Product Oriented Scheduling through Job Scheduling Patterns

Ana Almeida*, Carlos Ramos*, Sílvio do Carmo Silva†
*Instituto Superior de Engenharia do Porto, GECAD &, Dept. Engenharia Informática
Rua São Tomé, 4200 Porto – Portugal, Phone: 351 - 2 – 8340500
†Universidade do Minho, Dept. de Produção e Sistemas
Gualtar, Braga – Portugal, Phone: 351 – 53 – 604100
email: ana@dei.isep.ipp.pt , csr@dei.isep.ipp.pt, scarmo@dps.uminho.pt

Abstract. This paper is concerned with product oriented detailed scheduling of job-shop like manufacturing systems. It addresses the scheduling of jobs, either simple, requiring the manufacture of a batch of parts, i.e. simple products, or complex, comprehending the parts fabrication and their multistage assembly into a batch of products. The horizontal scheduling approach was adopted, assuming that full scheduling of a simple or complex job, based on the job routing network of operations, from the first operation to the last, is performed before another job is considered for scheduling, having in consideration existing manufacturing processors and their availability. We followed this approach because we aimed at compressing job flow time as a strategy to meeting job due dates. To further enhance this objective the idea behind Simultaneous Manufacturing was implemented. In particular, the widespread use of batch overlapping was implemented, which proved particularly effective in reducing job throughput time, maintaining operating simplicity and requiring reduced coordination.

Key words: Manufacturing, Horizontal Scheduling, and Extended Job-Shop.

Introduction

It is widely accepted that manufacturing scheduling problems are generally difficult or hard to solve to optimality (Baker, 1974), (French, 1982) and (Blazewicks, 2001). This has not been an argument to abandon efforts for finding methods that can achieve optimum or near optimum solutions, since much work appeared, during the last decades, on this matter. However, for most real world problems, found in practice, this can be a very difficult and, many times, an impossible job. This is so because the number of variables tends to increase in relation to theoretical formulated problems, which are already hard and, additionally, because the environment is predominantly dynamic and non-deterministic (French, 1982), which makes things even more difficult. This means that, in a given moment, an established schedule, even an optimum one, for some particular scheduling instance, soon becomes invalid and may have to be scraped or adapted. Therefore, methods for such uncertain environments and complex problems usually tend to be of heuristic nature, involving frequently simple mechanisms, designed for getting good or acceptable solutions, rarely optimum ones, in the short available time for taking decisions.

The most common academic approach to scheduling considers static and basic problems (Portmann, 1997). However, these problems are not very common in practice.
We address practical problems in job-shop like environments. For this we use both simple priority rules and some analytical procedures, which take into account actual time availability and unavailability of manufacturing processors, i.e. their agendas, every time scheduling is to be done.
A job as defined in this work, is typical of reality but untypical in theoretical scheduling problems addressed in the literature. Our approach considers a job to be a manufacturing order for a number of identical products. These may be simple, i.e. having a set of operations to be carried out in a given sequence in a batch of identical parts, called *simple products*. The products may also be *complex products*, having a set of operations carried out in a given sequence in a batch of products, each one made of several different parts. The manufacture of these complex products is typical of industries of type A

and T as defined by Chase and Aquilano (1998). It is mainly, but not only, for these types of industries that the scheduling approach and methods presented in this paper are most appropriate. A job is specified by the quantity of products necessary, i.e. its batch size, the manufacturing operations and their precedence relationship, processing times of operations and the manufacturing processors to be used. Processing times include auxiliary time elements such as set up time and time for parts handling at a machine. The most generic job requires several operations on identical and different parts followed by assembly operations. A typical job is, for example, the manufacture of an ordered quantity of identical chairs. The manufacture of a certain amount of a specific component of a chair is not a job.

We strongly explore the possibility of different batch overlapping and batch splitting schemes for finding good schedules. While the former is easy to implement and can be of great impact in reducing job flow times, in any manufacturing processors setting, when alternative manufacturing processors exist batch splitting may become a specially attractive strategy for shortening flow times. Actually, we focus the quality of scheduling in shortening job throughput times and meeting job due dates. The reason is that this has a positive contribution to both profits and customer service. To further enhance this multi-objective we adopt a strategy to scheduling referred as horizontal scheduling and related with horizontal loading (Vollmann, 1996). Under horizontal scheduling, a job is scheduled first in all the required processors before another job is considered for scheduling. Because a job is associated with a product, involving both manufacturing of parts and their assembly, we decided to name this scheduling strategy as Product Oriented Scheduling (POS). Other authors, such as Hastings and Yeh (1990), in similar, but not identical, manufacturing settings, have also adopted this name. The scheduling approach combining all or some of these factors for reducing job throughput time has been named as Simultaneous Manufacturing (Silva, 1995).

1 Problem Definition

The problem under consideration can be included in the *job-shop* class with renewable resources. Because it assumes the existence of alternative processors to process the job we may say that is a *flexible job-shop* problem (Brucker and Schlie, 1990), (Mastrolilli and Gambardella, 2000). Besides that, it is assumed a variable number of operations per job, so it is not restricted to the *pure job-shop* problem but it has characteristics of the *general job-shop* (Conway *et al.*, 1967) problem. However, a job may require simultaneous processing in different machines. This happens because not only several parts may belong to the same job, but also because batch overlapping and batch splitting may be used. This scheduling problem scenario is more general than the basic JSSP, and is in line with what happens in practice. We call the scheduling problems of the type defined as Extended Job-Shop Scheduling Problems (EJSSP).

2 Mechanisms and Concepts

2.1 Product Oriented Scheduling

Under POS a job is a manufacturing order of products, involving parts manufacturing and their assembly. The scheduling process is focused on the job, in such a way that all the operations are scheduled in all the required resources or processors before the next job is considered for scheduling. Using this approach it is possible to have a good perception of the state of each job during the scheduling time horizon and to easily establish the scheduled job completion date, which permits verifying if job due date is likely to be accomplished.
Usually, POS is used by ERP software for loading and capacity planning based on lead times. We use POS in shop floor or detailed scheduling without constraining schedules to lead times. These are variable in time and depend on job and processors availability.

2.2 Batch Scheduling

In traditional batch production, a job is considered as a set of identical parts that are always processed as a whole, i.e. the full batch must be processed in a processing stage before it can be transferred to another to carry out further processing. In cases where the batch size is large, this can become a too great penalty to the full duration of the job processing. So the performance of the manufacturing system, mainly regarding job throughput time and accomplishment of job due dates can become highly poor. This operating weakness can be highly reduced through POS, batch overlapping and batch splitting. We explore these strategies in our approach to solve the EJSSP.

Batch Splitting. Batch splitting means partition of a batch into several smaller batches to be processed independently on machines. In this way, a substantial reduction on average batch flow time may be achieved. This may be particularly so when alternative machine or processors exist for simultaneous manufacture of split batches. An apparent disadvantage of this strategy is the overall increase of batch set-up times. This may not be important, neither be pernicious to job flow times, if processors, which are set-up, are not critical, i.e. are not bottlenecks.

Batch Overlapping. Batch overlapping means transferring work from a machine, which is processing an operation of the job, to another machine, for processing the next operation, before the entire batch has been finished on the previous machine. This is very common, in practice, sometimes done randomly, with different amounts of overlapping, and other times under a well defined overlapping procedure. In this case, *transfer batches* are clearly defined. These are batches, normally smaller than the total job batch size, transferred between two successively required machines. When a transfer batch is equal to the total job batch size batch overlapping does not take place. In the extreme, when trying to fully implement Simultaneous Manufacturing we should seek maximum overlapping, i.e. the transfer of work between processors should be continuous, which means transfer batches of size one.

Batch overlapping does not necessarily changes the processing batch size at a processing stage. A *processing batch size* is the amount of units of a job processed in a machine continuously before it takes another job.

Batch overlapping requires combination with batch streaming. This means that batch processing and batch transfer should be linked in a manner that each transfer batch can be transferred to its successor operation, immediately upon completion. Therefore, several transfer batches can be concurrently processed at different job operations. This contributes for short job throughput times and, due to this, for improved product delivery. However, splitting a batch into multiple transfer batches for overlapping, couples machines at different stages of processing. This coupling calls for a more close control of production, eventually requiring an increased effort in rescheduling under disturbances. Additionally, if products arrive at machines in identical transfer batches, intermittent idling may exist at machines whenever a predecessor operation requires a longer processing time. In our research, we developed a mechanism for avoiding the existence of these inactive or idle times based on *flexible batch transfer sizing* which is the same to say that different transfer batch sizes must be determined between succeeding adjacent job operations.

2.3 Simultaneous Manufacturing

The Simultaneous Manufacturing (SM) philosophy aims at the complete manufacturing of each single product of a product order in the minimum possible time. The intention is to take the minimum time to manufacture the whole job, i.e. the product order. To achieve this, each set of parts belonging to each product of the order must flow in a coordinated way through the system, i.e. in a way such that they are processed before any other parts, and arrive simultaneously to where they are needed for assembly. This assembly, as long resources are available, should be performed immediately after parts arrival. In this way, the throughput time for the complete manufacture and assembly of each product of the product order is short and, therefore, the full job throughput time is short too. Additionally, the work-in-process is likely to be low. As we can envisage SM calls for intensive batch overlapping and for batch splitting under

C_l with l=1, 2, …, c represents each graph path

$prec(op_1) = \{\} \wedge succ(op_n) = \{\}$

tp_{opi} it is the processing time of operation op_i

tp_{C_l} it is the jobs total processing time for the graph path under consideration

The following set is obtained: $Tp_{C_l} = \{tp_{C_l}\}$

where: C_l with l=1, 2, …, c represents each graph path

***Step 2* – Computation of the starting and finishing instants for the first operation in critical paths**

$$t_{start(a)} = r_k ; \quad (2)$$

where: r_k its the job release date

$$t_{fin(a)} = t_{start(a)} + n * tp_a \quad (3)$$

***Step 3* – Computation of the displacement or overlapping time between two succeeding adjacent operations *a* and *b* ($\Delta ts_{(a \succ\succ b)}$) from critical paths**

The batch *displacement* between two succeeding adjacent operations a and b, of the same job, is the lowest time interval between the starting of the two operations that ensures enough feeding to the machine which processes operation b for avoiding processing interruption, i.e. intermittent idling. This displacement allows defining the *transfer batch* $lt_{(a \succ\succ b)}$.

Considering a job batch size of n units, the size of the *transfer batch* depends on factors such as the minimum and maximum amount work that can be transferred between the manufacturing processors and the *buffer* size of the receiving manufacturing processor or machine. The transfer batch can, therefore, be unitary or have a size not exceeding the size of the job batch size. In order to simplify the computation procedure we consider that $n/lt_{(a \succ\succ b)} \in \square$.

Two cases can occur depending on operation processing times, so the computation mechanisms are different.

Case a) If $tp_a \leq tp_b$

then $tp_b * n \geq lt_{(a \succ\succ b)} * tp_b + (n - lt_{(a \succ\succ b)}) * tp_a$

This means that the total processing time for all the units of the batch for the operation b is long enough to ensure that no interruption occurs during batch processing at operation b. So the processing of operation b can start as soon as the transfer batch arrives at the respective manufacturing processor, after the first transfer batch has been processed in operation a.

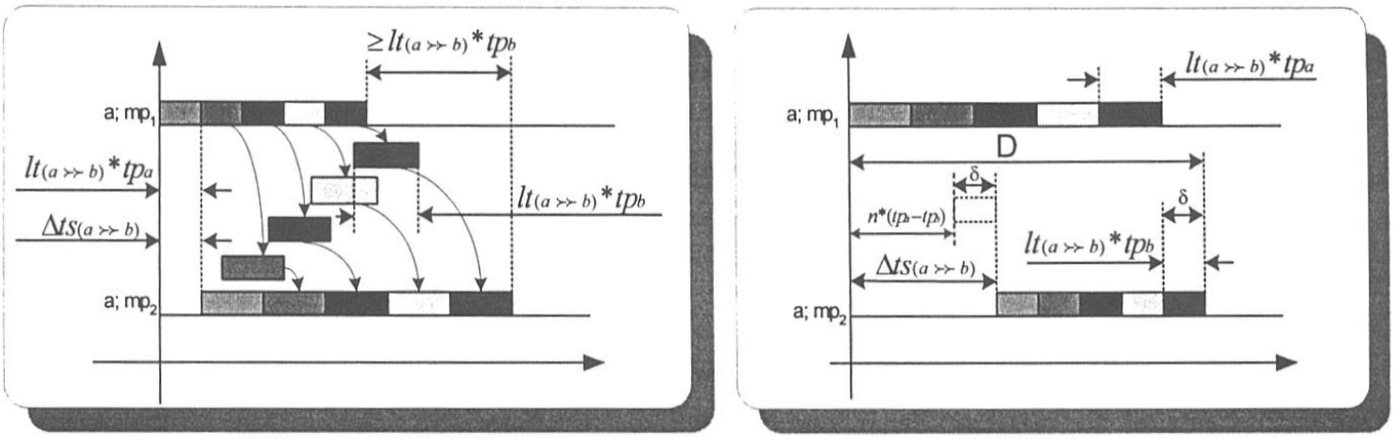
Figure 2 a) illustrates the relationships between important variables in such a way that one can easily understand how the displacement for this case, is obtained, namely through the following expression:

$$\Delta ts_{(a \succ\succ b)} = lt_{(a \succ\succ b)} * tp_a \quad (4)$$

Case b) If $tp_a > tp_b$

then $tp_b * (n - lt_{(a \succ\succ b)}) < (n - lt_{(a \succ\succ b)}) * tp_a$

so: $tp_b * n < lt_{(a \succ\succ b)} * tp_b + (n - lt_{(a \succ\succ b)}) * tp_a$



a) b)

Figure 2 Displacement representation for *Step* 3 a) when $tp_a \leq tp_b$ b) when $tp_a > tp_b$

This means that the total processing time for all the units of the batch for the operation *b* it is not long enough to avoid interruption during batch processing. So if the processing of operation *b* starts as soon as the transfer batch arrives at the respective manufacturing processor, after operation *a* has been processed, then it would occur a batch processing interruption due to starving, i.e. intermittent idle time would take place. Therefore, the displacement must be long enough to ensure processing of operation *b* without interruption. Figure 2 b), illustrates how such displacement is obtained in this case.

Thus, from figure 2 b) we can see that $\delta = lt_{(a \succ\succ b)} * tp_b$

In fact, being: $D = n*tp_a + lt_{(a \succ\succ b)} * tp_b$

and $D = n*(tp_a - tp_b) + \delta + n* tp_b$

it comes $\delta = lt_{(a \succ\succ b)} * tp_b$

Therefore, the displacement for this case, as illustrated in figure 3, is:

$$\Delta ts_{(a \succ\succ b)} = n*(tp_a - tp_b) + lt_{(a \succ\succ b)} * tp_b \quad (5)$$

***Step 4* – Computation of the starting and finishing instants for the other operations of critical paths**

$$t_{start(b)} = t_{start(b)} + \Delta ts_{(a \succ\succ b)} \quad (6)$$

where *a* is the operation with the greater starting instant (t_{start}), in the universe of all the preceding operations of *b*.

$$t_{fin(b)} = t_{start(b)} + n * tp_b \quad (7)$$

***Step 5* – Computation of the starting and finishing instants for the all the operations of the other paths**

a) If $tp_a \leq tp_b$

$$t_{start(a)} = t_{start(b)} - lt_{(a \succ\succ b)} * tp_a \quad (8)$$

$$t_{fin(a)} = t_{start(a)} + n * tp_a \quad (9)$$

b) If $tp_a > tp_b$

$$t_{fin(a)} = t_{fin(b)} - lt_{(a \succ\succ b)} * tp_b \quad (10)$$

$$t_{start(a)} = t_{fin(a)} - n * tp_a \quad (11)$$

When operation *a* also belong to the graph path with the highest duration its starting and finishing time instants are already know, so in this case it is not necessary to compute those values again.

Step 6 – **Computation of the displacement $\Delta ts_{(a \succ\succ b)}$ between the starting time instants of two succeeding adjacent operations *a* e *b*, belonging to a non-critical path.**

Case a) If $tp_a \leq tp_b$

$$\Delta ts_{(a \succ\succ b)} = t_{start(a)} - t_{start(b)} \quad (12)$$

Case b) If $tp_a > tp_b$

$$\Delta ts_{(a \succ\succ b)} = t_{start(b)} - t_{start(a)} \quad (13)$$

We adopt the following representation for a JSP:
$JSP_k = \{(mp_1, (op_1, t_{start}, t_{fin}), ..., (op_n, t_{start}, t_{fin})), ..., (mp_n, (op_1, t_{start}, t_{fin}), ..., (op_n, t_{start}, t_{fin}))\}$

3.3 Illustrative example

Supposing that we intend to manufacture a batch with 8 product units within a deadline, *td*, of 70 *tu*, being the release date, *rk*, the instant 0, we have: $n = 8$, $td = 70$ and $rk = 0$. The precedence graph is represented in figure 1. In table 1 are represented the processing times, the respective manufacturing processor for each operation, the preceding and succeeding operations as well as the occupation times for each manufacturing processor.

Table 1 Processing times and operation precedence

Operation - op_{ki}	Manufact. Processor.- mp_j	Proc. Time – $tp\ op_{kij}$	Occupation time (n*$tp\ op_{kij}$)	$Prec(op_{ki})$	$Succ(op_{ki})$
op_1	mp_1	1	8	-	op_2
op_2	mp_2	3	24	op_1	op_4
op_3	mp_3	1	8	-	op_4
op_4	mp_4	2	16	op_2 , op_3	-

Considering the data of the example and a unitary transfer batch, the result obtained by the job scheduling pattern generator mechanism, is illustrated in figure 3 where the batch displacement between op_1 and op_2, and between op_2 and op_4 is shown.

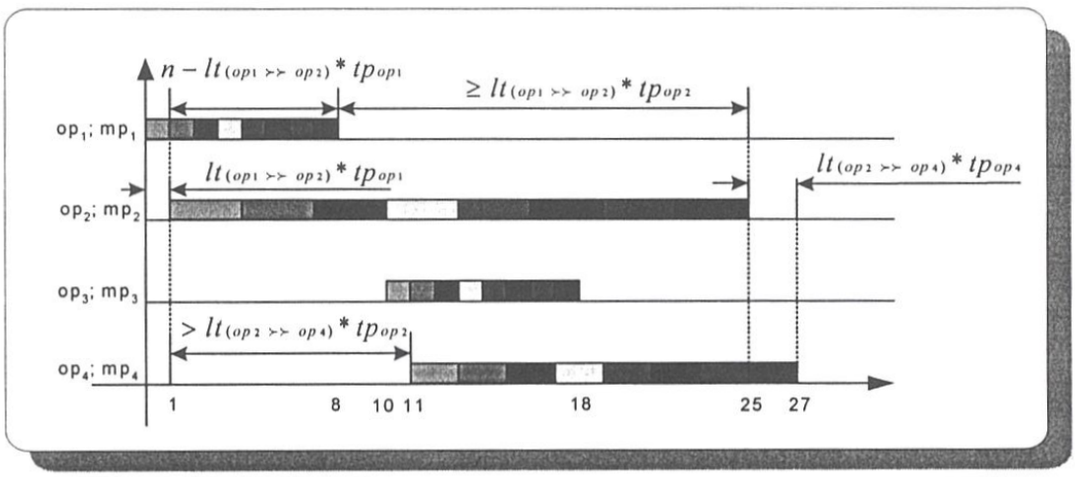


Figure 3 Job Scheduling Pattern with batch displacement representation

$JSP_k = \{(mp_1,(op_1, 0, 8)), (mp_2,(op_2, 1, 25)), (mp_3,(op_3, 10, 18)), (mp_4,(op_4, 11, 27))\}$

We point out that operation op_3 could start anytime between instants 0 and 10. By starting at instant 10 a late start JSP is used. If had started at instant 0 we were adopting the earliest start JSP.

If, instead of a unitary transfer batch, we consider, in the other extreme, a transfer batch with the same size of the job batch the resulting job scheduling pattern is different and shown figure 4.

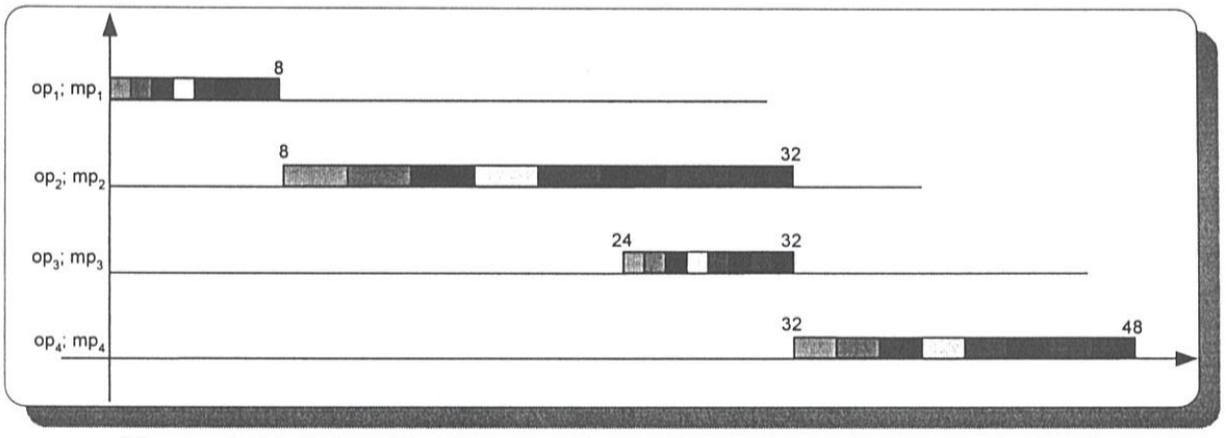


Figure 4 Job Scheduling Pattern with transfer batch equal to processing batch

$JSP_k = \{(mp_1,(op_1, 0, 8)), (mp_2,(op_2, 8, 25)), (mp_3,(op_3, 24, 32)), (mp_4,(op_4, 32, 48))\}$

It can be noticed that the job flow time has increased from 27 *tu* to 48 *tu*. The starting and finishing instants of the occupation periods of the processors became different although the amount of time involved is the same.

4 Product Oriented Scheduling through JSP

With basis on the JSP and processor *agendas* the Scheduling Plan Generator (SPG) (Almeida, 2002), which implements the POS approach, generates a scheduling solution. The SGP draws upon a scheduling algorithm developed in previous work by the authors (Almeida *et al.*, 1999). Such algorithm involves two scheduling phases: *the forward influence phase* and the *backward influence phase*. Mechanisms for implementing these phases were reported and explained in Almeida *et al.* (2002).

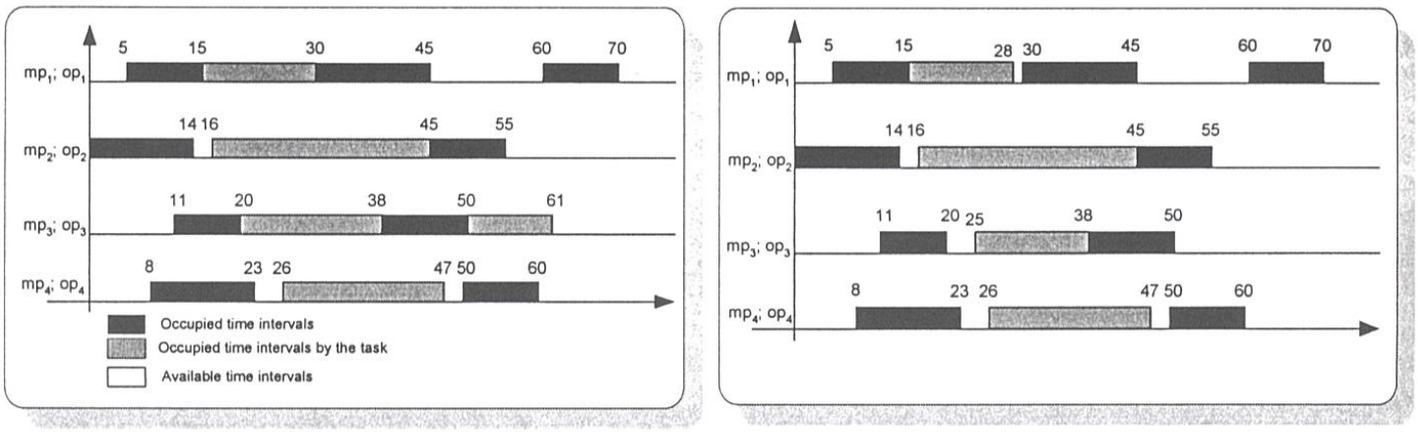
In the scheduling mechanism, the concept of a manufacturing processor (MP) *agenda* is important. It represents the MP available time intervals for scheduling jobs. Figure 5 shows the *agendas* of four manufacturing processors required for processing the job referred above, represented by the black time intervals.

The SPG picks up information about the work and processors, related with the scheduling problem and suggests possible solutions if they exist. It verifies the time horizon of the processors associated to each operation and looks for their availability in a relevant time interval.

One of the strong points of the SPG is its capability of, within the available time intervals of the manufacturing processors, pin pointing those intervals where manufacturing jobs can be successful and efficiently scheduled. Another important feature the SPG, when it is not possible to schedule the whole job batch to meet its due date, is its ability to suggest a split batch size that can be manufactured within the due date. This may be important for time phasing delivery and negotiation with customers.

Going back to the example, considering a unitary transfer batch, which means using the JSP shown in figure 3, the obtained results for each phase of the algorithm, implemented by the SPG, are represented in figure 5: a) for the forward influence phase and b) for the backward influence phase, being b) the final result for the scheduling of the referred job.

It can be observed that, on the backward influence phase, the time interval [50,61] on manufacturing processor mp_3 has been eliminated, the time interval [15,30] on mp_1 has been limited to [15,28], the time interval [20,38] on mp_2 has been limited to [25,38] and the time interval [25,47] on mp_4 has been limited to [26,47].



a) b)

Figure 5 Results of application of the Scheduling Plans Generator

If more than a single set of intervals exist for scheduling, further thinking is necessary to choose among alternative schedules. So, some additional criteria and heuristics may be put on top of the scheduling mechanisms used to arrive to improved solutions.

When it is not possible to schedule a job within the processors agendas, based on a given JSP, then some strategies can be applied to obtain a solution, within the available time intervals in the manufacturing processors agendas. These strategies can include trying new JSP, considering different transfer batches sizes, and applying a batch splitting process followed by JSP generation for the split batches. The scheduling mechanisms in the SPG give suggestions about where to split a job batch. Thus, we can break one job batch in two or more smaller job batches whose sizes can fit within the available time intervals of the manufacturing processors.

6 Conclusions

The main concern in detailed scheduling in practice is to obtain good solutions to schedule jobs at hand. Frequently, due to the lack of aids for this, scheduling relies solely on the ability of schedulers on the shop floor. However, under complex manufacturing environments, such as those of industries of type A or T (Chase and Aquilano, 1998) most probably the obtained solutions are less than good. It simply it is not possible for a person, alone and without tool aids for scheduling, to envisage the implications of decisions taken on the whole performance of the system, and, as a consequence poor solution are likely to be obtained. It seems, therefore, to be important that tools are provided, which can improve the task of schedulers in detailed scheduling.

Moreover, under today's highly competitive markets it is important to provide good service to customers and, at the same time, reduce costs in manufacturing. One clear contribution on these lines is to manufacture job orders within the shortest time that is reasonably possible, keeping work in process low. This has a direct effect on profitability, since fast turnover of short-term investment is achieved, and, of course, on fast deliveries, which is an important requisite for customer satisfaction.

We presented in this paper an approach oriented towards achieving very short manufacturing throughput times of customer orders. The approach uses the Job Scheduling Pattern concept, described in detail, and implements a philosophy to scheduling known as Simultaneous Manufacturing. The main imbedded strategy used is what we described as Product Oriented Scheduling. Based on the described scheduling approach a Scheduling Plan Generator was developed, which can be seen as a powerful tool to aid users to improve detailed shop floor scheduling in complex manufacturing environments, integrating both fabrication of parts and their assembly into customer ordered products.

References

Almeida, Ana e Ramos, Carlos and Silva, Silvio Carmo (1999). "Dynamic Scheduling of Manufacturing Orders: A Decision Support System Approach", *IEPM'99 International Conference on Industrial Engineering and Production Management*, Glasgow, Scotland.

Almeida, Ana, Ramos, Carlos and Silva, Silvio Carmo (2002). "Toward Dynamic Scheduling of Manufacturing ", *The International Journal for Manufacturing Science & Production*, Freund Publishing House, Ltd..

Almeida, Ana M.N. (2002). "Analysis and Development of Mechanisms and Algorithms to Support Product Oriented Scheduling ", *Doctoral Thesis on Production Engineering*, submitted to the University of Minho, Portugal.

Baker, K. R. (1974). "*Introduction to Sequencing and Scheduling*", John Willey & Sons, Inc.

Blazewicks, J., Ecker, K. H., Pesch, E., Schmidt, G., and Weglarz, J. (2001). "*Scheduling in Computer and Manufacturing Systems*", Springer, Heidelberg, New York (2nd Edition).

Brucker, P., Schlie R. (1990). "Job-shop scheduling with multi-purpose machines", *Computing 45*, 369-375.

Chase, R. B., Aquilano, N.J. and Jacobs, F.R. (1998) "*Production and Operations Management: A Life Cycle Approach*", 8th ed., Irwin, Homewood, Boston.

Conway, R. W., Maxwell, W. L., and Miller, L. W. (1967). "*Theory of Scheduling*", Addison-Wesley Publishing Company.

French, S. (1982). "*Sequencing and Scheduling*", New York: Halsted.

Hastings, N.A.J. and Yeh, C.H. (1990). "Job oriented production scheduling", *European Journal of Operacional Research*, nº47, pp 35-48.

Mastrolilli, M. and Gambardella, L. (2000). "Effective Neighborhood Functions for he Flexible Job Shop Problem", *Journal of Scheduling*, Vol. 3, 3-20.

Portmann, M. C. (1997). "Scheduling methodology: optimisation and compu-search approaches I", *The Planning and Scheduling of Production Systems, Methodologies and applications*, edited by A. Artiba and S. E. Elmaghraby, Chapman & Hall..

Silva, S.C. (1995). "Project on Intelligent Manufacturing cell at University of Minho", *1st World Congress on Intelligent Manufacturing Processes & Systems*, Mayguez, Puerto Rico.

Vollmann, T. E., William, L. B. and Whybpark, D. C. (1996). "*Manufacturing planning and control systems*", Richard D. Irwin, Inc..