Heuristics Supported Local Search for Optimization of Multi Job Shop Scheduling

M. Nandhini
Research and Development Centre
Bharathiar University
Coimbatore-46, Tamil Nadu, India.

S.Kanmani
Department of Information Technology
Pondicherry Engineering College,
Puducherry-14, India.

Rajesh Kumar Sahoo
Department of Computer Science
School of Engineering and Technology
Pondicherry University
Puducherry-14, India.

ABSTRACT
The main objective of the Multi Job Shop Scheduling problem (MJSSP) is to find a schedule of operations that can minimize the final completion time. In this paper, the various approaches with heuristics used to solve MJSSP are studied and its constraints clearly represented in mathematical model. MJSSP has been implemented with Steepest-Ascent Hill Climbing(SAHC) algorithm with constructive heuristics and compared against with the results of depth-first- Dynamic Consistency Enforcement(DCE) . Also SAHC's efficiency is experimentally proved with more optimal and consistent results obtained for various instances.

General Terms
Heuristics, Local Search, Combinatorial Problem.

Keywords
Constraints, heuristics, multi job shop scheduling, mathematical model, steepest ascent hill climbing, depth first.

1. INTRODUCTION
The goal of combinatorial optimization is finding the best possible solution from the set of feasible solutions. The Multi Job-Shop Scheduling Problem (MJSSP) is one of the most difficult problems, as it is classified as NP-Hard problem. This can be solved using either Artificial Intelligence or Operation Research. Scheduling deals with the timing and coordination of activities which are competing for common resources.

MJSSP is one of the most eminent machine scheduling problems in manufacturing systems, operation management, and optimization technology. The goal of MJSSP is to allocate machines to complete jobs over time, subject to the constraint that each machine can handle at most one job at a time. The complexity of MJSSP increases with its number of constraints and size of search space.

The problem formulated is extremely difficult to solve, as it comprises of several concurrent goals and several resources which must be allocated to lead to our goals, which are to maximize the utilization of machines and to minimize the time required to complete the entire process being scheduled (Mesghouni et al., 2004).

Therefore, the exact methods such as the branch and bound method, dynamic programming and constraint logic programming need a lot of time to find an optimal solution. So, it is expected to find an optimal solution using a heuristic search method.

Performance criteria such as machine utilization, each job's execution speed, and total jobs completion time are all dependent on how efficiently the jobs are scheduled in the system. Hence, it becomes increasingly important to develop effective scheduling approaches that help in achieving the desired objectives.

Scheduling is broadly defined as the process of assigning a set of tasks to resources over a period of time (Pinedo , 1995).

The processing complexity increases as moving from single stage shops to job shops. Various methods have been developed to solve the different types of scheduling problems in different shop configurations for the different objectives. These range from conventional methods such as mathematical programming and priority rules to meta-heuristic and artificial intelligence-based methods (Holland, 1992).

Most of the real world manufacturing companies aim at successfully meeting the customer needs while improving the performance efficiency(Tamilarasi et.al, 2010). Huiyuan *et al.* (2009) established a dual resource (machines and moulds) constrained JSSP model, and received outperformed results.

Ping-Teng Chang,Yu-Ting Lo(2001) modeled the multiple objective functions containing both multiple quantitative(time and production)and multiple qualitative objectives in their integrated approach to model the JSSP, along with a genetic algorithm/tabu search mixture solution approach.

Hong Zhou.et.al(2009) proposed a hybrid framework integrating a heuristic and a genetic algorithm (GA) for JSS to minimize weighted tardiness. In which, for each new generation of schedules, the GA determines the first operation of each machine, and the heuristic determines the assignment of the remaining operations.

Li-Ning Xing et al.(2009) proposed a feasible and effective algorithm to move a step closer to the ultimate vision of an automated system for generating optimal or near-optimal production schedules. M.A. Adibi et al.(2010) developed dynamic job shop scheduling that consists of variable neighborhood search (VNS), a trained artificial neural network (ANN). ANN updates parameters of VNS at any rescheduling.

R.A.Mahdavinejad (2007) solved MJSSP by a heuristic algorithm based on the hybrid method of priority dispatching rules according to an ant colony optimization algorithm. By using the suitable hybrid method of priority dispatching rules, the process of finding the best solution would be improved.

Nandhini.M, Kanmani.S(2010),identified the significance of local search with value adding heuristics in improving optimality over timetabling problem.

A. Garrido et al.2000, proposed heuristic techniques for variable and value orderings to be included in two known searching algorithms: *Basic-Depth-First Backtrack and Depth-First-with-DCE(*N.M. Sadeh et al.(1995 & 1996)). The former algorithm is the classical chronological backtracking procedure heuristically improved. The latter uses the additional heuristic *Dynamic Consistency Enforcement* (*DCE*), which dynamically focuses its effort on critical resource sub problems and learns from its previous faults.

In order to show the significance of heuristics in getting feasible solution and the attitude of local search in improving optimal solution of MJSSP, SAHC with constructive heuristic is proposed and implemented on MJSSP. Also, its performance is compared with the results of Depth-First with-*Dynamic Consistency Enforcement(DCE).*

This paper is organized as follows. In Section.2, problem description of *job-shop scheduling* and mathematical representation of its constraints are mentioned. The proposed method and its implementation with heuristics are presented in Section 3. Empirically evaluated experimental results on a set of typical instances are shown with variable sizes of data sets in Section 4. Conclusions and final remarks are discussed in Section 5.

2. MULTI JOB - SHOP SCHEDULING PROBLEM

The JSSP consists of n jobs and m machines. Each job must go through m machines to complete its work. It is considered that one job consists of m operations. Each operation uses one of m machines to complete one job's work for a fixed time interval. Once one operation is processed on a given machine, it cannot be interrupted before it finishes the job's work. In general, one job being processed on one machine is considered as one operation noted as Oji (means j th job being processed on i th machine, $1 \le j \le n, 1 \le i \le m$) (Garey et al. 1976 and Lawler et al. 1993). Each machine can process only one operation during the time interval.

The objective of JSSP is to find an appropriate operation permutation for all jobs that can minimize the makespan Cmax i.e., the maximum completion time of the final operation in the schedule of n×m operations with minimum waiting time of jobs and machines.

The problem can be made to understand with its known constraints (mandatory &optional) / assumptions as listed below. The mathematical modeling of those constraints is also designed.

C1: No machine may process more than one job at a time.
C2: No job may be processed by more than one machine at a time.
C3: The order in which a job visits different machines is predetermined by technological constraints.
C4: Different jobs can run on different machines simultaneously.
C5: At the moment *T*, any two operations of the same job cannot be processed at the same time.
A1: Processing time on each machine is known.
S1: Idle time of machines may be reduced
S2: Waiting time of jobs may be reduced.

Mathematical Modeling of Constraints:

Entities used in mathematical representation of constraints are given below.

J (j1, j2, j3)	:Jobs
M (m1, m2, m3)	:Machines
O (01, 02, 03)	:Sequence of operations for a job
TS	:Processing time of an operation
Cmax	:Makespan
M[j][ts]->m	:Machine name with job j at time ts
J[m][ts]-> j	:Job name on machine m at time ts
Mac[j][o][ts]->m	:Machine name with operation o of job j at time ts

Capacity Constraints

C1: No machine may process more than one job at a time.

$$\exists m1 \in M, \exists j1, j2 \in J, \exists ts \in TS$$
$$if M[j1][ts] = m1, M[j2][ts] \neq m1$$

C2: No job may be processed by more than one machine at a time.

$$\exists m1, m2 \in M, \exists j \in J, \exists ts \in TS$$
$$if J[m1][ts] = j, J[m2][ts] \neq j$$

C3:The order in which a job visits different machines is predetermined by technological constraints.

$$\forall j \in J,$$
$$\exists m,1, m2 \ldots mm \in M, \exists o1, o2, \ldots om \in O,$$
$$j[o1][ts] = m1,\ j[o2][ts] = m2, \ldots.$$
$$j[on][ts] = mm \mid o1 < o2 < \ldots om;$$

C4: Different jobs can run on different machines simultaneously.

$$\exists j1, j2 \in J, \exists m1, m2 \in M, ts \in TS,$$
$$if j1 \neq j2 then M[j1][ts] = m1, M[j2][ts] = m2$$

Precedence Constraints

C5: At the moment *T*, any two operations of the same job cannot be processed at the same time.

$$\exists ts \in TS, \exists j \in J, \exists o1, o2 \in j,$$
$$\exists m1, m2 \in M,$$
$$if \text{Mac[j][o1][ts]} = \text{m1 then Mac[j][o2][ts]} \neq \text{m2}$$

Soft Constraints

S1. Idle time of machines may be reduced.

$$\forall m \in M, idleTime(m) = Min\{t\}$$

S2. Waiting time of jobs may be reduced.

$$\forall j \in J, waitTime(j) = Min\{t\}$$

3. HEURISTICS LOCAL SEARCH

With the elements described in the problem description and by satisfying precedence and capacity constraints , a schedule with minimum fitness value in the states space is to be found.

We propose constructive heuristically improved Steepest-Ascent Hill climbing algorithm to find the optimal solution.

3.1 State Representation

State formation takes number of jobs, number of operations for each job, sequence of operation of each jobs, processing time of each job's operation and allotment of operation over machines as inputs.

In our work, each state is represented as an array of structures. Each structures consists of job name and its operation as members.

Job No.	Op. No.	Job No.	Op. No	… ..	… ..	…	. .	Job No.	Op. No.

Where,
Job No. *i: 1..n*
Operation No. *j : 1..m such that 1<2<3<….<.m*

3.2 Feasible Solution generation

A feasible schedule is generated by the proposed Constructive Heuristics shown in figure 1. First operation in a schedule can be scheduled *if it is been first operation of any of the jobs. This is done by Verifying Initial Allotment* module. Following this , unscheduled operation of all the jobs are scheduled by verifying operation consistencies and capacity constraints.

3.3 Objective function

The main objective of the MJSSP is to find a schedule of operations that can minimize the final completion time (called makespan), that is the completed time of carrying total operations out in the schedule for n jobs and m machines.

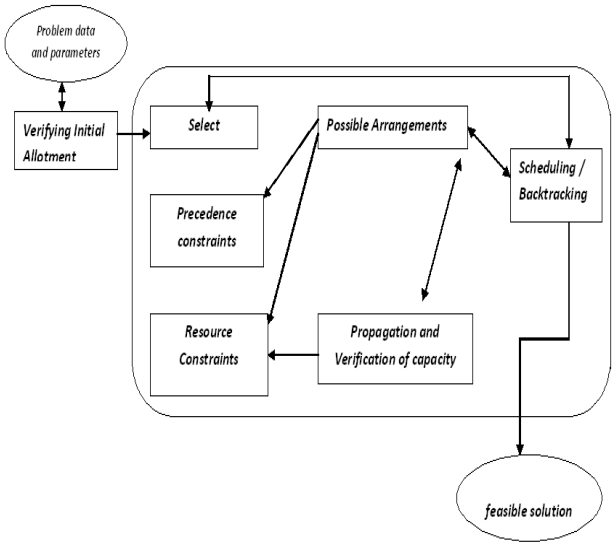

Fig.1: Feasible Solution with Constructive Heuristic

The objective or fitness function takes input as the number of jobs, number of operations/machines, operation time sequence and machine sequence of the corresponding operation. Also, a multi-objective performance measure is included in the objective function that consists of makespan and tardiness of job and machine.

Objective function Min Z = Min (Makespan+ Jobs Waiting Time+---------- eq.(1) Machines Waiting Time)

Where,
Job No. *I : 1..n*
Operation No. *j : 1..m; 1<2<3<….<.m*

3.4 Methods for Controlling the Execution

Searching for optimal combination could be controlled by the following strategies.

- *Reaches the optimum***:** This option is enabled if a combination satisfying all hard and at the most soft constraints have been obtained and saves the result of combination and quit from the execution.
- *Backtracking:* This option is enabled when a combination having the best evaluation value is not reached in the successive levels. By enabling this, either another initial combination or next best adjacent combination in the current level will be taken up for further iterations.
- *Breadth Wise Search :* When two successive combinations of same fitness value are found in two successive levels of a current state, to continue the neighborhood search, the adjacent combination (breadth wise) of current state is taken up for further processing.

3.5 Steepest Ascent – Hill Climbing

This process starts with initial schedule attained with heuristics. Its neighborhood feasible schedules establish state space which are formed by altering the allotment of jobs and its operations in a schedule, called as generation. Each schedule fitness is evaluated as *eq.(1)*. Searching for optimal solution is done by the following algorithm and is shown in figure 2.

Algorithm : Steepest Ascent Hill Climbing

1.Evaluate the initial state.
2.Loop until a solution is found or a complete iteration produces no change to current state:
- SUCC = a state such that any possible successor of the current state will be better than SUCC (the worst state).
- For each operator that applies to the current state, evaluate the new state:
if goal → quit
if better than SUCC → set SUCC to this state
if SUCC is better than the current state → set the current state to SUCC.

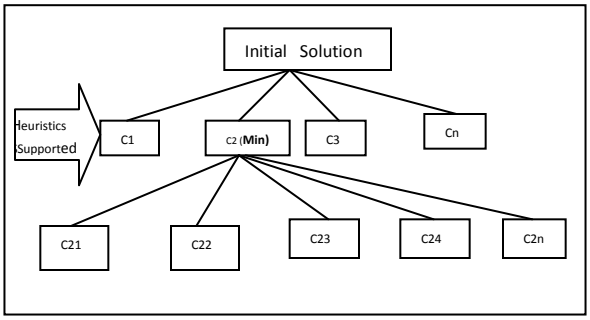

Fig.2: Steepest Ascent Hill Climbing Procedure

3.6 Depth First with DCE

A. Garrido.et.al(2000) used the additional heuristic *Dynamic Consistency Enforcement* (DCE), which dynamically focuses its effort on critical resource sub problems and learns from its previous faults along with chronological backtracking. In this method, *Consistency according to the precedence constraints and Consistency according to constraints about resource capacities* are considered in the scheduling process.

4. EMPIRICAL EVALUATION

In this section, we study the empirical evaluation of the developed heuristic method. The empirical method performance is compared with the algorithms D*epth- First-with-DCE and SAHC*. Both algorithms use the same variable/value ordering heuristics and the same techniques of consistency enforcing.

- *Depth-First-with-DCE*. This algorithm uses chronological backtracking and D*CE* heuristic.
- *Steepest Ascent Hill Climbing.* This algorithm uses the efficiency of a chronological backtracking and taking the best while moving along the search space.

4.1 Design of data set

Six instances of *job-shop scheduling* problems have been designed with small, medium and large sizes of number of jobs, operations and resources given in Table 1.

Table.1: Data set design

	Type	Jobs	Operations	Machines
SD1	Type 1	5	5	5
SD2	Type 2	5	6	6
MD1	Type 3	7	6	6
MD2	Type 4	7	7	7
LD1	Type 5	10	6	6
LD2	Type 6	10	10	10

4.2 Algorithm Comparison

The algorithms have been implemented using Java (Jdk 1.6). These algorithms are tested with a standard MJSSP requirements specified in Section 2. The parameters used in implementation are generation from 0 to 50, various types of problems instances set of machines , jobs and its operations. The value of makespan, waiting time of job, waiting time of machine ,fitness in different generations from 0 to 50 for various types of problems instances are found and tabulated in Table 3&4.

In depth first with DCE, since a depth-bound was set in the solution search, when it reaches 50th level, the search process stops. As the chronological backtracking method is not enough to solve complex *job-shop* problems, depth first with DCE gives better results because of its consistency checking of precedence and resource constraints. In spite of the low rate of reduction in finding the solutions, the efficiency of SAHC is appropriate enough and exposed through difference of fitness given in Table 2. This difference is due to the fact that taking the best in each iteration along with consistent checking.

Table .2: Fitness Difference

	5th generation	10th generation	50th generation
	Fitness	Fitness	Fitness
Type1	48	23	10
Type 2	29	28	28
Type 3	185	99	3
Type 4	52	47	20
Type 5	452	126	150
Type 6	374	251	230

4.3 Results and Discussions

From the fitness difference comparison chart in figure 3 , it is understood that improvement in fitness exists on generation grows in both the algorithms. But the difference(ie. Fitness of depth first with DCE - Fitness of SAHC) in each generation reveals that in majority of data sets, SAHC with heuristics is outperforming the other one.

4.3.1 Sample Schedule

Operation Sequence

	M 1	M 2	M 3	M 4
J1	1	2	3	4
J2	4	1	2	3
J3	3	4	1	2
J4	2	3	4	1

Processing Time for each operation

	O1	O2	O3	O4
J1	1	2	3	4
J2	5	6	7	8
J3	7	6	5	1
J4	8	2	4	3

Optimal Sequence of Scheduling:

10	00	30	01	20	11	31	21	12	32	22	23	33	02	13	03

Fitness Value of Optimal Solution:

Fitness value	=70
Makespan	=29
Jobs Waiting Time	=19
Machines Waiting Time	=22

4.4 Gantt Chart Inference

Maximum time taken to complete all jobs operations is 29 units. In order to reduce time, parallel accessing of operations of different jobs on different machines are made possible. In Gantt chart, figure 4, each job's execution is shown with one color and processing time is given within it. Each machine waiting time is the sum of time gap between any two operations allocated on it. Each job's waiting time is the sum of time gap between consecutive operations of that job.

Table :3: Fitness : Steepest Ascent Hill Climbing

	STEEPEST ASCENT HILL CLIMBING															
	Initial Fitness				5th Generation				10th Generation				50th Generation			
	MS	WJ	WM	Fitness	MS	WJ	WM	Fitness	MS	WJ	WM	Fitness	MS	WJ	WM	Fitness
Type1	213	194	305	**712**	208	145	290	**643**	208	132	299	**639**	205	125	280	**610**
Type2	273	239	445	**957**	260	230	428	**918**	255	222	434	**911**	250	218	434	**902**
Type3	443	765	671	**1879**	436	609	599	**1644**	436	609	599	**1644**	430	605	595	**1630**
Type4	537	987	1223	**2747**	537	811	1096	**2444**	537	800	1083	**2420**	529	790	1080	**2399**
Type5	1028	2234	642	**3904**	956	1817	547	**3320**	956	1817	547	**3320**	910	1700	490	**3100**
Type6	921	3789	4128	**8838**	851	3812	3297	**7960**	850	3820	3290	**7960**	860	3810	3228	**7898**

Table .4 : Fitness : Depth First with DCE

	Depth First with DCE															
	Initial Fitness				5th Generation				10th Generation				50th Generation			
	MS	WJ	WM	Fitness	MS	WJ	WM	Fitness	MS	WJ	WM	Fitness	MS	WJ	WM	Fitness
Type1	213	194	305	**712**	211	167	313	**691**	206	153	303	**662**	200	130	290	**620**
Type2	273	239	445	**957**	235	271	441	**947**	235	267	437	**939**	233	263	434	**930**
Type3	443	765	671	**1879**	440	732	657	**1829**	432	677	634	**1743**	420	650	563	**1633**
Type4	537	987	1223	**2747**	512	893	1145	**2496**	486	888	1093	**2467**	480	870	1069	**2419**
Type5	1028	2234	642	**3904**	1028	2141	603	**3772**	941	1926	579	**3446**	910	1820	520	**3250**
Type6	921	3789	4128	**8838**	917	3716	3701	**8334**	883	3716	3612	**8211**	881	3700	3547	**8128**

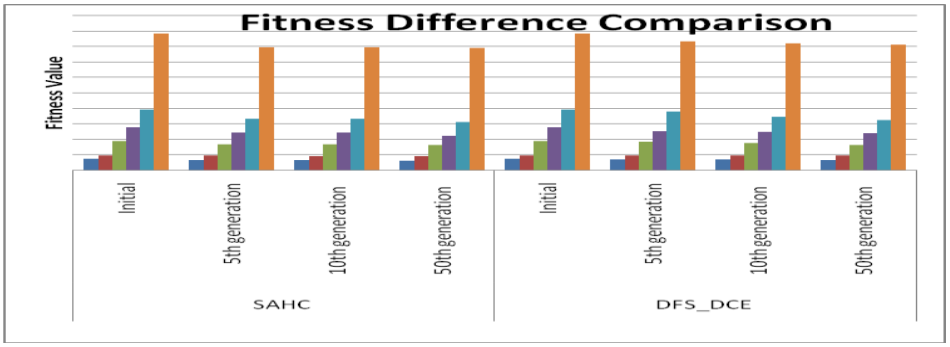

Fig. 3: SAHC & DFS-DCE Fitness Difference Comparison

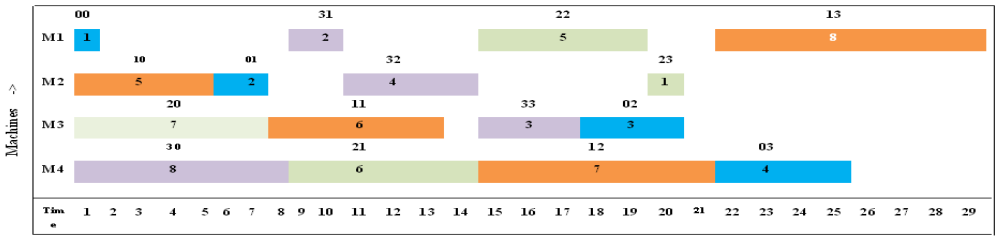

Fig.4: Gantt Chart

5. CONCLUSION

A static multi constrained combinatorial problem of multi Job shop scheduling is implemented using local search algorithms; steepest ascent hill climbing and depth first search-DCE with heuristics. Significance of heuristics in finding feasible could be identified. SAHC with heuristics proved to be the one to give near optimal solutions by comparing with depth first - DCE. In future, this local search could be combined with evolutionary algorithms

6. REFERENCES

[1] Mesghouni, K., Hammadi, S., and Borne, P. 2004. Evolutionary algorithms for Job-Shop Scheduling. Int. J. Applied Mathematics and Computer Science,14(10),(2004), 91-103.

[2] Pinedo,, M. 1995. Scheduling: theory, algorithms and systems . Englewood Cliffs, NJ; Prentice Hall.

[3] Holland, J.H 1992. Adaptation in Natural and Artificial Systems . MIT Press, Cambridge.

[4] Tamilarasi,A., Anantha Kumar, T. 2010. "An enhanced genetic algorithm with simulated annealing for job-shop Scheduling. Int. J. of Engineering, Science and Technology, 2 (1), 144-151.

[5] Huiyuan, R., Lili, J., Xiaoying, X., and Muzhi, L. 2009. Heuristic optimization for dual-resource constrained job shop scheduling. In Proceedings of Int. Asia Conference on Informatics in Control, Automation and Robotics.

[6] Ping-Teng Chang, Yu-Ting Lo. 2001. Modeling of Job-shop Scheduling with Multiple Quantitative and Qualitative Objectives and a GA/TS Mixture approach. Int. J. Computer Integrated Manufacturing, 14(4),367-384..

[7] Hong Zhou, Waiman Cheung, Lawrence C. Leung. 2009. Minimizing Weighted Tardiness of Job-shop Scheduling Using a Hybrid Genetic Algorithm. European Journal of Operational Research, 194(3), 637-649.

[8] Li-Ning Xing, Ying-Wu Chen, Ke-Wei Yang. 2009. An Efficient Search Method for multi-objective Flexible Job Shop Scheduling Problems. Journal of Intelligent Manufacturing, 20(3),283-293.

[9] Adibi, M.A., Zandieh,M., and Amiri,M. 2010 . Multi-objective Scheduling of Dynamic Job Shop Using Variable Neighborhood Search. Expert Systems with Applications, 37(1), 282-287.

[10] Mahdavinejad, R.A .(2007). Multiple Job Shop-Scheduling using Hybrid Heuristic Algorithm. Int. J. Engineering and Natural Sciences,1 (1), 53-58.

[11] Nandhini,M., and Kanmani,S.2010. Design of Mathematical representation and optimization of course timetabling using local search. Int. J. of Combinatorial Optimization and Informatics,1(2), 20-30.

[12] Garrido, A., Salido, M.A., Barber, López, F.A. 2000. Heuristic Methods for Solving Job-Shop Scheduling Problems. ECAI-2000-Workshop on New Results in Planning, Scheduling and Design, 44-49.

[13] Sadeh, N.M., Sycara, K., and Xiong, Y. 1995. Backtracking techniques for the job-shop scheduling constraint satisfaction problem. *Artificial Intelligence*, 76,.455-480.

[14] Sadeh, N.M., and Fox, M.S. 1996. Variable and value ordering heuristics for the job shop scheduling constraint satisfaction problem. *Artificial Intelligence*, 86, 1-4.

[15] Garey, J.E., Johnson, D.S., and Sethi, R. 1976. The complexity of flowshop and jobshop scheduling. Mathematics of Operations Research, 1(2), 117–129.

Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.H.G., and Shmoys, D.B. 1993. Sequencing and scheduling: Algorithms and complexity. Handbooks in Operations Research and Management Science, 4, 445-552.