Scheduling Support System based on Meta-Heuristics for Dynamic Manufacturing Scheduling

ANA MADUREIRA

Computer Science Department
Institute of Engineering - Polytechnic of Porto
GECAD – Knowledge Engineering and Decision Support Research Group
Porto, Portugal
http://www.dei.isep.ipp.pt/~anamadur/

Abstract: - The planning of Manufacturing Systems involves frequently the resolution of a huge amount and variety of combinatorial optimization problems with a important impact on the performance of manufacturing organizations. The proposed approach is in line with reality and away from the approaches that deal with static and classic or basic Job-Shop scheduling problems. In fact, in real world, where problems are essentially of dynamic and stochastic nature, the traditional methods or algorithms are of very little use. This is the case with most algorithms for solving the so-called static scheduling problem for different setting of both single and multi-machine systems arrangements. This reality, motivated us to concentrate on tools, which could deal with such dynamic, disturbed scheduling problems, for multi-machine manufacturing settings, even though, due to the complexity of these problems, optimal solutions may not be possible to find.

Key-Words: - Scheduling, Dynamic Scheduling, Tabu Search, Meta-Heuristics, Manufacturing

1 Introduction

The ability to deal with a larger number of smaller quantity orders increases the stress on the scheduling process. A dynamic scheduling system takes into consideration what equipment and inventory is required and available to fulfill an incoming order. In doing so, equipment is better utilized and work-in-progress dramatically reduced.
The planning of Manufacturing Systems involves frequently the resolution of a huge amount and variety of combinatorial optimisation problems with a important impact on the performance of manufacturing organisations. Examples of those problems are the sequencing and scheduling problems in manufacturing management, routing and transportation, layout design and timetabling problems. The classical optimisation methods are not enough for the efficient resolution of those problems or are developed for specific situations.
The interest of the Meta-Heuristic (MH) approaches based on Local Search is that they converge, in general, to satisfactory solutions in an effective and efficient way (computing time and implementation effort).
There have been significant advances in the theory and application of Meta-Heuristics to solve hard optimization problems. Meta-Heuristics form a class of powerful and practical solution techniques for tackling complex, large-scale combinatorial problems. The family of Meta-Heuristics includes, but it is not limited to Tabu Search, Simulated Annealing, Soft Computing, Evolutionary Methods, Adaptive Memory procedures, Ant Systems, Scatter Search and their hybrids.
Meta-Heuristics approaches have proved to be a very effective tool for finding good approximate solutions for difficult scheduling and optimization problems arising in industrial, economic, and scientific domains.
Many real-world optimization problems take place in environments constantly changing. Because of this, optimization algorithms instead of looking for single optimal solutions should continuously track the optimum through time. Real world scheduling requirements are related with complex systems operated in dynamic environments. This means that they are frequently subject to several kinds of random occurrences and perturbations, such as new job arrivals, machine breakdowns, employees sickness, jobs cancellation and due date and time processing changes, causing prepared schedules becoming easily outdated and unsuitable. Scheduling under this environment is known as dynamic.
Tabu search has traditionally been used on combinatorial optimization problems. The technique is straight-forwardly applied to continuous functions by choosing a discrete encoding of the problem.
Most of the heuristic techniques or approximation methods proposed for Job-Shop Scheduling problems

are tailored techniques, i.e. developed specifically for a problem in consideration. There is a need to develop methods robust and flexible capable of being applicable not only to a specific problem and environment but also to a variety of scheduling problems and environments. Most research in scheduling focuses on optimization of static and dynamic deterministic problems where all problem data are known before scheduling starts. How-ever many real world optimization problems tend to be complex and non-deterministic, with random changes occurring continually.

The purpose of this paper is to describe an framework based on Tabu Search for solving a class of real world scheduling problems, where the products (jobs) to be processed have due dates, release times and different assembly levels. This means that parts to be assembled may be manufactured in parallel, i.e. simultaneously. Therefore, in this work, we define a job as a manufacturing order for a final item, simple or complex. It may be simple, like a part, requiring a set of operations to be processed. We call it a simple product or simple final item. Complex final items, requiring processing of several operations on a number of parts followed by assembly operations at several stages, are also dealt with.

We adopt a strategy to scheduling known as Re-source-Oriented Scheduling (ROS). This is related with vertical loading [1], but much more detailed in order to define start and completion times of jobs in the available processors. In ROS [1] each job is taken in turn for scheduling on a resource or machine at a time. Although any objective function may be considered, resource-oriented scheduling has an underlined aim of getting good use of resources. An alternative, and somewhat opposite strategy to scheduling, is horizontal or Product-Oriented Scheduling (POS) [1]. This strategy completely schedules a job, from first to last operation, on the necessary resources or machines, before considering other jobs for scheduling.

Our approach, implementing ROS, starts by solving a surrogated set of a Single Machine Scheduling Problems (SMSP), whose solutions are repaired, if necessary, by means of a suitable mechanism, in order to arrive to feasible solutions for the original problem[2].

The remaining sections are organized as follows: sec-tion 2 provides a description of the scheduling problem under consideration. Section 3 describes the proposed approach for dynamic scheduling problems. Section 4 briefly describes Tabu Search (TS) and the parameterization of the TS algorithm. Section 5 presents some computational results and the performance of the Tabu Search based method on the resolution of a set of academic instances of the Job-Shop Scheduling problem. Finally, the paper presents conclusions and some ideas for future work.

2 Problem Definition

The scheduling problem can be seen as a decision making process for operations starting and resources to be used. A variety of characteristics and constraints related with jobs and production system, such as operation processing time, release and due dates, precedence constraints and resource availability, can affect scheduling decisions. For literature on this subject, see for example [5] [9] [13][15].

If all jobs are known before processing starts a scheduling problem is said to be static, while, to classify a problem as dynamic it is sufficient that job release times are not fixed at a single point in time, i.e. jobs arrive to the system at different times. Scheduling problems can also be classified as either deterministic, when processing times and all other parameters are known and fixed, or as non-deterministic or stochastic, when some or all parameters are uncertain [15].

The general Job-Shop Scheduling Problem (JSSP) of size $n \times m$ can generally be described as a decision-making process concerning about the allocation of a limited set of $m=\{0, 1, ...,m\}$ resources over time to perform a set of $n=\{0, 1, ..., n\}$ tasks or jobs. Most real-world multi-operation scheduling problems can be described as dynamic and extended versions of the classic or basic Job-Shop scheduling combinatorial optimization problem. In a Job-Shop each job has a specified processing order through the machines, i.e. a job is composed of an ordered set of operations each of which is to be processed, for certain duration, on a machines. In the basic Job-Shop scheduling problem several constraints on jobs and machines are considered []

In practice, many scheduling problems include further restrictions and relaxation of others [7]. Thus, for example, precedence constraints among operations of the different jobs are common because, most of the times, mainly in discrete manufacturing, products are made of several components that can be seen as different jobs whose manufacturing must be coordinated. Additionally, since a job can be the result of manufacturing and assembly of parts at several stages, different parts of the same job may be processed simultaneously on different machines.

Therefore, in this work, we define a job as a manufacturing order for a final item, that could be **Simple** or **Complex**. It may be simple, like a part, requiring a set of operations to be processed. We call it a **Simple Product** or **Simple Final Item**. **Complex**

Final Items, requiring processing of several operations on a number of parts followed by assembly operations at several stages, are also dealt with.
Moreover, in practice, scheduling environment tend to be dynamic, i.e. new jobs arrive at unpredictable intervals, machines breakdown, jobs are cancelled and due dates and processing times change frequently. This non-basic JSSP, focused in our work, which we call Extended Job-Shop Scheduling Problem (EJSSP), has major extensions and differences in relation to the classic or basic JSSP. The existence of operations on the same job, on different parts and components, processed simultaneously on different machines, followed by components assembly operations, which characterizes EJSSP, is not typical of scheduling problems addressed in the literature. However, such is common in practice. This approach to job definition, emphasizing the importance of considering complex jobs, which mimic customer orders of products, is in accordance with real world scheduling in manufacturing.

5 MH based Scheduling System

A Job-Shop like manufacturing system has a natural dynamic nature observed through several kinds of random occurrences and perturbations on working conditions and requirements over time. For this kind of environment it is important the ability to efficient and effectively adapt, on a continuous basis, existing schedules according to the referred disturbances, keeping performance levels. The application of Tabu Search to the resolution of this class of real world scheduling problems seems really promising. Although, most of the known work on scheduling deals with optimization of classic Job-Shop scheduling problems, on static and deterministic environments.
This work is concerned with the resolution of real world scheduling of simple or complex products, comprehending the parts fabrication and their multistage assembly, in dynamic environments, both deterministic and non-deterministic. This class of problems, focused in this work, was named Extended Job-Shop Scheduling Problem.
In the approach proposed here, a Meta-Heuristic are used, namely Tabu Search, using a strategy known as Resource Oriented Scheduling. The Dynamic Extended Job-Shop Scheduling Problem is decomposed into a series of deterministic Single Machine Scheduling Problems (SMSP), which are solved one at a time, consecutively, by a Meta-Heuristic (Tabu Search). The obtained individual solutions are then integrated into the main problem. So, some mechanisms and algorithms were developed, jointly named MEM (Meta-Heuristics based Scheduling Method).
Based on the MEM method and on a simple architecture, a Scheduling System for Dynamic Non-Deterministic Problems (SEPDynNDET) is proposed. The purpose of SEPDynNDET is to extend the resolution of the deterministic problem to the non-deterministic one, in which changes may occur continually. This takes into account dynamic occurrences in a manufacturing system and adapts the current schedule.
The implemented approach on the SEPDynNDET system allows obtaining a predictive schedule, based on the MEM method, which is dynamically adapted through an integration procedure selected according to the occurred disturbances.

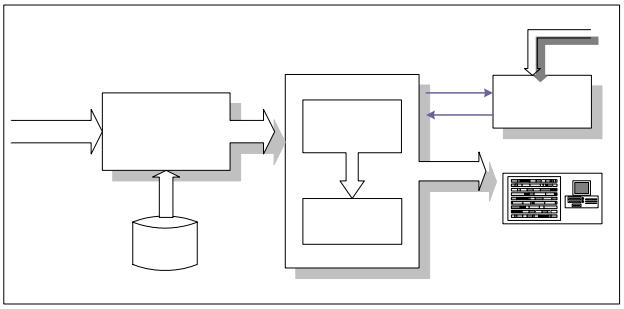

Figure 1 - SEPDynNDET Scheduling System

The scheduling approach presented in this section is applicable to both static and dynamic manufacturing environments for any optimizing criteria that are possible to establish (Figure 1). It is based on the decomposition of the problems into SMSP, one for each machine involved in processing, and later integration for obtaining a solution to the original problem, i.e. to the Extended JSSP
SEPDynNDET consists on a centralised TS-based scheduling system. One advantage of this is that the solutions (schedules) planned for single machines guarantees some consistency of manufacturing.

5.1 TS based Scheduling Method

Initially, we start by decomposing the deterministic EJSSP problem into a series of deterministic Single Machine Scheduling Problems. We assume the existence of different and known job release times r_j, prior to which no processing of the job can be done and, also, job due dates d_j. Based on these, release dates and due dates are determined for each SMSP and, subsequently, each such problem is solved independently by the Tabu Search. Afterwards, the solutions obtained for each SMSP are integrated to obtain a solution to the main problem instance, i.e. the instance of the EJSSP [1].

1st PHASE	**Finding a 1st job shop schedule based on single machine scheduling problems**
Step 1	Define completion time estimates (due dates) for each operation of each job.
Step 2	Define the interval between starting time estimates (release times) for all operations of each job.
Step 3	Define all SMSP 1| r_j |C_{max} based on information defined on Step1 and Step 2.
Step 4	Solve all SMSP 1| r_j |C_{max} with those release times and due dates using a TS.
Step 5	Integrate all the obtained near-optimal solutions into the main problem.
2nd PHASE	**Check feasibility of the schedule and, if necessary apply the coordination mechanism**

Figure 2 - TS based scheduling method

The completion due times for each operation of a job are derived from job due dates and processing times by subtracting the processing time from the completion due time of the immediately succeeding job operation:

$$\mathbf{C_{ijk-1l} = C_{ijkl} - p_{ijkl}} \quad \mathbf{(1)}$$

This procedure begins with the last job operation and ends with the first. When an operation is precedent to more than one operation, i.e. there exists a multilevel structure, the completion due time is the lower value as defined on equation 3.

$$C_{ijkl-1} = \min\{C_{ijkl} - p_{ijkl}\}\ l > l-1 \quad (2)$$

The operations starting due time intervals $[t_{ijkl},T_{ijkl}]$ are also defined considering the job release times and the operation processing times. The earliest starting time t_{ijkl} corresponds to the time instant from which the operation processing can be started. The latest starting time T_{ijkl} correspond to the time at which the processing of the operation must be started in order to meet its completion due time (due date). This means that no further delay is allowed. When an operation has more than one precedent operation, i.e. there exists a multilevel structure, the interval $[t_{ijkl},T_{ijkl}]$ is the interval intersection from precedent operations correlated by the respective processing times.

The starting time interval (STI) for operations without precedents is defined as follows:

$$STI_{ijkl} = [r_{ijkl}, C_{ijkl} - p_{ijkl}] \quad (3)$$

The starting time interval of an operation with one precedent operation is defined by equation.

$$STI_{ijkl} = [t_{ijk-1l}, T_{ijk-1l}] + p_{ijk-1l} \quad (4)$$

The starting time interval of an operation with more than one precedent operation is the intersection interval of the starting time intervals from all precedent operations correlated by the respective processing times (Equation 5).

$$\mathbf{STI_{ijkl} = [t_{ijK1L}+p_{ijK1L},\ T_{ijK1L}+p_{ijK1L}] \cap t_{ijK2L}+p_{ijK2L},\ T_{ijK2L}+p_{ijK2L}] \cap \ldots \cap t_{ijKnL}+p_{ijKnL},\ T_{ijKnL}+p_{ijKnL}]}$$
$$\text{with } K_n<k \text{ e } L<l \quad (5)$$

Using this procedure it is easy to deal with situations were an operation has more than one precedent operation, i.e. there exists a multilevel task structure involving assembly operations. In this case the previous operations may be processed simultaneously, and therefore a relaxation of an underlining characteristic of the EJSSP problems is assumed. This situation is typical of real world manufacturing requirements. This means that a more generalized and realistic problem is dealt with the scheduling approach adopted in this work.

At this stage, only technological precedence constraints of operations and job due dates will be considered for defining completion and starting times.

The release date r_j correspond to the earliest starting times of each operation. The due date d_j correspond to the opera-tion completion times. The notation r_j and d_j used at this point, considers that we are dealing with single machine problems.

The integration of the SMSP solutions may give an unfeasible schedule to the EJSSP. This is why schedule repairing may be necessary to obtain a feasible solution. The repairing mechanism named Inter-Machine Activity Coordination Mechanism (IMACM) carries this out.

5.2 Coordination Mechanism

The repairing is carried out through coordination of machines activity, having into account job operation precedence and other problem constraints. This is done keeping job allocation order, in each machine, unchanged. The IMACM mechanism (Figure 3) establishes the starting and the completion times for each operation. It ensures that the starting time for each operation is the highest of the two following values: the completion time of the immediately precedent operation in the job, if there is only one, or the highest of all; and the completion time of the immediately precedent operation on the machine.

Step 1	Coordinate the machine activities, starting by initial operations (level 1), i.e. operations without precedents. At this level the starting and completion times are the defined by TSScheM method.
Step 2	Next, we consider all operations with precedents already scheduled $t_{ijkl} \leftarrow max(C_j,\ Tconcl_j)$ were $Tconcl_j$ is the completion time of operation j, which is precedent on machine i, and C_j is the completion time of the precedent operation on the job.
Step 3	The procedure (step 2) will be repeated until all operations have been scheduled

Figure 3 - Coordination Mechanism

Most of the research on JSSP focuses on basic problems as described above. The method developed and just described is in line with reality and away from the approaches that deal solely with static and classic or basic job-shop scheduling problems. Thus, the method is likely to perform worse than the best available algorithms found for such problems. However, it is not our purpose, neither it would be reasonable, to rate our method against such good performing algorithms for academic and basic JSSP. Our aim is to provide an efficient tool, which we think we managed with our method, for obtaining good solutions, for a variety of criteria, for many real world scheduling problems, i.e. complex non-basic JSSP as described above, which we named Extended JSSP. For these problems, the referred best performing algorithms are unable to give solutions. Further, through the survey we made to the literature we were unable to find methods to solve the EJSSP as here described.

5.3 Rescheduling Mechanism

For non-deterministic problems some or all parameters are uncertain, i.e. are not fixed as we assumed in the deterministic problem. Non-determinism of variables has to be taken into account in real world problems. For generating acceptable solutions in such circumstances our approach starts by generating a predictive schedule, using the available information and then, if perturbations occur in the system during execution, the schedule may have to be modified or revised accordingly, i.e. rescheduling is performed. Therefore, in this process, an important decision must be taken, namely that of deciding if and when should rescheduling happen.

In SEPDynNDET, rescheduling is necessary due to two classes of events [2]: **Partial events** which imply variability in jobs or operations attributes such as processing times, due dates and release times; and **Total events** which imply variability in neighbourhood structure, resulting from either new job arrivals or job cancellations.

While, on one hand, partial events only require redefining job attributes and re-evaluation of the objective function of solutions, total events, on the other hand, require a change on solution structure and size, carried out by inserting or deleting operations, and also re-evaluation of the objective function. Therefore, under a total event, the modification of the current solution is imperative. In this work, this is carried out by mechanisms described in [3] for SMSP.

Considering the processing times involved and the high frequency of perturbations, rescheduling all jobs from the beginning should be avoided. However, if work has not yet started and time is available, then an obvious and simple approach to rescheduling would be to restart the scheduling from scratch with a new modified solution on which takes into account the perturbation, for example a new job arrival. When there is not enough time to reschedule from scratch or job processing has already started, a strategy must be used which adapts the current schedule having in consideration the kind of perturbation occurred.

The occurrence of a partial event requires redefining job attributes and a re-evaluation of the schedule objective function. A change in job due date requires the re-calculation of the operation starting and completion due times of all respective operations. However, changes in the operation processing times only requires re-calculation of the operation starting and completion due times of the succeeding operations. A new job arrival requires definition of the correspondent operation starting and completion times and a regenerating mechanism to integrate all operations on the respective single machine problems. In the presence of a job cancellation, the application of a regenerating mechanism eliminates the job operations from the SMSP where they appear. After the insertion or deletion of positions, neighbourhood regeneration is done by updating the size of the neighbourhood and ensuring a structure identical to the existing one. Then the scheduling module can apply the search process for better solutions with the new modified solution.

5 Computational Study

In order to evaluate the performance of the proposed TS based scheduling system some computational tests were performed. It is important to refer that we can not find other results obtained by others methods which dealing with the same constraints For that, some relaxations were considered in order to simulate identical scenarios. For our experiments, we consider some benchmark problems [16]. For release dates, we consider zero for all in-stances. Due dates are considered to be the optimal makespan value. The obtained results with our method based on Tabu Search (TS) are compared with those obtained from using the General Shifting Bottleneck method (SB) imple-mented in the LEKIN scheduling tool [8].

With a simple implementation of the Tabu Search algorithm and a small parameterisation effort it was possible to achieved good performance for most instances of the problem having the Shifting Bottleneck results in consideration (Table 8). Thus, from a global analysis the Shifting Bottleneck method

was the most effective on Cmax, in most instances, and the Tabu Search the most effective on total tardiness (∑Tj) and total number of late jobs (∑Uj). In relation to the last two, it was best in all instances tested, excepted the first. Tabu Search presents an average relative deviation of 9.7% from optimal Cmax value, while for the Shifting Bottleneck is 8.1%. Comparing the efficiency, TS and SB have an aver-age computational time of 8.1 and 8.9 seconds, respectively. Additionally, both methods achieve the optimal solution on the instance La05.

It is important to refer that our scheduling framework, which here uses TS, is flexible in several ways. It is prepared to use other Local Search Meta-Heuristics and to drive schedules based on practically any performance measure. Moreover, the framework is not restricted to a specific type of scheduling problem, as is the case Shifting Bottleneck.

One novel approach, rarely addressed in the literature, but very important in practice, is considered in our scheduling framework, namely that of being able to schedule jobs with complex processing structures, i.e. with both parallel proc-essing of product component parts followed by their assembly at several stages and a given job may pass on a given machine more than once, i.e., a recirculation is permitted. Additionally, the jobs are subject to release dates and due dates.

7 Concluding Remarks

The work reported in this paper is concerned with the resolution of realistic Job-Shop Scheduling Problems, called here Extended Job-Shop Scheduling Problems (EJSSP). Moreover, it is concerned with integrated scheduling of jobs which are products composed by several parts or components that may be submitted to a number of manufacturing and multi-level assembly operations. We propose a scheduling system, having two main pieces of intelligence, to solve such complex problems. One such piece is a TS-based method for deterministic scheduling. This includes a Tabu Search method for single machine problems and an inter-machine activity coordination mechanism that attempts to ensure a good feasible solution for the deterministic EJSSP. The other piece is a rescheduling mechanism that includes a method for neighborhood regeneration under dynamic environments, increasing or decreasing it according new job arrivals or cancellations.

We adopt a strategy to scheduling known as Re-source-Oriented Scheduling. In Resource-Oriented Scheduling each a job is taken in turn for scheduling on a resource or machine at a time. Although any objective function may be considered, resource-oriented scheduling has an underlined aim of getting good use of re-sources.

Using Tabu Search we achieved good solutions, for the problem instances tested. The results were, in general, as good as, and for some measures better than, those obtained through the Shifting Bottleneck method. However, we recognize the need for further testing, particularly to evaluate the suitability of the proposed framework and mechanisms for the EJSSP under dynamic environments. We also recognize that this is not an easy task because it is difficult to find, in the literature, test problems with the job structure that we selected and think important, in practice. This structure assumes a job made of several parts processed and assembled through several assembly operations and stages.

References:

[1] Ana Madureira, Carlos Ramos, Silvio C. Silva, Toward Dynamic Scheduling Through Evolutionary Computing, *WSEAS Transactions on Systems*, Issue 4, Volume 3, ISSN 1109-2777, pp. 1596-1604, 2004.

[2] Ana M. Madureira, Carlos Ramos and Sílvio do Carmo Silva, A Genetic Algorithm for The Dynamic Single Machine Scheduling Problem, In *4th IEEE/IFIP Intl. Conf. on Information Technology for Balanced Automation Systems in Production and Transportation*,(Germany), pp. 315-323, 2000.

[3] Ana M. Madureira, *Meta-Heuristics Application to Scheduling in Dynamic Environments of Discrete Manufacturing*, PhD Thesis, University of Minho, 2003 (in portuguese).

[4] Fred Glover, Future paths for Integer Programming and Links to Artificial Intelligence, *Computers and Operations Research*, 5:533-549, 1986.

[5] J. Blazewicz, K. H. Ecker, E. Pesch, G. Smith and J. Weglarz, *Scheduling Computer and Manufacturing Processes*, Springer, 2nd edition, New York, 2001.

[6] Joseph Adams, Egon Balas and Daniel Zawack, The Shifting Bottleneck Procedure for Job Shop Scheduling, *Management Science*, Vol. 34, nº 3, USA, 1988.

[7] M. C. Portmann, *Scheduling Methodology: optimization and compu-search approaches, in the planning and scheduling of production sys-tems*, Chapman &Hall, 1997.

[8] M. Pinedo, *Scheduling – Theory, Algorithms and Systems*, 2nd edition, Prentice-Hall, 2002.

[9] *Intelligent Manufacturing*, 14(5), Kluwer Academic Publishers, pp.441-455, 2003.

[10] Peter Brucker, Scheduling Algorithms, Springer, 3rd edition, New York, 2001.

[11] Report from EVONET Working Group on Evolutionary Approaches to Timetabling and Scheduling, *The state of the art in evolutionary Approaches to timetabling and scheduling*, 1999.

[12] S. French, *Sequencing and Scheduling: An introduction to the Mathematics of the Job Shop*, Ellis Horwood, Chichester, 1982.

[13] S. Jain and S. Meeran, Deterministic Job Shop scheduling: past, present and future, *European Journal of Operational Research*, nº113, 390-434, 1999.

[14] Thomas E. Vollmann, William Berry, and D. Clay Whybark, *Manufacturing Planning and Control Systems*, McGraw-Hill, New York, 1997.