final section is devoted to two scheduling models that do not fit into the preceding framework.

12.1. *Cyclic scheduling*

The (k,m)-cyclic staff scheduling problem is to minimize the number of workers in an m-period cyclic schedule such that requirements varying over the periods are met and each person works for k consecutive periods. In the obvious integer programming formulation, the coefficient matrix has a special structure that is capitalized on in the following papers.

J.J. Bartholdi III, H.D. Ratliff (1978). Unnetworks, with applications to idle time scheduling. *Management Sci. 24,* 850-858.

The complement of the matrix has exactly $m-k$ ones in each column. On the basis of this observation, the (5,7)-problem and several related problems are solved in polynomial time by a series of network flow or matching problems.

J.J. Bartholdi III, J.B. Orlin, H.D. Ratliff (1980). Cyclic scheduling via integer programming with circular ones. *Oper. Res. 28,* 1074-1085.

The (k,m)-problem is solved by transforming the integer program to a series of network flow problems. An unusual round-off property allows the problem also to be solved as a linear program. These techniques are extended to more general cyclic scheduling problems.

J.J. Bartholdi III (1981). A guaranteed-accuracy round-off algorithm for cyclic scheduling and set covering. *Oper. Res. 29,* 501-510.

If the workers are only intermittently available, the cyclic staff scheduling problem turns out to be $\mathcal{NP}$-hard, but the linear-programming round-off technique has an acceptable worst case absolute error.

12.2. *Hierarchical scheduling*

Often, scheduling is the last step in a sequence of planning decisions, where each decision affects the form and the constraints of its successors. If resources have to be acquired under uncertainty about what will be required of them,

multistage stochastic integer programming formulations in which the scheduling decision appears at the last stage are a natural class of models. In the following papers, heuristics with strong properties of asymptotic optimality are developed for such models. The probabilistic analyses in question are based on accurate estimates of the value of an optimal schedule.

M.A.H. Dempster, M.L. Fisher, L. Jansen, B.J. Lageweg, J.K. Lenstra, A.H.G. Rinnooy Kan (1981). Analytical evaluation of hierarchical planning systems. *Oper. Res. 29,* 707-716.

This introductory paper provides the motivation for the approach sketched above and gives some preliminary results.

M.A.H. Dempster, M.L. Fisher, L. Jansen, B.J. Lageweg, J.K. Lenstra, A.H.G. Rinnooy Kan (1983). Analysis of heuristics for stochastic programming: results for hierarchical scheduling problems. *Math. Oper. Res. 8,* 525-537.

Results are presented for the case that the scheduling problem involves the minimization of maximum completion time on a set of identical or uniform parallel machines that has to be acquired when only the number of jobs and the probability distribution of their processing times are known.

M.A.H. Dempster (1982). A stochastic approach to hierarchical planning and scheduling. M.A.H. Dempster, J.K. Lenstra, A.H.G. Rinnooy Kan (eds.). *Deterministic and Stochastic Scheduling,* Reidel, Dordrecht, 271-296.

This paper includes a survey of relevant results in stochastic scheduling and discusses some interesting open questions.

J.B.G. Frenk, A.H.G. Rinnooy Kan, L. Stougie (1984). A hierarchical scheduling problem with a well-solvable second stage. *Ann. Oper. Res. 1.*

Here, the scheduling problem involves the minimization of total completion time.

J.K. Lenstra, A.H.G. Rinnooy Kan, L. Stougie (1984). A framework for the probabilistic analysis of hierarchical planning systems. *Ann. Oper. Res. 1.*

Relations between various measures of asymptotic optimality are derived, and general conditions are established under which a two-stage heuristic is asymptotically clairvoyant with probability 1.