

Capacity planning and lead time management

W.H.M. Zijm*, R. Buitenhek

Department of Mechanical Engineering, University of Twente, P.O. Box 217, 7500 AE Enschede, Netherlands

Abstract

In this paper we discuss a framework for capacity planning and lead time management in manufacturing companies, with an emphasis on the machine shop. First we show how queueing models can be used to find approximations of the mean and the variance of manufacturing shop lead times. These quantities often serve as a basis to set a *fixed* planned lead time in an MRP-controlled environment. A major drawback of a fixed planned lead time is the ignorance of the correlation between actual work loads and the lead times that can be realized under a limited capacity flexibility. To overcome this problem, we develop a method that determines the earliest possible completion time of any arriving job, without sacrificing the delivery performance of any other job in the shop. This earliest completion time is then taken to be the delivery date and thereby determines a *workload-dependent* planned lead time. We compare this capacity planning procedure with a fixed planned lead time approach (as in MRP), with a procedure in which lead times are *estimated* based on the amount of work in the shop, and with a workload-oriented release procedure. Numerical experiments so far show an excellent performance of the capacity planning procedure.

Keywords: Queueing network; Job shop; Scheduling; Due dates

1. Introduction

Many manufacturing companies have adopted Materials Requirements Planning (MRP I) as a means for production control and materials coordination. The basic planning procedure starts from a Master Production Schedule (MPS) which contains the planned production quantities of end-products (MPS items) for a certain planning horizon. It then uses the manufacturing Bill of Materials (product structure) to calculate the time-phased needs of subassemblies, parts and raw materials. Fixed off-set lead times are used to perform

the time-phasing, i.e. to determine when exactly to start each phase. At each phase production quantities are principally determined by the demand for MPS items multiplied by an explosion factor. These quantities may be adjusted due to lot sizing and inventory considerations (netting), see e.g. Ref. [1].

The basic question however is: What is a satisfactory and realistic MPS? To answer this question one has to consider the planning problem from two angles: the demand point of view and the capacity point of view. Clearly, in a make-to-order company the MPS should reflect customer orders and in a make-to-stock company it should reflect anticipated demand. On the other hand, it should *match the available capacity* in the subsequent

* Corresponding author.

departments. The available capacity obviously restricts the production quantities, although capacity may be somewhat flexible due to overtime work and possibilities for subcontracting.

Unfortunately, MRP I does not consider capacity at all, while the initial promises of the more extensive Manufacturing Resource Planning (MRP II) system (cf. Ref. [1]) are not fulfilled. The Rough Cut Capacity Planning (RCCP) module of MRP II concerns only the long-term capacity availability on a high aggregation level while Capacity Requirements Planning (CRP) performs just a check on the amount of capacity needed. In case of a mismatch between available and required capacity it is left to the planner to adjust the MPS. The temporary alteration of lot sizes offers another possibility, but the difficulties in doing this in the formal system forces a planner to rely on informal procedures, thereby undermining the data accuracy of the MRP system.

The basic drawback of the MRP philosophy is the decoupling of the time-phasing procedure (using prefixed off-set lead times) and the time-dependent work load in each department. Work-load independent lead times are best justified in a materials procurement phase and in an assembly phase. In an assembly phase, the simplicity of product routings usually allows for a simple input–output control procedure while the amount of available capacity is often easily adjusted. In a parts manufacturing shop, however, where products follow very diverse routings and capacity primarily refers to machine tool capacity, *actual manufacturing lead times are highly dependent on actual work loads, lot sizes, shop order release procedures, etc.* (see e.g. Ref. [2]). Typically, parts represented at one level of a Bill of Materials in an MRP system have to undergo a sequence of operations on different machines in a fabrication department. A major part of the total lead time is caused by delays due to the interference of different products and batching of similar products to avoid too many setups. The average delays of the jobs currently present in the shop depend on the current work load. The earlier completion time of some job can be pursued by using sophisticated shopfloor scheduling procedures, but unfortunately almost always at the cost of some other jobs.

In other words: actual job lead times in a manufacturing system can be influenced in many ways, but the impact of the actual work load is clearly dominant. This explains why work-load control rules [3] or work-load-oriented release rules [4] have become popular in some companies. Work-load control aims at reducing both the average *shoptime* and its variability by releasing orders only when the work load on the relevant machines does not exceed a certain limit. However, the *overall lead times* of jobs may still vary significantly dependent on the shop work load. The problem is only shifted because a job is now waiting *in front of the shop*, instead of *within the shop*, if some bottleneck cannot handle it in time (a similar procedure is advocated in OPT, see Ref. [5]). Although this definitely enhances transparency on the shopfloor it does not really solve any problem (on the contrary, it increases overall lead times).

Fixed lead times can only be maintained by influencing either the required or the available capacity. Order acceptance procedures based on actual work load information (with the option to refuse orders when the load is too high or to negotiate longer lead times) offer a natural way to influence demand but are useless if no accurate work load information is available. Indeed, another possibility is demand smoothing by production to stock but this is only realistic for relatively standard (i.e. non-customized) items. Altering process plans to shift work from bottleneck to non-bottleneck machines offers a third, technical, possibility (see e.g. Ref. [6]) but this requires extremely sophisticated process planning systems not available to date. Adapting the available capacity may be possible by working in overtime or by using subcontractors temporarily. If none of the above options offers a sufficiently effective solution, one is left with developing a *planning system which uses work-load-dependent lead times as a starting point*. Such a planning system should result in a better delivery date performance than planning systems which ignore the influence of the work load or planning systems which hide any variability in the lead times by defining planned lead times which are much longer than necessary (as MRP does, see Ref. [2]).

This paper aims at developing a manufacturing planning and control framework taking into

account the above observed phenomena. The system should at any time predict realistic lead times and thereby realistic delivery dates but at the same time allow these lead times to depend on the current work load. This clearly contrasts with actual MRP practice and requires a thorough integration of capacity planning and lead time management. In Section 2 we apply queueing network techniques to determine the mean and the variance of the lead times as a function of lot sizes, product mix and estimated annual production volumes. In addition, it is shown how to calculate the average work load for each machine (group) in the shop. These quantities may serve as a basis for determining fixed planned lead times or a work-load-oriented release procedure, respectively. However, when the work load is high, neither fixed lead times nor work load control procedures guarantee the timely completion of all orders. Therefore, a capacity planning procedure, based on aggregate scheduling techniques (cf. Section 3) will be developed. Basically, this procedure determines the earliest possible completion time of an order, under the condition that all jobs present in the shop are still completed in time. This earliest completion time is then defined as the job's delivery date and determines the ultimate *planned lead time* (cf. Section 4). In Section 5, we compare the capacity planning procedure with a fixed lead time approach, a work load control rule and a procedure which estimates completion times based on actual work loads on each machine. In Section 6 we conclude the paper and discuss future research.

2. Determining the lead time and work load characteristics with queueing models

In this section we briefly discuss how job shops can be modeled by queueing networks. By using queueing network theory we find approximations for the mean and variance of the lead times and for the mean of the work load in the shop. Required input parameters are the shop configuration and job characteristics such as routings, processing times, setup times and lot sizes. Part of the analysis is based on the framework developed by Karmarkar [2] and Karmarkar et al. [7]. These

authors basically treat each service station in the network as an $M|G|1$ queue with multiple part types. The application of the Pollaczek–Khintchine formulas and their extension to multiple part types then immediately yields the desired results per workstation. The overall lead times follow directly by combining the results per workstation.

It should be emphasized that modeling a job shop as a network of $M|G|1$ workstations is not exact. Arrival streams at a workstation are superpositions of departure streams from the other stations and an external arrival stream. Since the departure streams at the $M|G|1$ stations are not Poisson, the arrival streams cannot be Poisson either. Still we believe that the framework serves as a useful modeling tool to estimate the desired quantities.

Consider a job shop with multiple part types h ($h = 1, \dots, H$) and a part-type-dependent *deterministic* routing for each part. Let $D^{(h)}$ denote the demand rate of part type h . Let each job represent a batch of one part type, with part-type-dependent deterministic lot size $Q^{(h)}$. Assume that each batch is completely processed at one station before any part of it is transferred to the next station. Define the indicator function $\delta_{jk}^{(h)}$, which fixes the part-type-dependent routings, as follows:

$$\delta_{jk}^{(h)} = \begin{cases} 1 & \text{if the } k\text{th operation of part } h \text{ is} \\ & \text{performed on station } j, \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

Assume that no parts are scrapped. For the arrival rate $\lambda_{jk}^{(h)}$ of batches of part h at station j for their k th operation we then find

$$\lambda_{jk}^{(h)} = \frac{D^{(h)}}{Q^{(h)}} \delta_{jk}^{(h)}. \quad (2)$$

Let $a_{jk}^{(h)}$ be the deterministic processing time per part required for the k th operation of part h at station j and let $\tau_{jk}^{(h)}$ denote the corresponding deterministic setup time. For the *batch processing time* $p_{jk}^{(h)}$, corresponding to the k th operation of an order of parts of type h at station j , we then have

$$p_{jk}^{(h)} = \tau_{jk}^{(h)} + Q^{(h)} a_{jk}^{(h)}. \quad (3)$$

The n th moment ($n \in \mathbb{N}$) of the part type and operation-independent batch processing time $\mathbb{E}(p_j)^n$

at station j can be found by conditioning on the part type and the operation

$$\begin{aligned} \mathbb{E}(p_j)^n &= \sum_{hk} \Pr(k\text{th operation of part type } h) \cdot (p_{jk}^{(h)})^n \\ &= \sum_{hk} \frac{\lambda_{jk}^{(h)}}{\sum_{hk} \lambda_{jk}^{(h)}} (p_{jk}^{(h)})^n \\ &= \sum_{hk} \frac{\lambda_{jk}^{(h)}}{\lambda_j} (p_{jk}^{(h)})^n, \end{aligned} \tag{4}$$

where λ_j denotes the total part type and operation-independent arrival rate at station j . The average work load ρ_j at station j now satisfies

$$\begin{aligned} \rho_j &= \lambda_j \mathbb{E}p_j \\ &= \lambda_j \sum_{hk} \frac{\lambda_{jk}^{(h)}}{\lambda_j} p_{jk}^{(h)} \\ &= \sum_{hk} \lambda_{jk}^{(h)} p_{jk}^{(h)}. \end{aligned} \tag{5}$$

Assuming Poisson arrival processes at each station, the Pollaczek–Khintchine formula (cf. Ref. [8, p. 190]) yields the following expression for the mean waiting time in queue $\mathbb{E}W_j$ at station j :

$$\begin{aligned} \mathbb{E}W_j &= \frac{\lambda_j \mathbb{E}(p_j)^2}{2(1 - \rho_j)} \\ &= \frac{\lambda_j \sum_{hk} (\lambda_{jk}^{(h)} / \lambda_j) (p_{jk}^{(h)})^2}{2 \left(1 - \sum_{hk} \lambda_{jk}^{(h)} p_{jk}^{(h)} \right)} \\ &= \frac{\sum_{hk} \lambda_{jk}^{(h)} (p_{jk}^{(h)})^2}{2 \left(1 - \sum_{hk} \lambda_{jk}^{(h)} p_{jk}^{(h)} \right)}. \end{aligned} \tag{6}$$

The mean lead time $\mathbb{E}T_{jk}^{(h)}$ corresponding to the k th operation of part type h at station j is now given by

$$\mathbb{E}T_{jk}^{(h)} = \mathbb{E}W_j + p_{jk}^{(h)}, \tag{7}$$

leading to the following result for the mean shop manufacturing lead time $\mathbb{E}T^{(h)}$ of an order of part type h :

$$\mathbb{E}T^{(h)} = \sum_{jk} \delta_{jk}^{(h)} \mathbb{E}T_{jk}^{(h)}. \tag{8}$$

These results can be refined considerably by using more sophisticated approximations for queueing networks. For instance, it is possible to model each node as a $GI|G|1$ station or in case of parallel machines, as a $GI|G|c$ station and next use two-moment approximations to obtain more accurate formulas for the mean waiting time in queue (see e.g. Refs. [9, 10] and [11, Ch. 7]). Bitran and Tirupati [12] show that the restriction to jobs with deterministic routings may lead to more accurate approximations than those derived by Whitt. All these results have been implemented in the software package MPX by Suri et al. (see e.g. Ref. [13]).

To calculate the variance of the shop lead time per part type, we first determine the second moment of the waiting time of a batch of part type h . Applying the second moment version of the Pollaczek–Khintchine formula (cf. Ref. [8, p. 201]) yields

$$\mathbb{E}(W_j^2) = 2(\mathbb{E}W_j)^2 + \frac{\sum_{hk} \lambda_{jk}^{(h)} (p_{jk}^{(h)})^3}{3(1 - \rho_j)}, \tag{9}$$

from which we immediately deduce the variance of the waiting time $\text{Var}(W_j)$ by

$$\text{Var}(W_j) = \mathbb{E}(W_j^2) - (\mathbb{E}W_j)^2. \tag{10}$$

Recall that the batch processing time is fixed and assume that the lead times at different stations are mutually independent. We then find the overall lead time variance for a batch of part type h from

$$\text{Var}(T^{(h)}) = \sum_{jk} \delta_{jk}^{(h)} \text{Var}(W_j). \tag{11}$$

The assumption that the lead times at the different stations are mutually independent is fulfilled in product-form networks and may yield reasonable approximations in more general networks. Other approximations can be found in Ref. [11].

Finally, let $L_{jk}^{(h)}$ denote the mean number of batches of part h in queue or in process at work-station j for their k th operation. From Little’s law it follows that

$$L_{jk}^{(h)} = \lambda_{jk}^{(h)} \mathbb{E}T_{jk}^{(h)}, \tag{12}$$

from which we may derive an approximation for the average work in the shop $\mathbb{E}V_j$ that has to be

processed at station j :

$$\mathbb{E}V_j = \sum_{hk} L_{jk}^{(h)} \left(p_{jk}^{(h)} + \sum_{n>k} p_{jn}^{(h)} \right) + \sum_{hk} \sum_{i \neq j} L_{ik}^{(h)} \left(\sum_{n>k} p_{jn}^{(h)} \right). \quad (13)$$

The first term of this expression refers to the jobs currently present at station j (in queue or in process) and includes both current and future operations. The second term refers to jobs currently present at other stations, which may still have to visit station j (possibly more than once). Also, the expression slightly overestimates the amount of work in the shop by assuming that a job currently in process at a machine will be present for its full (instead of its remaining) processing time. The latter inaccuracy can be corrected, but for our purposes the present expression is sufficient.

The expressions derived in this section are usually used for performance evaluation purposes. Here they will serve to determine fixed planned lead times and to estimate work-load-dependent planned lead times in Section 4. In the next two sections, a capacity planning procedure is developed which yields individual planned lead times for each batch, based on the actual work load. The underlying scheduling procedure will be outlined in Section 3. In Section 5 we numerically compare the capacity planning procedure with the global planning procedures based on the results in this section.

3. A decomposition-based scheduling approach

In this section we outline a job shop scheduling algorithm, which is based on an iterative decomposition of the job shop. This algorithm is known as the Shifting Bottleneck method (SBM) and was initially developed by Adams et al. [14], albeit in a slightly different version. Consider the job shop introduced in Section 2. Suppose that at time 0 we face the problem of scheduling N given jobs. Each job represents a batch of parts of one type only. If the job concerns part type h ($h = 1, \dots, H$) then the processing time of its k th operation, to be performed on workstation j , is given by Eq. (3).

In particular, each job will be processed at each workstation without any interruption (non-preem-

ptive scheduling) while its setup time is included in the processing time. Since in this section we deal with a *static scheduling problem*, we will not refer to the part type any longer. Instead, we simply number the jobs from 1 to N and we say that the processing time of the k th operation of job i ($i = 1, \dots, N$) on machine j equals p_{ijk} . Each job is further characterized by its release date r_i and its due date d_i .

A schedule S resulting from any scheduling algorithm specifies the starting time and completion time of all operations on all N jobs. Denote by $c_i(S)$ the completion time of job i under schedule S . We can then define the lateness $L_i(S)$ of job i under schedule S as follows:

$$L_i(S) = c_i(S) - d_i. \quad (14)$$

If $L_i(S) \leq 0$ then job i is completed in time. If $L_i(S) > 0$ then job i is late. Denote by $L^*(S)$ the maximum lateness under schedule S , defined by

$$L^*(S) = \max_{i=1, \dots, N} [L_i(S)]. \quad (15)$$

We want to find the schedule which yields the minimum maximum lateness. This is the classical job shop scheduling problem with the maximum lateness criterion, with the additional feature that a job may visit a workstation more than once (on the other hand, not each job necessarily visits each workstation). Although clearly NP-complete, the SBM heuristically solves the problem with very satisfactory results. Below, we outline the basic algorithm and illustrate it with a simple example.

The basis of the SBM is a decomposition procedure leading to a series of one machine scheduling problems. The decomposition procedure generates virtual release dates and due dates for the operations to be scheduled in the one machine scheduling problems. The objective of each one machine scheduling problem is again to minimize the maximum lateness. These problems are optimally solved by the algorithm of Carlier [15]. In addition, the so-called delayed precedence constraints may be generated by the decomposition procedure. Such a constraint specifies a precedence relation between two operations as well as the minimum time interval between the completion of the first

operation and the start of the second one. Delayed precedence constraints are needed, for instance, when two operations on one machine concern the same job. To solve optimally the one machine scheduling problem with delayed precedence constraints, an extension of the algorithm of Carlier, developed by Balas et al. [16], is exploited.

For the description of the SBM, we introduce some notation:

$$\delta_{ijk} = \begin{cases} 1 & \text{if the } k\text{th operation of job } i \text{ is} \\ & \text{performed on machine } j, \\ 0 & \text{otherwise,} \end{cases}$$

O_{ik} = the k th operation of job i

r_{ik} = the virtual release date of operation O_{ik}

d_{ik} = the virtual due date of operation O_{ik}

p_{ik} = the processing time of operation O_{ik}

Ω = the set of machines that have been scheduled

M = the number of machines

N = the number of jobs

K_i = the number of operations of job i .

The job shop scheduling algorithm is easily explained by using a so-called *disjunctive graph model*. For any job shop scheduling problem as specified above, a corresponding disjunctive graph is constructed as follows (see Fig. 1 for an example with three machines and three jobs). For every operation O_{ik} , there is a node with weight p_{ik} . For every two consecutive operations of the same job, there is a *directed* arc. For every two operations that require the same machine, there is an undirected edge; we call these edges the *machine edges*. Thus, the arcs represent the job precedence constraints, and the edges represent the machine capacity constraints. There are also two extra unweighted nodes, the source and the sink, which do not represent an operation. From the source to the first operation of each job and from the last operation of each job to the sink there are directed arcs. The former are weighted by the release dates r_i of the corresponding jobs. The latter are weighted by $D - d_i$, where d_i is the due date of the corresponding job and $D = \max_{i=1}^N d_i$. These nodes and weighted edges are useful for performing the longest path calculations, which will be introduced further on in this section.

Finding a schedule now corresponds to orienting the initially undirected edges. The schedule is feasible if there are no cycles in the resulting graph. Finding the optimal schedule corresponds to finding the orientation of the initially undirected edges, such that the maximum lateness of any job is minimized.

During each iteration longest path calculations are performed to determine the virtual release date and due date of each operation. The virtual release date of an operation is equal to the length of the longest weighted path from the source to the operation. The virtual due date of an operation is equal to D minus the longest weighted path from the operation to the sink.

Next, we present the SBM.

1. *Initialization* $\Omega = \emptyset$
2. *Compute the virtual release dates and due dates of the operations.* If this is the first time to compute the virtual release and due dates, then the required longest path calculations simplify to

$$r_{ik} = r_i + \sum_{l=1}^{k-1} p_{il} \quad (16)$$

and

$$d_{ik} = d_i - \sum_{l=k+1}^{K_i} p_{il}. \quad (17)$$

If this is not the first time to compute the release and due dates, perform the longest path calculations on the modified graph.

3. *Identify a bottleneck machine.* For all machines $j = 1, \dots, M$ schedule all operations O_{ik} for which $\delta_{ijk} = 1$ with the algorithm of Balas et al. [16]. These instances are specified by the virtual release dates r_{ik} and the virtual due dates d_{ik} , together with the processing times p_{ik} . This results in M schedules; one for each machine. Each schedule yields a minimum maximum lateness. Select the machine with the largest minimum maximum lateness as the bottleneck machine (where ties are broken arbitrarily). Retain the *sequence* of operations in the corresponding schedule (not the schedule).
4. *Add precedence constraints.* Let machine j be the latest scheduled machine. Add precedence constraints for all operations on machine j according

to the retained sequence. That is, fix the order of operations as found in the schedule. In the disjunctive graph the edges linking operations on the bottleneck machine j are oriented according to the sequence found in step 3.

5. *Recompute the virtual release dates and due dates*
This can be done by performing the longest weighted path calculations for the modified graph.
6. *Reschedule the bottleneck machines if $|\Omega| > 1$.*
Order the machines that have already been scheduled in decreasing order of maximum lateness. For each of the scheduled machines do the following:
 - (a) release the operations on the machine and disorient the corresponding edges.

- (b) recompute the release and due dates of the just released operations and solve the one-machine scheduling problem on that machine.
- (c) retain the new sequence and orient the edges accordingly.

If at least one of the machines has been changed, repeat this step. If $|\Omega| < M$, repeat this step at most three times. If $|\Omega| = M$, repeat this step until there are no changes in the schedule any more, and then stop the procedure.

7. *Stop criterion.* If $|\Omega| < M$, go to step 2. Otherwise the procedure is completed.

The next example should clarify the SBM.

Example 3.1. Consider a job shop with three machines, M_1 , M_2 and M_3 . Let there be three jobs J_1 , J_2 and J_3 with each three operations. The jobs have release date $r_i = 0$ ($i = 1, \dots, 3$) and due date $d_i = 18$ ($i = 1, \dots, 3$). For this example denote by μ_{ik} the machine on which operation O_{ik} has to be performed. The routings and processing times for the jobs are given in Table 1. Fig. 1 shows the initial disjunctive graph for this example.

We perform the SBM:

1. *Initialization* $\Omega = \emptyset$

Table 1
Routing and processing times for the example instance

J_i	K_i	μ_{i1}	μ_{i2}	μ_{i3}	p_{i1}	p_{i2}	p_{i3}
J_1	3	M_1	M_3	M_2	4	7	6
J_2	3	M_2	M_1	M_3	3	5	8
J_3	3	M_3	M_2	M_1	2	6	7

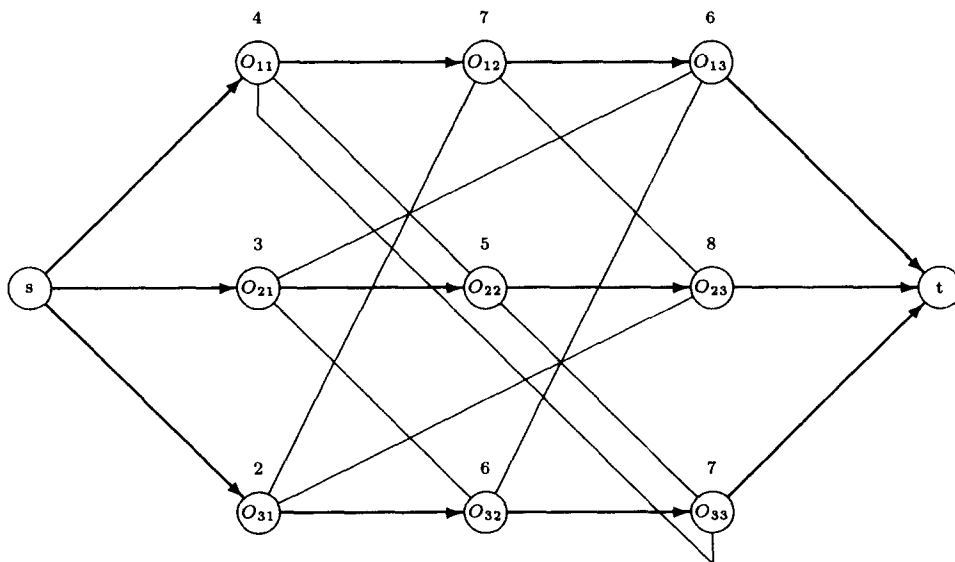


Fig. 1. Initial disjunctive graph.

First iteration:

1. We find the following virtual release dates:

	Release dates			Due dates		
	r_{i1}	r_{i2}	r_{i3}	d_{i1}	d_{i2}	d_{i3}
J_1	0	4	11	5	12	18
J_2	0	3	8	5	10	18
J_3	0	2	8	5	11	18

2. Note that in this example no job visits a machine more than once. We can therefore apply Carlier's algorithm [15] instead of its extension by Balas et al. [16]. Applying Carlier's algorithm to the three resulting single machine problems yields a largest maximum lateness of 1 for machine 3 (combine Table 1 and the release and due dates above). Hence, $\Omega = \{M_3\}$. Retain the optimal sequence on M_3 : (O_{31}, O_{12}, O_{23}).
3. Precedence constraints are now added to ensure that the optimal sequence on M_3 is maintained. In the disjunctive graph the corresponding ma-

chine edges are directed. The resulting graph is depicted in Fig. 2.

4. By performing the longest path calculations we find the following virtual release and due dates:

	Release dates			Due dates		
	r_{i1}	r_{i2}	r_{i3}	d_{i1}	d_{i2}	d_{i3}
J_1	0	4	11	3	10	18
J_2	0	3	11	5	10	18
J_3	0	2	8	3	11	18

5. $|\Omega| = 1$: do not perform step 6.
 6. $|\Omega| < M = 3$: go to step 1.

Second iteration:

2. Identify whether M_1 or M_2 is the bottleneck machine. We find that M_1 is the bottleneck machine with $L_{max} = 1$ and with (O_{11}, O_{22}, O_{33}) as the optimal sequence. Update $\Omega = \{M_1, M_3\}$. Orient the edges accordingly. We get the disjunctive graph depicted in Fig. 3.

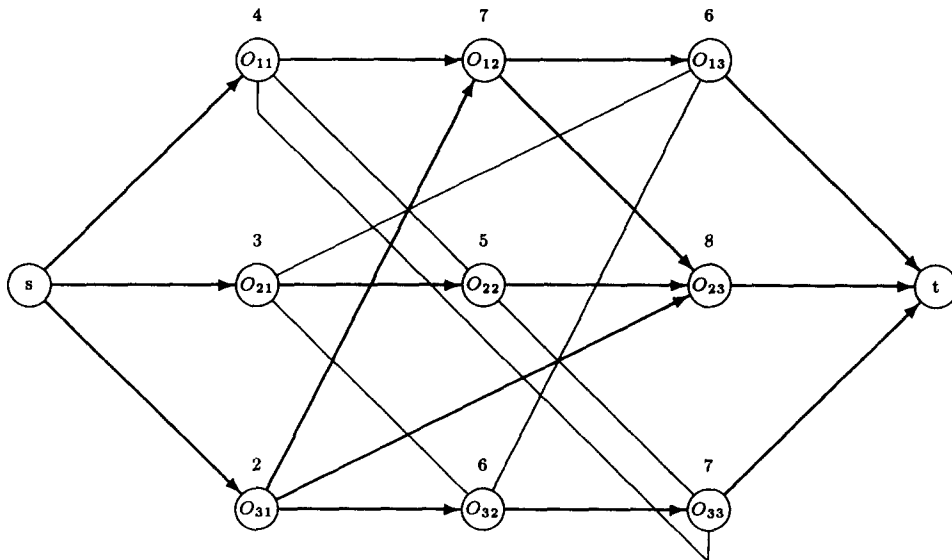


Fig. 2. Disjunctive graph after step 3.

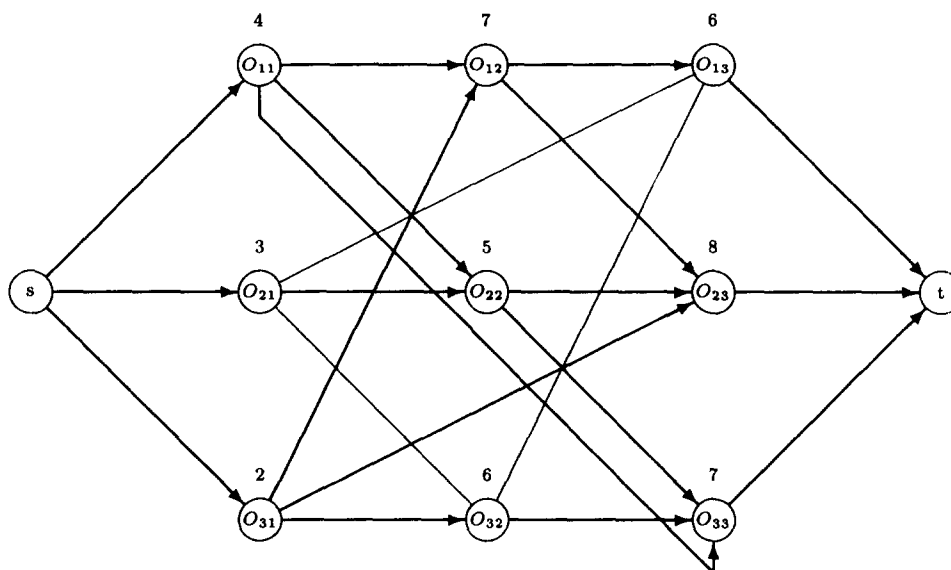


Fig. 3. Disjunctive graph after sequencing M_1 and M_3 .

3. Recomputing the virtual release and due dates results in:

	Release dates			Due dates		
	r_{i1}	r_{i2}	r_{i3}	d_{i1}	d_{i2}	d_{i3}
J_1	0	4	11	3	10	18
J_2	0	4	11	5	10	18
J_3	0	2	9	3	11	18

4. Since $|\Omega| = 2$, we reschedule both machines M_1 and M_2 . We find that rescheduling gives the same sequence in both cases, so no adjustment is required.

Third iteration:

- Schedule M_2 . The optimal sequence is (O_{21}, O_{32}, O_{13}) .
- Orienting the edges gives the graph depicted in Fig. 4.
- If we recompute the virtual release and due dates we find no change. The algorithm is therefore terminated. (The reader may easily verify that the final schedule is optimal.)

The above outlined scheduling procedure has been extended by Schutten et al. [17] and Meester

and Zijm [18, 19] to deal with multiple resources, parallel machines at several stages and part-type-dependent setup times. Also transportation times and unequal availability times can be handled. A special case of the latter feature occurs when each machine m is available only from some time t_m onwards (where t_m may be positive for some machines m). It can be shown that the SBM proceeds in almost the same way, with equally good results, in such a situation. This feature will be used when developing a rolling horizon capacity planning procedure. That will be the topic of the next section.

4. Capacity planning using aggregate scheduling

In this section we describe a capacity planning tool that uses aggregate scheduling. This tool basically sets due dates of jobs arriving at a parts manufacturing job shop. To set the due date of an arriving job, we determine the earliest possible completion time of that job, which does not cause any other already present job to be late. A due date, once assigned to a job, cannot be modified.

The term "aggregate scheduling" relates to the fact that the job shop configuration considered in

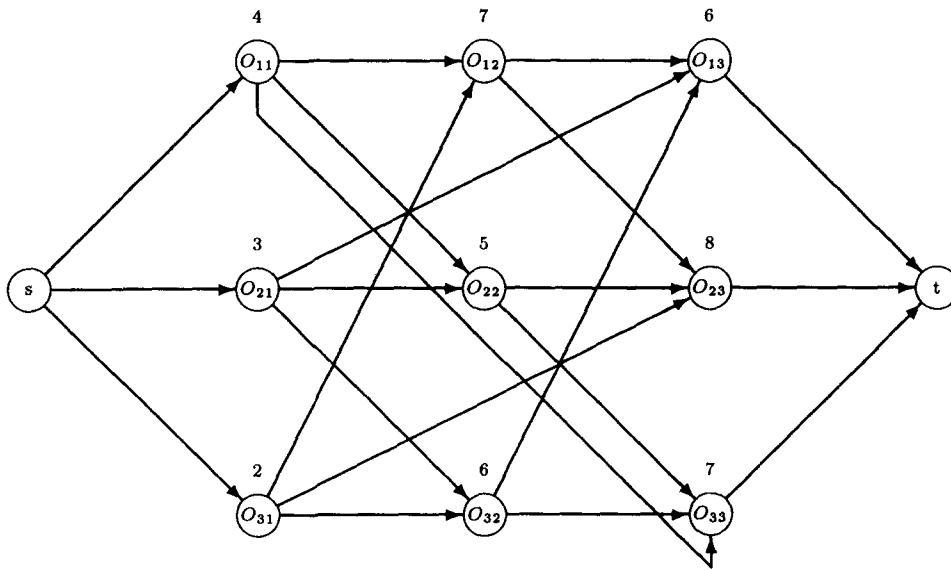


Fig. 4. Disjunctive graph for the solution of the example. Note that all machine edges have been directed.

the preceding two sections usually represents a highly simplified and aggregate picture of a more realistic machine shop. For instance, in one industrial case that we have studied, the machine shop consisted of a flexible manufacturing cell (consisting of three identical workstations with an integrated pallet pool and a centralized tool store), some stand-alone CNC machining centers with additional tooling and fixturing constraints, a workcell consisting of several identical lathes with part-family-dependent setup times, a workcell consisting of identical conventional drilling machines, etc. (cf. Ref. [20]). In addition, the presence of some specially skilled operators should be taken into account when developing a final shopfloor schedule. All the additional constraints, referring to cutting tool, fixture or operator availability, are usually not considered at the capacity planning level. Also setup times are added to the processing time of a batch while M parallel machines are replaced by one machine with a capacity which is M times as large. Only machine capacity is considered.

Now consider the due date assignment problem. Let, at some point in time t , a new job i of part type h arrive at the shop. The key question is: Given the

current work load, when can this job be delivered? or equivalently: Given the current work load, what should the planned lead time and thus the due date of the job be? For the moment we assume that the customer *cannot* negotiate a particular short lead time. Furthermore, neither working in overtime nor subcontracting is possible. All these assumptions will be relaxed at the end of this section.

In order to answer the above questions we apply a scheduling procedure taking into account all work orders previously accepted by the system which are not yet finished. Not all these jobs have necessarily been released to the shop already, but we assume that their release dates are known; for example, when their required materials arrive. Suppose that a due date has been assigned to each job present in the system, such that all present jobs can be finished in time. The following procedure then determines the planned lead time with release and due dates for job i . We will use the quantities defined in the previous sections.

The procedure is formally described as follows:

1. Specify a release date $r_i \geq t$ for a job i (arriving at time t), based on e.g. material availability (but *not* on workload considerations). Furthermore,

as an initial proposal, assign to job i the following provisional due date (see (8)):

$$d_i = r_i + \mathbb{E}T^{(h)}.$$

2. Apply the SBM to all jobs present in the system (including job i), with the following modifications of the job and machine data:
 - (a) For all jobs that are operated at some machine at time t , delete all operations that have started before t (including the operations that are still running at time t).
 - (b) For each machine that has started an operation before time t which will be completed at some time $t + s_m$ (with $s_m > 0$), increase its availability time to $t + s_m$.
 - (c) Reset the release dates of the (new) starting operations of all jobs.
3. If the planning procedure yields a positive lateness $L^* > 0$, increase the provisional due date and go to step 2. If $L^* < 0$ or $L^* = 0 > L_i$, decrease the provisional due date and go to step 2. If $L^* = L_i = 0$, stop. In the last case we have found the *smallest* date d_i at which job i can be completed without causing any other job to be completed too late.

Note that by applying the above procedure we end with a situation in which all jobs can be finished before or upon their due date. For this it is essential to apply a scheduling technique with the maximum lateness as a performance measure. Other performance measures such as the makespan or the average lateness cannot guarantee all jobs to be completed in time. A number of remarks regarding the above capacity planning procedure are in place:

- The essential difference between the capacity planning procedure given above and a classical “bucket filling” capacity planning procedures is that routing constraints are taken into account. In particular, if a machine typically at the beginning or at the end of the routings of many jobs is overloaded, this is clearly indicated by the above procedure. Classical bucket filling procedures fail to signal such occurrences because they “smooth” required capacity over the length of some planning period (a day or a week), ignoring precedence relations.
- The reader may verify that, if the first check of the provisional due date based on $\mathbb{E}T^{(h)}$ results in

a positive lateness L_i , while in addition all other jobs already present in the system at time t can be finished in time according to the derived schedule, then the *definitive* due date of job i should be set equal to $r_i + \mathbb{E}T_h + L^*$, and the remaining part of the procedure can be skipped. If some other jobs are also late, then $r_i + \mathbb{E}T_h + L^*$ is a lower bound for job i 's due date.

- Step 3 of the above capacity planning procedure can be implemented by a bisection procedure. This requires an initial lower bound and an initial upper bound for the due date of job i . The initial lower bound is given by the sum of job i 's release date r_i and the total of all processing times for job i . The initial upper bound is obtained by adding job i to the back of the current schedule (without job i). That is, on each machine job i is scheduled after the jobs that were scheduled earlier. The upper bound for the due date of job i is given by the completion time of the last operation of job i in the resulting schedule. In the procedure, if a positive L^* is found, the next provisional due date for job i is then equal to the mean of the current provisional due date and the upper bound, while the lower bound is reset to the current provisional due date. If a negative L^* is found the next provisional due date for job i is chosen equal to the mean of the lower bound and the current provisional due date.
- The mean lead times presented in Section 2 are not necessary to start the procedure. Any value would do.
- The procedure presented in Section 3 is easily adapted by including a fixed delay between each two consecutive operations of a job (cf. Ref. [19]). Such a delay may cover a transport time but may also serve to absorb small disturbances during some operation without influencing the succeeding operations of the same job. The fixed delays could then provide some robustness against small deviations of e.g. planned processing times.
- The capacity planning procedure described above is easily adjusted when one allows some jobs to be completed after their due dates. Also, a fixed slack per job or per operation is easily incorporated to provide some flexibility in accepting rush orders.

- If the resulting planned lead times are becoming too large (e.g. when a temporary overload occurs), the procedure can be used to examine the consequences of overtime work on some bottleneck machines (where a bottleneck machine is defined as the machine causing the largest lateness). As mentioned earlier, the scheduling procedure works well under unequal machine availability conditions [19]. Obviously, the effects of shifting work to subcontractors can be evaluated by the same procedure, by skipping some jobs or operations.

The ultimate result of the capacity planning procedure can of course only be examined after the determination of the detailed shop floor schedules at a short term, taking into account all the additional resources mentioned earlier. Indeed, we assume that the limited availability of cutting tools, fixtures, transports means and operators have only minor influence. If this does not hold for any particular resource, it should be taken into account at the capacity planning level already. Since the scheduling procedure described in Section 3 can be extended to incorporate multiple simultaneous constraints [19] this presents no essential difficulties.

The above procedure has been explained for a job shop configuration. Typical examples are parts manufacturing shops in metal working industries. The procedure can be extended to cover the assembly phase too, even though the structure of an assembly department is often more flow line than job shop oriented. The principal condition is that the assembly phase should fit in the decomposition procedure. For this assembly phase special scheduling algorithms have to be developed. In passing we note that the shifting bottleneck algorithm can be extended to fit convergent product structures (assembly of components in a single end item). Hence, the procedure can be extended to fit all fabrication and assembly stages corresponding to the levels of a Bill of Materials.

5. Numerical experiments

In this section we compare the capacity planning procedure outlined above (to be referred to as

AS = Aggregate Scheduling) with some alternative methods to determine the due dates of arriving orders.

We have performed experiments on job shops in which each job class has a fixed routing and each workstation has only one machine. We have considered four due date setting policies:

1. MRP type due date setting (FL):

$$d_i = r_i + \mathbb{E}T_h + 1.61 \text{ var}(T_h),$$

where $\mathbb{E}T_h$ is the expected lead time of part type h if we model the job shop as a network of $M|G|1$ nodes. Expressions for $\mathbb{E}T_h$ and $\text{var}(T_h)$ have been presented in Section 2. This policy assigns a fixed planned lead time, only based on job characteristics. Note that 1.61 is the 95%-confidence bound for the standard normal distribution.

2. Work-load-dependent due date setting (LL):

$$d_i = r_i + \mathbb{E}T_h + \max(\text{real}(\text{WOR}_i) - \mathbb{E}(\text{WOR}_i), 0),$$

where WOR_i is the amount of work in the shop that has to be processed on work stations on the routing of job i . The term $\mathbb{E}(\text{WOR}_i)$ is the expected value of WOR_i . This expectation is the sum of the terms $\mathbb{E}V_j$ in Eq. (13) that apply to the machines on job i 's routing. The term $\text{real}(\text{WOR}_i)$ is the amount of work found on the routing of job i at its release date.

3. Due date setting based on aggregate scheduling (AS): see Section 4.
4. Work load control (WC): This policy uses the same due date assignment as policy 2, but differs because it applies another dispatching policy. An arriving job is released to the shop only if the current work load is below a predetermined value. A job that cannot be released to the shop is kept in a backlog (or dispatch area). At every job arrival it is checked if any job in the backlog can be released to the shop. The due date of the job is assigned at its arrival time.

To investigate the results of the four procedures FL, LL, WC and AS, various configurations have been studied. For each configuration jobs of various part types were generated according to a Poisson process (with type-dependent parameter). Below, we discuss in detail the results of one example with 7 part types and 6 machines. The

maximum number of operations per part equals 4, lot sizes are all equal to 1. The data of this configuration are presented in Table 2. In this table a routing is presented as a sequence of tuples of machine numbers and processing times. Note that the average work loads of the machines vary significantly; we find $\rho_1 = 0.8$, $\rho_2 = 0.6$, $\rho_3 = 0.5$, $\rho_4 = 0.6$, $\rho_5 = 0.5$ and $\rho_6 = 0.9$.

First we generate jobs for each part type until the shop is in steady state and next we investigate the results obtained by the four planning procedures outlined above for a sequence of 3500 jobs (hence approximately 500 jobs of each part type). Table 3 shows the resulting averages of the planned lead times under each of the four policies.

Note that although WC and LL set planned lead times by the same method, WC nevertheless results in longer planned lead times, because the amount of work in the shop is usually higher. Also note that the variation in the planned lead times for the various parts are much larger under FL, LL and WC than under AS (in particular jobs 5, 6 and 7 have to be processed by the bottleneck machine 6).

Table 2
The data of the test configuration

Part type	Arrival rate	NOP	Routing
1	0.10	3	(1,3)-(2,4)-(3,2)
2	0.10	2	(4,5)-(3,1)
3	0.10	3	(5,3)-(2,1)-(3,1)
4	0.10	4	(5,1)-(1,1)-(2,1)-(3,1)
5	0.10	2	(5,1)-(6,1)
6	0.10	2	(1,2)-(6,1)
7	0.10	3	(1,2)-(4,1)-(6,7)

Table 3
Planned lead times

Part type	PLT-FL	PLT-LL	PLT-WC	PLT-AS
1	26	16	25	13
2	17	10	17	14
3	15	9	19	13
4	24	14	27	14
5	74	36	39	19
6	78	40	43	17
7	90	50	57	29

The AS procedure tends to smooth these differences (it does not use ET_n). Finally, the AS procedure leads to smaller planned lead times on average.

Table 4 shows the standard deviation of the lateness under the four policies. This table clearly shows the advantage of working with an approach that is based on aggregate scheduling. The due dates are very well predicted under the AS procedure.

Table 5 shows overall results.

The superb results of the AS policy with respect to the maximum lateness are not surprising; the capacity planning procedure based on aggregate scheduling is designed to yield a maximum lateness equal to zero. However, note that also the *realized average* lead time is lower under the AS policy than under any other tested policy.

It is important to realize that the due dates set by the AS policy are always met, regardless the jobs that arrive later. For instance, a similar due date setting policy procedure would not work if the jobs were served according to a *FCFS* discipline since then indeed operations can be delayed due to operations of jobs that have arrived later but are released to a machine earlier (as is possible in a job shop configuration).

Table 4
The standard deviation of the lateness

Part type	SDL-FL	SDL-LL	SDL-WC	SDL-AS
1	2	3	9	3
2	2	3	6	3
3	1	2	9	1
4	3	3	14	4
5	21	22	11	4
6	20	21	12	5
7	22	21	13	3

Table 5
Overall results

Quantity	FL	LL	WC	AS
Maximum lateness	26	14	29	0
Average lateness	-27	-6	-9	-1
Standard dev. lateness	23	8	12	3
Average shop time	19	19	23	16

One may wonder why under the AS policy a positive standard deviation appears. Why are not all jobs finished exactly on their due date? The explanation follows from the fact that the Shifting Bottleneck Method is a heuristic method, not necessarily always leading to an optimal result. A careful analysis of the procedure shows indeed that the overall lateness can be decreased sometimes due to an interference with some later arrived job. We will not treat this analysis here; the current results of the CP procedure are more than satisfactory and serve the main purpose: *to define reliable planned lead times and to guarantee that once set due dates are met.*

A large number of job shop configurations have been studied in the same way as shown above. In all cases the results bear similar characteristics as those shown above and are therefore not discussed any further. The due dates set at the capacity planning level serve as an input for a final shop floor scheduling procedure which treats capacities in much more detail. The final scheduling procedure should take into account operator and cutting tool availability constraints, batching with respect to family structures and the like. A hybrid shop floor scheduling system which can deal with these constraints is discussed in Ref. [19]; see also Ref. [18, 17, 21]. The basis for this scheduling system is again formed by the Shifting Bottleneck decomposition approach.

6. Conclusions and a preview on future research

In this paper we have pointed to a number of major drawbacks of existing manufacturing planning and control systems. In particular, the separation between lead time and capacity management is not justified, since lead times are directly related to actual machine utilization rates. Therefore, an alternative capacity planning and lead time management system has been proposed, which explicitly considers work-load-dependent lead times. This approach starts with the mean lead times determined by a rough queueing analysis and next exploits an aggregate scheduling procedure to calculate work-load-dependent planned lead times. The release and due dates determined at the capa-

city planning level provide inputs for the detailed shopfloor scheduling, taking into account additional (multiple) resources as well as setup characteristics, job clustering and job splitting effects.

Future research will be directed to capacity planning procedures for convergent and divergent product structures. Some preliminary experiments have indicated that the decomposition procedure described in Section 3 performs well in product structures covering both assemblies and component commonality. Finally, the relationships with the process planning in a make-to-order company will be investigated. In particular, the use of alternative process plans to reach a more balanced shop may provide a useful tool to facilitate capacity planning.

References

- [1] Vollmann, T.E., Berry, W.L. and Whybark, D.C., 1992. *Manufacturing Planning and Control Systems*, 3rd ed. Irwin, Homewood, IL.
- [2] Karmarkar, U.S., 1987. Lot sizes, lead times and in-process inventories. *Mgmt Sci.*, 33: 409–418.
- [3] Bertrand, J.W.M. and Wortmann, J.C. 1981. *Production control and information systems for component manufacturing shops*, Elsevier Amsterdam.
- [4] Wiendahl, H.P., 1987. *Belastungsorientierte Fertigungssteuerung*. Carl Hanser, Munich.
- [5] Goldratt, E.M., 1988. Computerized shop floor scheduling. *Int. J. Prod. Res.*, 26: 433–455.
- [6] Lenderink, A., 1994. *The Integration of Process Planning and Machine Loading in Small Batch Part Manufacturing*. Ph.D. Thesis, University of Twente.
- [7] Karmarkar, U.S., Kekre, S. and Kekre, S., 1985. Lotsizing in multi-item multi-machine job shops. *IIE Trans.*, 17: 290–297.
- [8] Kleinrock, L., 1975. *Queueing Systems, Vol. 1: Theory*. Wiley, New York.
- [9] Khuen, P.J., 1976. Approximate analysis of general queueing networks by decomposition. *IEEE Trans. Commun.*, 27: 113–126.
- [10] Whitt, W., 1983. The queueing network analyzer. *Bell Systems Tech. J.*, 62: 2779–2815.
- [11] Buzacott, J.A. and Shantikumar, J.G., 1993. *Stochastic models of Manufacturing Systems*. Prentice-Hall, Englewood Cliffs, NJ.
- [12] Bitran, G.R. and Tirupati, D., 1989. Multiproduct queueing networks with deterministic routing. *Mgmt. Sci.*, 34: 75–100.
- [13] Suri, R., 1988. RMT puts manufacturing at the helm. *Manuf. Eng.*, 100: 41–44.

- [14] Adams, J., Balas, E. and Zawack, D., 1988. The shifting bottleneck procedure for job shop scheduling. *Mgmt. Sci.*, 34: 391-401.
- [15] Carlier, J., 1982. The one machine sequencing problem. *Eur. J. Oper. Res.*, 11: 42-47.
- [16] Balas, E., Lenstra, J.K. and Vazacopoulos, A., 1995. The one-machine problem with delayed precedence constraints and its use in job shop scheduling. *Mgmt. Sci.*, 41: 94-109.
- [17] Schutten, J.M.J., van de Velde, S.L. and Zijm, W.H.M., 1993. Single-machine scheduling with release dates, due dates and family setup times. Working paper LPOM-93-4, Department of Mechanical Engineering, University of Twente. *Mgmt. Sci.*, accepted.
- [18] Meester, G.J. and Zijm, W.H.M., 1993. Multi-resource scheduling of an FMC in discrete parts manufacturing, in M.M. Ahmad and W.G. Sullivan (Eds.), *Flexible Automation and Integrated Manufacturing 1993*. Atlanta. CRC Press, Boca Raton, FL, pp. 360-370.
- [19] Meester, G.J. and Zijm, W.H.M., 1994. Development of a shop floor control system for hybrid job shops. Working paper LPOM-94-1, Department of Mechanical Engineering, University of Twente. *Int. J. Prod. Econ.*, accepted.
- [20] Slomp, J. and Zijm, W.H.M., 1993. A manufacturing planning and control system for a flexible manufacturing system. *Robot. Comput. Integrated Manuf.*, 10: 109-114.
- [21] Schutten, J.M.J. and Leussink, R.A.M., 1994. Parallel machine scheduling with release dates, due dates and family setup times. Working paper LPOM-94-8, Department of Mechanical Engineering, University of Twente. *Int. J. Prod. Econ.*, accepted.