MULTI-OBJECTIVE OPTIMIZATION AND HEURISTIC APPROACHES FOR SOLVING SCHEDULING PROBLEMS

Gyula Kulcsár, Ferenc Erdélyi and Olivér Hormyák

University of Miskolc, Department of Information Engineering
Egyetem Road, Miskolc-Egyetemváros 3515, Hungary

Abstract: In this paper a mixed approach is presented to solve multi-objective production scheduling and rescheduling problems. This paper describes a new method which can be used in meta-heuristics for comparing schedules in accordance with multiple objectives. The scheduling task consists of batching, assigning, sequencing and timing parts. A new integrated approach has been developed by the authors to solve all the sub-problems as a whole without decompositions. An application of the approach is presented to the multi-objective extended flow shop scheduling and rescheduling problems. The applied method is based on the well-known tabu search meta-heuristic. Moreover the advanced structure of the tabu list is presented besides new relational and neighbourhood operators for multi-objective aim.

Keywords: shop floor control, scheduling, rescheduling, multi-objective optimization, tabu search, meta-heuristic.

1. INTRODUCTION

In a multi-objective scheduling problem class, we wish to find such a feasible schedule which optimizes a set of objective functions and subjects to a set of well-defined special constraints. The task is NP-hard therefore the "optimal" schedule is defined as a result of evolving process in which an engineer and/or a computer program may reach the desired (and compromised) values of the scheduling variables.

Meta-heuristics (e.g. genetic algorithms, simulated annealing and tabu search) have become successful methods for optimization problems wich are too complex to be solved by deterministic techniques (see i.e. Baykasoğlu, *et al*., 2002; Sbalzarini, *et al*., 2000; Smith, *et al*., 2004; Yamada, 2003). To solve a multi-objective scheduling problem, it is necessary to answer an additional important question: What does "good" schedule mean? It is not easy to specify the answer in mathematical form because in real-life situations there are many key performance indices used as objectives (e.g. one based on cost, delivery capability, machine utilization rate, stock level, etc.) and they are usually conflicting. The actual importance of objectives may change frequently in time. A typical appearance of this problem in customized mass production is the decision emerged at shop floor scheduling and rescheduling tasks.

Optimization problems often involve more than one aspect thus it is required to use multiple criterions simultaneously. An objective can be regarded as a measure to evaluate the quality of the solution from a given point of view.

In the literature, different approaches can be found considering multi-objective scheduling problems, as they are surveyed i.e. by Loukil, *et al*. (2005) and Smith, *et al*., (2004). Four main approaches are as follows:

- Simultaneous optimization method.
- Weighting objectives method.
- Hierarchical optimization method.
- Goal programming method.

The simultaneous method (also known as Pareto approach) aims to generate the complete Pareto set or to approximate a set of efficient solutions. A set of solution is said to be Pareto set if passing from solution s_A to another solution s_B in the set, any improvement in one of the objective functions from its current value would cause at least one of the other objective functions to deteriorate from its current value.

The weighting objectives method creates a weighted linear combination of the objective functions to obtain a single function, which can be solved using any single optimization method.

The hierarchical optimization method allows the decision maker to rank the objectives in a descending order of importance. Each objective function is then minimized individually subject to a constraint that does not allow the minimum for the new function to exceed a prescribed fraction of a minimum of the previous function.

Goal programming method takes the objectives into constraints which express satisfying goals. The aim is to find a solution which provides good values of pre-defined goals for each objective.

In multi-objective optimization approaches, comparison of the solutions is an important issue.

2. MULTI-OBJECTIVE SEARCHING APPROACH BASED ON RELATIVE QUALITY

According to our new approach the relative goodness of a solution is more important than the absolute goodness of one. The base of the approach is the following: the relative goodness of the selected solution is measured by comparing it with another solution in the feasible space.

Let S be the search space under consideration. It is the set of all possible solutions to our problem. Suppose that we have a number of objective functions $f_1, \ldots, f_K$ such that:

$$f_k : S \to \Re^+ \cup \{0\}, \ \forall k \in \{1,2,\ldots,K\}. \quad (1)$$

The problem is to find an $s \in S$ that minimizes every $f_k(s)$. This is known as a multi-objective optimization problem. In many cases, it is not possible to find a solution to a multi-objective optimization problem. Successfully minimizing one of the component objective functions will typically increase the value of another one. So we must find solutions that represent a compromise among the various criteria used to evaluate the quality of solutions.

More formally, let $s_x, s_y \in S$ be two solutions. We define the function F to express the quality of s_y compared to s_x as a real number:

$$F : S^2 \to \Re,$$

$$F(s_x, s_y) = \sum_{k=1}^{K} (w_k \cdot D(f_k(s_x), f_k(s_y))). \quad (2)$$

The definition of function D is the following:

$$D : \Re^2 \to \Re, \quad a, b \in \Re,$$

$$D(a,b) = \begin{cases} 0, if \ \max(a,b) = 0 \\ \dfrac{b-a}{\max(a,b)} \cdot 100, \ otherwise \end{cases}. \quad (3)$$

The *max(a,b)* used in (3) denotes an operator:

$$\max : \Re^2 \to \Re,$$

$$\max(a,b) = \begin{cases} a, \ if \ a > b \\ b, \ otherwise \end{cases}. \quad (4)$$

Moreover, to express the importance of any component objective function f_k, we use the coefficient w_k which is an integer value within range *[0, 1, ..., W]*. It is allowed that the decision maker sets the actual priority of each objective function independently.

Using the function F, two solutions $s_x, s_y \in S$ are compared as follows:

s_x is better solution than s_y ($s_x < s_y$ is true) if

$$F(s_x, s_y) > 0. \quad (5)$$

s_x and s_y are equal ($s_x = s_y$ is true) if

$$F(s_x, s_y) = 0. \quad (6)$$

s_x is worse solution than s_y ($s > s_y$ is true) if

$$F(s_x, s_y) < 0. \quad (7)$$

These definitions of the relational operators are suitable for applying in meta-heuristics like tabu search, simulated annealing and genetic algorithms to solve multi-objective combinatorial optimization problem.

In the following, the application of the proposed approach in production scheduling tasks is presented.

3. PREDICTIVE SHOP FLOOR SCHEDULING

3.1 Problem description.

In this section, we are focusing on an extended flexible flow shop model with alternative routes, unrelated parallel machines and limited resource availability time intervals. The problem is inspired by a real case study concerning a multinational firm specialized in lighting products in Hungary. It is a customized mass production approach so that an order-book for a given time period corresponds to different products to be produced in required quantity. We concentrate on generating short-term, detailed production schedule of the manufacturing and assembly processes.

In the manufacturing-assembly system different products may be produced. Each production order includes the identifier of the product, the required quantity and the demanded due date. At the shop floor level, product-pallets can be moved between automated machines (production-lines). Each pallet consists of a pre-decided number of products. Each production order is identified to be consisting of a particular number of pallets. In the system, the basic handling unit is the product-pallet. It is worth to schedule pallets, so one pallet means one job. Each job has four main attributes: 1.) the type of the final product, 2.) the quantity of the products to be processed, 3.) the start time (the earliest time when all of the required material available in the needed quantity) and 4.) the due date demanded.

Each job has to visit a given number of technology steps in a specified sequence. A technology step may include more sub-operations, but no pre-emption is allowed at the level of the technology steps. Typical technology steps may be preparing, manufacturing, quality checking and packaging.

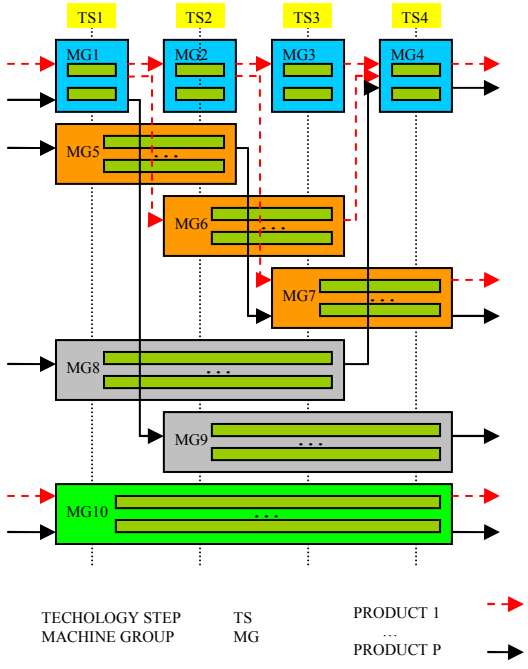

Fig. 1. Extended Flexible Flow Shop.

The workshop contains different machine groups (MG) connected to each other in a given configuration (Figure 1.). Each machine group contains a pre-defined number of machines. In a given machine group, each machine can process the same execution step which is a well-defined set and sequence of technology steps. The machines are not consecutively available for processing the pallets, therefore they may have one or more non-availability time-intervals. In addition, each machine may have different production rates (quantities producible per time unit) for different products. Similarly, each machine may be characterized by product sequence dependent setup times (time delay to changeover from one product type to another product type).

A given final product or a semi-finished product can be produced differently, because there are different execution routes on which the required components are taken through becoming the specified product. In detail, the sequence of execution steps is called execution route. So each execution route includes all the technology steps required in order that the product can be produced. An execution route can include one or more execution steps, but the common part of included execution steps, which is a set of technology steps, has to be an empty set.

At the start time of the scheduling process the machine lines has already been loaded but the actual state of the system is known. It means that the effect of the last confirmed schedule must be considered in the new schedule.

3.2 EFFS Model Specification.

Concentrating on the shop floor scheduling models and solving methods it comes into view that lots of flexible flow shop model exist in the literature (i.e.: Linn, and Zhang, 1999; Wang, 2005; Kis, and Pesch, 2005; Quadt, and Kuhn, 2007; Zhu, and Wilhelm, 2007) but they do not consider all the requirements of the scheduling problem summarized above. The flexible flow shop (FFS) model has to be extended to a new model that supports execution steps, alternative technological routes, and unrelated parallel machines with different capabilities. Sequence dependent setup times, special job characteristics and different objective functions are also considered at the same time. In order to formulate a new class of scheduling problems, the well-known formal specification $\alpha|\beta|\gamma$ is used, where:

- α denotes machine environment descriptors,
- β denotes processing characteristics and constraints,
- γ denotes the list of objective functions.

The Extended Flexible Flow Shop (EFFS) scheduling model can be described as follows:

$$Fx, M_g, Q_{i,m}, Set_{i,j,m}, Cal_m \left| R_i, D_i, Exe_i, A_i \right| [f_1, f_2, ..., f_K]. \quad (8)$$

$\alpha = [\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5]$

$\alpha 1$: The type of operation (technology step) sequence at shop floor level. *F*: Flow Shop, *x*: the maximum number of operations.

$\alpha 2$: The type of machine environment. M_g: group of multi-purpose machines which can execute one or more operations in a given sequence. Each machine group can include distinct parallel machines.

$\alpha 3$: The type of alternative machines. $Q_{i,m}$: unrelated parallel machines with job dependent production rates.

$\alpha 4$: The type of machine setups. $Set_{i,j,m}$: job sequence and machine dependent setup times.

$\alpha 5$: Special resource constraint. Cal_m: machine availability time intervals.

$\beta = [\beta_1, \beta_2, \beta_3, \beta_4]$

$\beta 1$: Constrained released time of the jobs. R_i: the earliest start time of the jobs.

$\beta 2$: Constrained due date of the jobs. D_i: the latest completion time of the jobs.

$\beta 3$: The type of manufacturing processes. Exe_i: required type and sequence of technology steps for jobs.

$\beta 4$: Constrained resource assignments. A_i: set of suitable machines for the jobs.

$\gamma = [\gamma_1, \gamma_2, ..., \gamma_K]$

In real-life situations, there are many different objectives. EFFS model is able to consider multiple objectives. For due date related objectives, we assume that there are production orders which consist of jobs J_i *(i = 1, 2,..., N)* and manufacturing tasks Q_t *(t = 1, 2,..., Z)*. Each job *i* has due date D_i. The completion time of job *i* is denoted by C_i. The lateness of the job *i* is:

$$L_i = C_i - D_i, \quad (9)$$

and the tardiness of the job *i* is:

$$T_i = \max(0, L_i), \quad (10)$$

In current study, five objective functions are considered for demonstrating multi-objective predictive scheduling. These are as follows: the number of tardy jobs (11), the sum of tardiness (12), the maximum tardiness (13), the number of setups (14), the maximum completion time (15).

$$f_1 = |\{J_i \mid T_i > 0\}|, \quad (11)$$

$$f_2 = \sum_i T_i, \quad (12)$$

$$f_3 = \max_i (T_i), \quad (13)$$

$$f_4 = |\{O_t \mid setuptime_t > 0\}|, \quad (14)$$

$$f_5 = \max_i (C_i). \quad (15)$$

3.3 An Integrated Heuristic Solving Method.

The extended flexible flow shop scheduling problem (EFFS) is difficult to solve because of its combinatorial nature. The model inherits the

difficulties of the classical flow shop (FS) and the flexible flow shop (FFS) models. Additionally, numerous strange features appear because of special extensions. The scheduling task consists of batching, assigning, sequencing and timing parts. We developed a new integrated approach to solve all the sub-problems as a whole without decompositions. In our approach all the issues are answered simultaneously (Figure 2.).

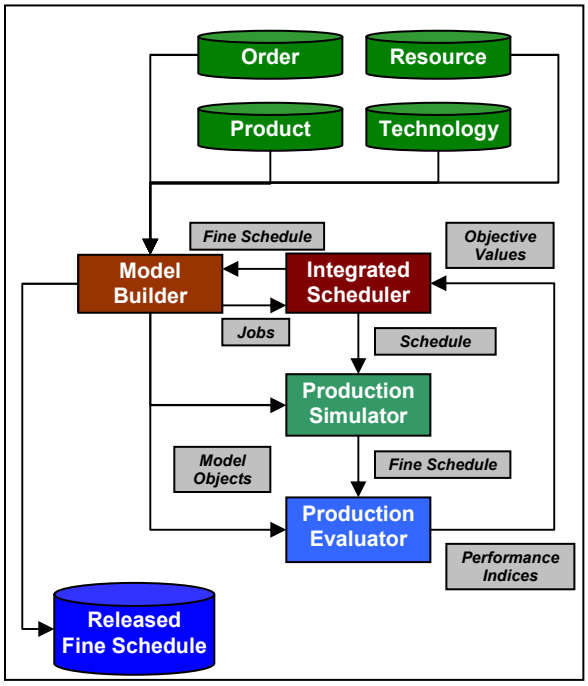

Fig. 2. Integrated Production Scheduler.

For solving the scheduling problem in an integrated form addressed above, we use an advanced tabu search algorithm based on simultaneous production simulation, overloaded relational operators, multiple neighbouring operators and special tabu list which stores schedules in coded form.

To accelerate the computation we use indexed arrays in our model. In these arrays, there are no full length identifiers and attributes of entities (i.e.: orders, jobs, machines, routes and so on), instead there are indexes, which are non-negative integer values assigned to the entities, to point to the position of the target object in the base array. In any of indexes of a given array, we can use any value of the same array or another array. The developed data structure supports the association of two or more different type arrays. The model builder creates the full indexed data model which includes the possible technology and resource alternatives.

In our model the product-pallet plays the role of the basic scheduling unit. Each production order is divided into pallets that mean individual jobs. The production batch sizes are formed dynamically by scheduling the jobs on machines.

In order to create a detailed schedule for the production of each order, it is necessary for each job: 1.) to assign to one of the possible routes, 2.) to assign to one of the possible machines at each possible machine group according to selected route, 3.) to fix its position in the queue of each selected machine, and 4.) to fix its starting time on each selected machine.

Different heuristic procedures have been developed to solve the problem. These procedures are integrated into a scheduling engine (integrated scheduler). At present, two kinds of classes of heuristic algorithms are used in two phases. In the first phase, constructive algorithms based on combined heuristic priority rules create good initial solutions in short time. In the second phase, iterative searching algorithms improve the best solution according to the multiple objectives.

This paper outlines one of these scheduling algorithms based on tabu search technique (TS). TS principle was first suggested by Glover, (1990). It is a heuristic problem independent optimization method. Our tabu search variant is a special search procedure which iteratively moves from a schedule s_x to a schedule s_y in the neighbourhood of s_x, until some stopping criterion has been satisfied. To explore regions of the search space that would be left unexplored by a simple local search procedure and escape local optimality, tabu search modifies the neighbourhood structure of each schedule as the search progresses. Our tabu list contains the schedules that have been visited in the recent past (less than a given number of moves ago). Schedules in the tabu list are excluded from the neighbourhood of the actual solution.

```
Multi-Objective Tabu Search (Parameter List)
{
 s0 ← Generate an initial solution ();
 s* ← s0;
 TabuList ← NULL;
 while ( Stop criterion is not satisfied () )
 {
  while ( Extension criterion is satisfied () )
  {
    s ← Generate a neighbour solution (s0);
    if ( TabuList do not include ( s ) )
    {
     Insert tabu into the first position of TabuList ( s );
     if ( Number of tabus () > Max number () )
       Delete tabu from the last position of TabuList( );
     if ( This is the first solution of the extension ( s ) )
         s_k ← s;
     else
         if ( F( sk, s) < 0 )
           sk ← s;
    }
  }
  s0 ← sk;
  if ( F( s*, sk ) < 0 )
    s* ← sk;
 }
 return s*;
}
```

A certain number of neighbours of the current schedule are generated random successively by using

priority controlled neighbouring operators. The applied neighbouring operators are as follows:

- *N1* operator removes an order randomly chosen from the schedule then inserts all the jobs of the order as a whole.
- *N2* operator removes a late order randomly chosen from the schedule then inserts all the jobs of the order as a whole.
- *N3* operator modifies the sequence of jobs on a randomly chosen machine by using random length permutation-cycle.
- *N4* operator removes a late job randomly chosen from the schedule then redefined the manufacturing tasks of the job by assigning resources and finally inserts the job into random position on each related machine.
- *N5* operator works similarly to the operator N4 but the target job is chosen from all jobs.

These operators create new feasible schedules by modifying resource allocations and job sequences. It is not necessary to check the feasibility of the generated solutions because the neighbouring operators make valid modifications by choosing allowed alternatives from the indexed model structure.

The objective functions concerning schedules are evaluated by a production simulator which represents the machines with their capacity and the technological constraints. The simulation means rule based numerical tracking of the production to calculate the time data of the manufacturing steps (starting time, setup time, processing time and finishing time). The simulator extends the predefined schedule (job-resource assignments and job sequences on machines) to a fine schedule by calculating and assigning time data. The simulation is able to transform the original searching space to a reduced space by solving the timing problem.

Production evaluator measures the performance of the fine schedules by calculating management indices based on job, order and machine data. The function $F(s_x,s_y)$ proposed in Section 2 is used to compare the generated schedules according to multiple objectives described in Section 3.2. The best schedule becomes the initial solution of the next loop. When the scheduling process has been finished or stopped by the user, the currently best known schedule is returned, so the method can be used in *any-time* working model.

To increase the flexibility and effectiveness of the scheduling process, we developed an advanced software module for supporting the user interactions. Graphical user interface provides useful charts, diagrams, tables and reports to show aggregate and detailed information of the production fine schedule. Scheduling process is also scrutinized and checked on the screen, the user can modify at runtime the control priority and parameters of searching algorithms. In addition, the user can suspend the automatic process and edit the actual schedule by using available operation tools manually. Similarly to the neighbouring operators, the usage of the manual planning tools can produce only valid and feasible solutions.

The outlined mixed scheduling approach based on human and artificial intelligence can yield significant performance improvements.

4. RESCHEDULING PROCESS AS A TOOL FOR SUPPORTING UNCERTAINTY MANAGEMENT

In practice, generating high quality predictive schedule according to multi-objectives is not enough because many unexpected events require the revision and modification of the released schedule. In the following, we shortly deal with the role of the shop floor rescheduling.

Rescheduling is a process of updating an existing production schedule in response to disruptions or creating a new one if the current schedule has become infeasible. Different type of uncertainty may occur e.g. machine failure or breakdown, missing materials or components, underestimation of processing time, job priority or due date changes and so on (see in detail i.e. Aytug, *et al*., 2005). Different rescheduling methods can be used according to the effects of the unexpected events: time shift rescheduling, partial rescheduling or complete rescheduling. Time shift rescheduling postpones executions of certain tasks and jobs in time, but their resource assignments and sequences are not changed. Partial rescheduling modifies only jobs and resources affected by the disruption. Complete rescheduling generates a new feasible schedule. These methods are presented in detail by Vierira, *et al*., (2003).

In order to solve the EFFS rescheduling problem we use multi-objective searching algorithms similarly to the predictive scheduling described in section 3. The aim of rescheduling is to find a schedule which 1.) considers the modified circumstances, 2.) is near-optimal according to some predefined criterion and 3.) is as close as possible to the original one.

Rescheduling methods are required to consider new demands added to predictive scheduling problem. The last released schedule appears as a new input element of the rescheduling system and it is very important to preserve this initial schedule as much as possible to maintain the system stability. For this purpose, we defined qualitative indices (i.e. related to setup and due date) for supporting comparison of schedule changes. For examples we used the following indices:

- Number of the jobs having modified execution route.
- Number of the jobs having at least one modified parallel machine.
- Number of jobs having late status deviation.
- Number of the new late jobs.
- Number of the orders which have at least one job with modified execution route.
- Number of the orders which have at least one job with at least one modified parallel machine.
- Number of orders having late status deviation.
- Number of the new late orders.
- Number of machines with setup deviation.
- Number of machines with new setup.

Such key performance indices can be considered as objective functions of rescheduling and they can be used by multi-objective scheduling methods proposed in the paper.

Lots of special rescheduling constraints have to be considered in order to satisfy additional demands. Some of these are detailed as follows:

- All jobs which are already finished when the rescheduling process starts are not changeable but can affect the other jobs and orders.
- The manufacturing tasks of jobs running at the starting time of the rescheduling process must not be interrupted and their possible execution route and parallel machines (and other alternatives) can differ from the original possibilities.
- Finished and running jobs on resources are known therefore they have to be regarded in the future.
- All production orders starting after the rescheduling process can be considered in their original status.

To satisfy such constraints we use *freezing* techniques. The main classes of these techniques are as follows: 1.) to freeze jobs, 2.) to freeze production orders and 3.) to freeze machines. Using these techniques the priority controlled neighbouring operators can work effectively without violation of the rescheduling constraints.

The rescheduling problem can be solved by applying multi-objective searching algorithm with the proposed extensions. Similarly to the predictive scheduling, the rescheduling process can be controlled and influenced by the user.

5. CONCLUSIONS

This paper described the proposition and application of a new method for multi-objective scheduling and rescheduling problems. It is based on new interpretation and usage of relational operators for schedules in searching algorithms. After developing computer program, the concept is successfully tested on extended flexible flow shop scheduling and rescheduling problems under multiple objectives.

The obtained results and the problem independent nature of the approach are encouraging to the application of the method in other multi-objective optimization problems.

ACKNOWLEDGEMENTS

The result of research and development summarized in this paper are connected with the NODT project entitled “VITAL” (National Office for Development and Technology founded by the Hungarian Government, Grant No.: 2/010/2004, project leader: László Monostori DSc). The authors express their thanks for the support.

REFERENCES

Aytug, H., M. A. Lawley, K. Mckay, S. Mohan and R. Uzsoy (2005). Executing Production Schedules in the Face of Uncertainties: A Review and some Future Directions. *European Journal of Operational Research*, 161, pp. 86-110.

Baykasoğlu, A., L. Özbakir and T. Dereli (2002). Multiple Dispatching Rule Based Heuristic for Multi-Objective Scheduling of Job Shops Using Tabu Search. *Proceedings of the 5th International Conference on Managing Innovations in Manufacturing*, pp. 396-402.

Glover, F. (1990). Tabu Search: a Tutorial. *Interfaces*, **20**, pp. 74-94.

Kis, T. and E. Pesch (2005). A Review of Exact Solution Methods for the Non-Preemptive Multiprocessor Flowshop Problem. *European Journal of Operational Research*, **164**, pp. 573-695.

Linn, R. and W. Zhang (1999). Hybrid Flow Shop Scheduling: A Survey. *Computers and Industrial Engineering*, **37**, pp. 57-61.

Loukil, T., J. Teghem and D. Tuyttens (2005). Solving Multi-Objective Production Scheduling Problems Using Metaheuristics. *European Journal of Operational Research*, **161**, pp. 42-61.

Quadt, D., and H. Kuhn (2007). A Taxonomy of Flexible Flow Line Scheduling Procedures. *European Journal of Operational Research*, **178**, pp. 686-698.

Sbalzarini, L. F., S. Müller and P. Koumoutsakos (2000). Multiobjective Optimization Using Evolutionary Algorithms. *Center of Turbulence Research, Proceedings of the Summer Program 2000*, pp. 63-74.

Smith, K. L., R. M. Everson and J. E. Fieldsend (2004). Dominance Measures for Multi-Objective Simulated Annealing. *Proceedings of Congress on Evolutionary Computation*, pp. 23-30.

Vieira, G., J. Hermann and E. Lin (2003). Rescheduling Manufacturing Systams: A Framework of Strategies, Policies and Methods. *Journal of Scheduling*, **6**, pp. 35-58.

Wang, W. (2005). Flexible Flow Shop Scheduling: Optimum, Heuristics, and Artificial Intelligence Solutions. *Expert Systems*, **22**, pp. 78-85.

Yamada, T. (2003). Studies on Metaheuristics for Jobshop and Flowshop Scheduling Problems. *Ph.D Thesis*, Kyoto University, Japan.

Zhu, X. and W. E. Wilhelm (2006). Scheduling and Lot Sizing with Sequence-Dependent Setup: A Literature review. *IIE Transactions*, **38**, pp. 987-1007.