Order acceptance in job shop environments

M.J.R. Ebben, E.W. Hans & F.M. Olde Weghuis

WP-91

BETA-publicatie: WP-91
ISSN: 1386-9213
NUGI: 684
Enschede: March 2003
Order acceptance in job shop environments

M.J.R. Ebben, E.W. Hans and F.M. Olde Weghuis

University of Twente, School of Business, Public Administration and Technology

P.O. Box 217, 7500 AE, Enschede, The Netherlands

Abstract

In this paper we investigate the importance of a good order acceptance method in over-demanded job shop environments. We present several approaches for integrated order acceptance and resource capacity loading. These approaches range from straightforward to sophisticated. The straightforward approaches consider capacity restrictions at an aggregate level, and ignore some technological constraints, such as precedence relations. The sophisticated methods do consider these technological restrictions, and consider capacity restrictions on individual resources. We use a simulation model of a generic job shop to test these approaches in a rolling horizon setting. We compare the performance of the approaches based on criteria such as delivery performance, robustness with respect to processing time uncertainty, and capacity utilisation. The test results show that the sophisticated approaches significantly outperform the straightforward approaches in case of tight due dates (little slack). In that case, improvements of up to 30% in utilisation rate can be achieved. In case of much slack, a sophisticated order acceptance method is less important, because there is a lot of flexibility on the loading and scheduling level to find a feasible solution.

Keywords: order acceptance, capacity planning, resource loading, simulation, job shop, manufacturing
1 Introduction

Order Acceptance (OA) is a tactical managerial activity that deals with accepting and rejecting incoming customer orders. It may also deal with related decisions, such as due date quotation and price determination. The order acceptance strategy has a large influence on the performance of a company. Accepting too many orders leads to an over-loaded production environment, where lead times increase and orders are increasingly delivered late. To deal with these problems management may try to find additional production capacity, for example by working in overtime, or by outsourcing. This additional irregular capacity normally incorporates significantly higher costs, which leads to lower or even negative profits. Tardy deliveries may also lead to higher (penalty) costs, and possibly lost customers. A good order acceptance strategy might be able to prevent these problems.

In practice, decisions on order acceptance and production planning are often functionally dispersed. For example, a sales department is concerned with order acceptance, while a production department is concerned with the production planning (loading and scheduling). The sales department will tend to accept all orders, regardless of the available capacity, because their goal is turnover. The production department tries to maximise utilisation and minimise the number of tardy deliveries. Given these conflicting goals, order acceptance decisions are often made without involvement of the production department or with incomplete information of the available capacity in the production department. The importance of order acceptance is often underestimated in practice and order acceptance is underexposed in the literature. Order acceptance related research is generally confined to single resource or flow shop type production systems. In section 2 we give an overview of the literature on order acceptance.

In this paper we study the order acceptance problem in a make-to-order (MTO) job shop environment. The MTO environment is typically characterized by small batches and a large
variety of products and routings through the shop. Order characteristics such as routings and material and machine requirements are generally not fully known at the stage of order acceptance. As a result, it is extremely difficult to measure the impact of accepting new customer orders on the operational performance of the production system. Especially in such environments a good finite capacity based order acceptance procedure is crucial. Ideally, such an OA method would use the available information as good as possible. Moreover, it would utilise as much of the capacity flexibility as possible, such as using non-regular capacity, and rescheduling already accepted orders. How to do this is far from trivial.

We investigate the importance of order acceptance and the benefits of cooperation between the sales and planning function. Based on recently developed advanced resource loading techniques and based on straightforward loading methods we developed several capacity based order acceptance methods. To test these methods, we developed a simulation model of a generic MTO job shop, which enables us to simulate the order arrival and production process. In this simulation model, we can easily plug in and test various OA methods and resource capacity loading methods in an online setting. By changing model parameters we can easily test the OA methods in different types of production systems. As performance measures we use the utilisation rate and the service level, i.e. the number of accepted orders that is completed before the due date. In this research we want to answer the following questions:

- Can a significant performance increase be obtained by using sophisticated OA methods as opposed to a simple method?
- Which method performs best under which circumstances?
- Should (and how should) the presented methods be used in practice?

This paper is organized as follows. In section 2 we present and categorise the available literature on order acceptance. In section 3 we describe different capacity based order
acceptance methods, varying from simple to sophisticated methods. In section 4 we present the simulation model and the experimental settings. In section 5 we present the results of the simulation experiments. Finally, in section 6 we give our conclusions and directions for further research.

2 Literature

Order acceptance has received limited attention in literature. Most papers on order acceptance consider the single resource case with deterministic processing times. Slotnick and Morton (1996) present an optimal solution for the static problem in which all order arrivals are known in advance and the problem is to select a subset of orders that maximizes revenues. The authors compare an optimal branch-and-bound approach with two heuristics and show that the heuristics work well for the given setting. Lewis and Slotnick (2002) extend this work to the multi-period case, for which they use a dynamic programming approach to find an optimal solution.

The assumption that all order arrivals are known in advance is not very realistic. Therefore, most other papers focus on the dynamic case, where orders arrive irregularly over time. In this case, the acceptance decision can be made upon arrival of each order, or upon a number of orders that have arrived in a specific decision period. Wang et al. (1994) propose a neural network approach in a periodic review setting to accept orders according to multiple criteria, such as profit and customer credit. Mainegra Hing et al. (2002) and Snoek (2000) also propose neural network based approaches to derive order acceptance policies in a single and multi-resource setting, respectively. We emphasise the difference between an order acceptance policy as opposed to an order acceptance algorithm. An order acceptance policy makes a quick order accept/reject decision based on the state of the system. An order acceptance algorithm requires optimisation every time an order arrives, which may require much computation time.
Another approach is order acceptance based on scheduling methods. Most scheduling based order acceptance research has focused on single resource production systems, such as: Akkan (1997), Wester et al. (1992) and Ten Kate (1994). Akkan (1997) suggests to accept orders only if they can be included in the schedule, such that it is completed before its due date, and without changing the schedule for already accepted orders. Wester et al. (1992) investigate three policies for order acceptance. A policy based on detailed scheduling (monolithic policy) and two workload based hierarchical policies. Simulation results show that the monolithic policy performs better than the hierarchic policies if the set-up times are large and the due dates are tight. In other cases the policies perform similarly. Ten Kate (1994) studies a similar setting, an integrated and hierarchical approach for a single machine case, and concludes that detailed information about the current production schedule is only valuable when lead times are short.

Raaymakers et al. (2000a) study the performance of workload rules for order acceptance in batch chemical manufacturing. They consider the total workload and the workload per work centre. It turns out that these methods do not perform well in their specific setting. Raaymakers et al. (2000b) compare a regression based makespan estimation policy with a workload-based policy for batch chemical manufacturing in a setting with deterministic processing times. When the utilisation is high and when there is a high variety in the job mix, the regression-based model outperforms the workload-based model. Ivanescu et al. (2002) build on this work by investigating uncertainty in processing times. Their results indicate that the regression policy can compete with the scheduling policy in situations with a high variety in the job mix and high uncertainty in the processing times.

We see that very limited attention has been paid to the multi-resource order acceptance problem. We will study this multi-resource case, for which we investigate both straightforward workload based policies, such as in Raaymakers et al. (2000a), as well as a sophisticated resource loading method (Hans, 2001). We will also investigate a production
system with stochastic processing times. As far as we know this has only been done by Ivanescu et al. (2002) for batch process industries with no wait restrictions and without precedence relations, which makes the results not directly viable for general job shop environments.

3 Order acceptance methods

In this section we describe four order acceptance methods that vary from straightforward to sophisticated. Orders arrive at the shop and consist of one or several jobs. These jobs have to be processed on a specified resource. On one resource several jobs cannot be processed in parallel; the whole resource capacity is claimed by one job. An order has a specified release and due date and between the jobs exist precedence relations. Jobs have to be assigned to one or more time periods. We assume that when a job is finished, the successor can immediately be started.

We focus on capacity based order acceptance methods to maximise the utilisation rate of the job shop. Because we focus on capacity, we use resource loading methods to support the order acceptance decision. Therefore, the names of the methods refer to this. First we introduce some notation.

\( p_{b,j,m} \): the estimated processing time of job \( b \) from order \( j \) on resource \( m \).

\( n_j \): the number of jobs in order \( j \).

\( r_j \): the release date of order \( j \).

\( d_j \): the due date of order \( j \).

\( M \): the number of resources (machine groups) in the job shop.

\( C_{t,m} \): the maximum available capacity on resource \( m \) in time period \( t \).

\( U_{t,m} \): the reserved capacity on resource \( m \) in time period \( t \), for example for already accepted orders.

\( \alpha \): expected utilisation rate.
3.1 Aggregate resource loading (ARL)

The Aggregate Resource Loading (ARL) method looks at aggregate information: the total (estimated) processing time for order \( j \) and the (estimated) available capacity between the release and due date of order \( j \). Capacity is aggregated over all resources. When the required total processing time is smaller than, or equal to the available capacity between the release and due date of order \( j \), then order \( j \) is accepted, see equation (1).

\[
\sum_{m=1}^{M} \sum_{b=1}^{n_j} p_{b,m} \leq \sum_{m=1}^{M} \sum_{i=r_j}^{d_i} (\alpha C_{i,m} - U_{i,m})
\]  

(1)

Note that this method does not take the different jobs of an order and the precedence relations between jobs into account. It also does not distinguish between the different resources. Therefore, this method overestimates the utilisation of the shop. To compensate for this we introduce the safety factor \( \alpha \), which can be seen as the estimated utilisation rate that can be achieved. The lower the value of \( \alpha \), the smaller the possibility that an order will be accepted that cannot be produced before its due date. However, when \( \alpha \) is low, the maximum utilisation rate is also low. Thus, \( \alpha \) should be chosen as large as possible, with the constraint that the service level must be above a given percentage, for example, the service level must be above 95%. The simulation model can determine the right value of \( \alpha \).

When an order is accepted, resource capacity for this order must be reserved in specific time intervals, which we refer to as periods. Several methods can be used to assign capacity to this order, such as forward loading and backward loading, as described by Akkan (1997). We use a forward loading method. Capacity is reserved starting from the release date of the order, say at the beginning of time period \( t \). The remaining capacity in time period \( t \) is \( \alpha C_r U_t \). When the processing time of order \( j \) is larger than the remaining capacity, \( U_t \) becomes equal to \( \alpha C_r \) and we go to the next time period \( t+1 \). We repeat this until the reserved capacity for order \( j \) is equal to the processing time of order \( j \).
3.2 **Resource loading per resource (RLR)**

Taking the precedence relations between the jobs into account may lead to a better estimation of the capacity utilisation and the possibility to complete the order on time. We use internal release and due dates to indicate the time window in which the jobs of an order must be processed on a particular resource. The internal job release and due dates can be calculated from the order release and due date, the job precedence relations, and the capacity restrictions. The time windows overlap when the order has slack, i.e. the required processing time is smaller than the available time between release and due date. An order is accepted when for each job capacity is available on the applicable resource within the time window of this job:

$$p_{ij} \leq \sum_{r} (\alpha C_{im} - U_{im})$$

(Disting)2

Distinguishing the different resource groups describes the situation in the shop more accurately than the previous method. Therefore, we expect that the value of $\alpha$ will be higher for this method than for the previous one. Time windows of jobs can overlap; a job can claim capacity within the time window of its successor. As a result, the precedence relations between the jobs are only partly satisfied. The next method does respect the precedence relations entirely, by loading jobs successively with a dispatching rule.

3.3 **EDD resource loading (EDD)**

The EDD resource loading method is based on the Earliest Due Date dispatching rule. When a new order arrives we construct a new plan with this new order included. In case one or more orders are tardy we reject the new order, otherwise we accept it. A significant difference with the previous methods is that once an order is accepted, the capacity assignment to time periods is not fixed, since we reschedule upon order acceptance. In the previous methods capacity is reserved when an order is accepted and this assignment is not changed afterwards. Furthermore, EDD does not violate precedence relations, as opposed to the previous methods.
Note that the loading plan found by EDD may therefore be more consistent with the operational plans at the lower level.

EDD uses the following parallel dispatching procedure. Starting in period $t = 0$, the decision set holds all jobs of all orders that are allowed to be dispatched. A job is allowed to be dispatched when: all its successors have completed, the current time period is on or past its release date, and the required resource is available. The priority of each job is then determined by its internal due date. We select the job with the highest priority, and load this job onto the concerning resource by adjusting its capacity profile. We then update the decision set. If the decision set is empty, we move to the first time period in which there are jobs that are allowed to be dispatched. We continue until all jobs have been planned.

The EDD loading method imitates an EDD dispatching procedure, which we could use for the operational planning / dispatching. This only holds when the input data is deterministic. When the input data is stochastic, or in case of processing disruptions, the loading method only gives an approximation of the operational plan. In this case, we may reserve capacity to deal with these uncertainties.

### 3.4 Branch-and-Price resource loading (BPRL)

The branch-and-price resource loading method (BPRL) is an exact approach for solving the pre-emptive resource loading problem. For an extensive description of this method we refer to Hans (2001). Here we give a brief description of this method.

The BPRL method revolves around an MILP formulation of the pre-emptive resource loading problem. The model distinguishes order plans and order schedules. Order plans indicate the periods in which each job of an order is allowed to be performed. A corresponding order schedule indicates per period the fraction of the job that is performed. The idea of the BPRL approach is that it generates order plans implicitly by column generation. The model must
select one order plan per order, and generate the corresponding order schedule. Since any selected order plan is feasible with respect to job release and due dates, precedence relations and minimum duration restrictions, the corresponding order schedule is also feasible to these restrictions. Also, since order plans are input of the model, this approach prevents that precedence relations are modelled explicitly, which would require many integer variables and restrictions.

The objective of the model is to generate an order schedule for each order, such that the total costs of the required non-regular capacity (working in overtime, hiring staff, and subcontracting) and the order tardiness penalties are minimized. The MILP model allows making trade-off analyses between due date performance on the one hand, and non-regular capacity levels on the other hand. The model can easily be adjusted to forbid either tardiness, or using non-regular capacity. This makes it very suitable for this research.

The (partial) integrality of the model is induced by the binary variables that select one order plan per order. Since there are exponentially many order plans, an explicit model of a problem of regular size is impossible to formulate and solve. Hans (2001) proposes various exact and heuristic methods, which are all based on first solving the linear programming (LP) relaxation of the MILP model by column generation. It starts from a restricted LP formulation (RLP), which has at least one order plan per order. After each RLP optimisation, order plans with negative reduced costs are added to the RLP, which must then be re-optimised. The column generation scheme terminates when no order plans with negative reduced costs exist. The optimal solution of the LP relaxation is then found. If the optimal LP solution happens to be integral, we have found an optimal solution of the resource loading problem. Otherwise, we could apply a branch-and-price algorithm to determine an optimal solution, or e.g. use a heuristic to judiciously round the LP solution to obtain a feasible solution. In this paper we use the branch-and-price approach. Hans (2001) shows that typically, this method finds good or even optimal solutions fast; however, the method requires much computation time to prove
optimality. Since we aim to use this method in a simulation model, in which it is called many
times, solving each problem to optimality may require too much computation time. We
therefore truncate the branch-and-price method after a specified time (in seconds). We
implemented BPRL in Delphi 7.0, using CPLEX 7.1 to optimise the RLP.

In the simulation model we use BPRL in the following way. The simulation model has
information about already accepted orders and about the newly arrived order. Furthermore,
there may be some jobs in process in the shop. These will not be reassigned; they claim part
of the available capacity. All this information is input for BPRL, which then computes an
order schedule. We accept the newly arrived order when none of the orders is tardy in the
order schedule; the use of non-regular capacity is not allowed. Note that BPRL allows pre-
emption. When we use BPRL also for dispatching the jobs, we take the selected order
schedules from BPRL and use these resource plans for dispatching the jobs on the resources.

4 Simulation model and experimental settings

We build our job shop simulation model in the object-oriented simulation package eM-Plant.
Many parameters can be varied in this simulation model. In this section we present our
experimental variables and the experimental setting. We distinguish three categories of
experimental variables: Order acceptance methods, order characteristics and shop
characteristics.

Order acceptance methods

In section 3 we presented four order acceptance methods. We want to test the performance of
these methods for several settings. These settings differ with respect to order and shop
characteristics. For ARL and RLR we also have to find the value of $\alpha$ that leads to the best
performance. When processing times are stochastic we introduce a parameter $\alpha$ for all
acceptance methods; because of the stochasticity in the processing times, some buffer
capacity must be reserved for these uncertainties. We truncate BPRL after 60 seconds when it
has not found the optimal solution or has not yet proven optimality. We use the best solution it has found so far to decide whether an order is accepted or rejected.

Order characteristics

Orders arrive according to a Poisson process. For each of the order characteristics we choose a high and a low factor level (see table 1). The number of jobs per order is uniformly (discretely) distributed between 1 and 3 at the low factor level and is uniformly distributed between 3 and 5 jobs per order at the high level. The variance in the processing time per job ($p_{bjm}$) can be low, uniformly distributed between 10 and 14 hours, or high, uniformly distributed between 6 and 18 hours. These processing times can also be deterministic or stochastic, representing the uncertainty in the processing times. When the processing time is stochastic, the processing time is perturbed by a factor that is uniformly distributed between 0.75 and 1.25. Orders that arrive at the shop always have some slack. This slack can be low, 0.5 times the processing time of the order, or high, 1.5 times the processing time of the order. A lot of slack makes it easier to account for uncertainties and leaves flexibility for dispatching the orders.

<table>
<thead>
<tr>
<th>Factor</th>
<th>Low</th>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Number of jobs per order</td>
<td>$U(1,3)$</td>
<td>$U(3,5)$</td>
</tr>
<tr>
<td>2. Processing time per job</td>
<td>$U(10,14)$</td>
<td>$U(6,18)$</td>
</tr>
<tr>
<td>3. Uncertainty in processing times</td>
<td>deterministic</td>
<td>$U(0.75,1.25)\cdot p_{bjm}$</td>
</tr>
<tr>
<td>4. Workload (demand/capacity ratio)</td>
<td>0.75</td>
<td>1.0</td>
</tr>
<tr>
<td>5. Slack per order</td>
<td>0.5$p_j$</td>
<td>1.5$p_j$</td>
</tr>
</tbody>
</table>

Table 1: Levels of the experimental factors

Shop characteristics

A job shop contains a number of resources. In our experiments there are 5 different resources in the shop. The resource on which a job must be processed is selected randomly. We include the constraint that 2 consecutive jobs of the same order are not allowed to be processed on the
same resource. The workload that arrives at the shop is expressed by a demand/capacity ratio. At the low level, the average demand requirements are 75% of the total available capacity, whereas the demand requirements are 100% at the high level. The utilisation in the job shop is between 40% and 75%, depending on the settings of the experimental variables. The settings for demand both represent situations where demand is close to or exceeds the available capacity.

For the dispatching of accepted orders in the shop we use two approaches. First, we use a simple EDD priority rule per resource. When the resource becomes available the job with earliest (internal) due date that is waiting in the buffer for this resource is selected. Second, in combination with the BPRL loading method (see section 3.4), we use this method for the dispatching in the shop. The order schedules produced by BPRL are used for dispatching in the shop.

We perform experiments for all combinations of the experimental factors in table 1 to determine which factors have a significant effect on the utilisation rate of the shop. The utilisation rate is the used capacity divided by the total available resource capacity over the complete simulation run length. Each run has a length of 360 days, including a warm-up period of 20 days. For each factor level combination we execute 5 runs with different seed values. We vary the factor $\alpha$ for the different OA methods and we select the results of the experiment with the highest $\alpha$, for which the percentage of tardy orders is below 5%. Thus, we demand a service level of at least 95%.

5 Computational results

Effects of the experimental factors

In table 2 we see the average effect of the experimental factors on the utilisation rate in the shop. Each number in table 2 shows the absolute change in the utilisation rate as a consequence of increasing the factor level from low to high. The results in table 2 give some
idea about the effects, but to draw conclusions we also have to look in more detail at the results for specific settings.

<table>
<thead>
<tr>
<th>OA method</th>
<th>#jobs/order</th>
<th>Processing time</th>
<th>Uncertainty</th>
<th>Workload</th>
<th>Slack</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARL</td>
<td>0.01</td>
<td>-0.01</td>
<td>-0.02</td>
<td>0.02</td>
<td>0.24</td>
</tr>
<tr>
<td>RLR</td>
<td>-0.04</td>
<td>-0.02</td>
<td>-0.03</td>
<td>0.05</td>
<td>0.23</td>
</tr>
<tr>
<td>EDD</td>
<td>0</td>
<td>-0.01</td>
<td>-0.03</td>
<td>0.06</td>
<td>0.15</td>
</tr>
<tr>
<td>BPRL</td>
<td>-0.03</td>
<td>-0.01</td>
<td>-0.02</td>
<td>0.07</td>
<td>0.12</td>
</tr>
</tbody>
</table>

Table 2: Average effect of the experimental factors on the utilisation (absolute difference).

The number of jobs per order has moderate impact on the utilisation rate. A higher utilisation rate implies that more orders are accepted. ARL accepts more orders than the other OA methods, because it does not consider precedence relations. The OA methods are only slightly sensitive to the variance in processing times. Especially in case of little slack the utilisation is moderately lower in case of a high variance in processing times. The results show that increasing processing time stochasticity results in slightly lower utilisation rates. In the stochastic case some capacity must be reserved for the uncertainty in processing times. All four methods show nearly the same effect (see also figure 1). With respect to the workload, the more orders arrive at the job shop, the higher the probability that one of the orders fits in the existing schedule. EDD and BPRL take most advantage of the higher number of arrivals. A reason is that these rules compute a new schedule every time an order arrives. They have the flexibility to move already accepted orders in time. This may not always be allowed in practice, but it is beneficial to do so. The slack of an order has the largest impact on the utilisation rate. Little slack makes it more difficult to fit a new order in the existing schedule. EDD and BPRL reoptimise every time, and can deal better with little slack. As a consequence, as slack increases, the simple methods show a larger increase in their utilisation rate. These methods can accept a lot of orders without worrying about the dispatching in the shop.
Figure 1: Relation between “number of jobs per order / processing time uncertainty” (LH=Low/High) and the utilisation rate, given a high variance in processing times, high workload and little slack.

Comparison of the order acceptance methods

We looked at the effect of several factors on the performance of the OA methods, and we are especially interested in the performance differences between the methods. We compare the different methods for all factor level combinations, using a paired t-test to construct a 95% confidence interval. Table 3 shows the average utilisation for the different order acceptance methods.

<table>
<thead>
<tr>
<th>OA method</th>
<th>Average utilisation rate over all experiments</th>
<th>Average utilisation rate for low slack and high demand experiments</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARL</td>
<td>0.56</td>
<td>0.46</td>
</tr>
<tr>
<td>RLR</td>
<td>0.62</td>
<td>0.53</td>
</tr>
<tr>
<td>EDD</td>
<td>0.64</td>
<td>0.60</td>
</tr>
<tr>
<td>BPRL</td>
<td>0.65</td>
<td>0.62</td>
</tr>
</tbody>
</table>

Table 3: Average utilisation for the different order acceptance methods.
RLR has a better performance than ARL, because RLR considers the capacity of the individual resources instead of the total capacity. EDD also shows significantly better results than ARL. On average, EDD achieves an additional 0.08 utilisation rate, which is an increase of 14% (see table 3). Note that the methods perform similarly in cases with a lot of slack (see figure 2). The advantages of EDD emerge when the workload is high and there is little slack (last column table 3). In that case, EDD achieves a utilisation rate of up to 0.15 higher than ARL, which is an improvement of 30% (figure 2). Compared with RLR, EDD shows an improvement of up to 15%. EDD performs better because it considers the capacity of the individual resources, accounts for all precedence relations, and reschedules upon order arrival. In case of high slack, RLR and EDD perform alike. BPRL performs slightly better than EDD. This can be explained by the fact that an EDD priority rule is used for dispatching the orders. A larger improvement might be expected when BPRL is used for both order acceptance and dispatching in the shop. We come back to this later.

Figure 2: Relation between “slack/workload” (LH=Low/High) and the utilisation rate, given that the other factors are at the high level.

Computation times
The computation times for ARL, RLR and EDD are negligible. BPRL requires a much longer computation time, and even longer when slack increases. The reason for this is that the number of feasible solutions grows exponentially. We see that in cases with little slack and a low workload a good solution is found within 10 seconds on a Pentium 800MHz. When there is much slack, more than 60 seconds is required to find a good solution. In figure 2 we see that with much slack BPRL achieves a lower utilisation than EDD, because BPRL is truncated after 60 seconds. However, in these cases the straightforward order acceptance methods also perform satisfactorily. BPRL is especially useful in the more difficult cases with little slack, high demand and several jobs per order.

![Graph showing the relation between α and the utilisation rate for the different OA methods.](image)

**Figure 3: Relation between α and the utilisation rate for the different OA methods.**

*The value of the parameter α*

An advantage of EDD and BPRL in the deterministic case is that we do not need to set the parameter α (α=1). The setting of this parameter for ARL and RLR strongly depends on the values of the experimental factors. The service level is required to be at least 95%. Accordingly, for ARL α is between 0.3 and 0.7. For RLR α is between 0.7 and 1.0. This indicates that RLR represents the situation in the shop better than ARL. In figure 3 and figure 4 we show the relation between α and the utilisation rate and between α and the service level for the different OA methods. Since ARL aggregates all capacity, the utilisation rate is close
to $\alpha$, as can be seen in figure 3. Figure 4 shows that for ARL and RLR the service level drops sharply with increasing $\alpha$, while the results of EDD and BPRL are less sensitive to the value of $\alpha$. For all these experiments $\alpha$ is between 0.8 and 1.

![Graph showing the relationship between $\alpha$ and service level for different OA methods.](image)

**Figure 4: Relation between $\alpha$ and the service level for the different OA methods.**

It is difficult to give the best value of $\alpha$ for a particular service level requirement, but there are some general observations. When the slack of the orders increases, $\alpha$ can also increase. More slack gives more flexibility in dispatching the orders and gives a higher utilisation rate. When the uncertainty increases, for example because of processing time stochasticity, the best value of $\alpha$ decreases. In this case, capacity has to be reserved for the uncertainties.

**Using BPRL for dispatching**

We can also use BPRL instead of EDD for dispatching. The order schedules produced by BPRL are used for the dispatching in the shop. We performed experiments with little slack, the difficult cases for which the BPRL order acceptance method showed the best results. For these cases there is only a slight improvement (0.01) in utilisation rate by using BPRL for dispatching. As may be expected, since it is designed to operate at a higher planning level, BPRL is perhaps not suitable as a scheduling / dispatching method.
6 Conclusions and recommendations

In job shop order acceptance it is important to control the workload. Accepting too much work leads to a low service level, whereas accepting too few orders leads to a low utilisation rate. We have demonstrated sophisticated OA methods that can significantly increase the utilisation rate given a pre-specified service level. An increase in the utilisation rate of up to 0.15 (30% improvement) can be realized in cases with little slack and high workload.

EDD and BPRL perform the best in difficult situations (cases with little slack and high demand), where BPRL shows slightly better results than EDD. In these cases it turns out to be beneficial to reschedule the already accepted orders when a new order is accepted. For simpler cases with a lot of slack all methods show acceptable results. There is a lot of flexibility for the loading and scheduling level to find a feasible solution. Nevertheless, EDD might be preferred for these cases when looking at performance, the setting of $\alpha$, and the computation times.

BPRL did not outperform EDD. One reason may be that we gave BPRL a limited computation time, to have acceptable experiment duration. In practice, perhaps more computation time is available. Another reason might be that we did not use all aspects of BPRL, such as the possibility to make a trade-off between due date performance and the use of non-regular capacity. The other methods cannot make such a trade-off. Therefore, it might be interesting to investigate the problem from a revenue-based perspective, considering non-regular capacity and the profits of the different orders.

The job shop simulation model reflects the situation of a make-to-order manufacturer in practice quite well. It is hard to compare the tested OA methods with the OA performance of a planner in practice, since a planner probably makes his decisions based on experience. Such an approach is hard to simulate. Computationally, we did show the importance of integrating
order acceptance with the resource capacity loading function (i.e. finite capacity order acceptance). It would be interesting to support our conclusions in a case study. The simulation model offers many possibilities for studying variants of the job shop OA problem, such as revenue based order acceptance, periodical order acceptance, learning methods for OA, order tardiness penalties, and planning irregular capacity.

References


Ivanescu, C.V., Fransoo, J.C., and Bertrand, J.W.M., 2002, Makespan estimation and order acceptance in batch process industries when processing times are uncertain. OR Spectrum, 24, 467-495.


