OR Spectrum (2005) 27: 107–122
DOI: 10.1007/s00291-004-0171-9

OR Spectrum

Workload based order acceptance in job shop environments

M.J.R. Ebben[1], **E.W. Hans**[2], **and F.M. Olde Weghuis**[2]

[1] TPG Post, Quantitative Support, P.O. Box 30250, 2500 GG The Hague, The Netherlands
[2] School of Business, Public Administration and Technology, University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands
(e-mail: e.w.hans@utwente.nl)

Abstract. In practice, order acceptance and production planning are often functionally separated. As a result, order acceptance decisions are made without considering the actual workload in the production system, or by only regarding the aggregate workload. We investigate the importance of a good workload based order acceptance method in over-demanded job shop environments, and study approaches that integrate order acceptance and resource capacity loading. We present sophisticated methods that consider technological restrictions, such as precedence relations, and release and due dates of orders. We use a simulation model of a generic job shop to compare these methods with straightforward methods, which consider capacity restrictions at an aggregate level and ignore precedence relations. We compare the performance of the approaches based on criteria such as capacity utilisation. The simulation 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.

Keywords: Order acceptance – Capacity planning – Resource loading – Simulation

1 Introduction

Order Acceptance (OA) is a tactical managerial activity that deals with accepting and rejecting 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 system, in which lead times increase and orders are

Correspondence to: E.W. Hans

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 separated, as we have experienced in a study at a shipyard. For example, a sales department is responsible for order acceptance, while a production department is responsible for the production planning. 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 of turnover and tardiness, order acceptance decisions are often made without involving the production department or with incomplete information on the available capacity in the production department. Zijm (2000) also mentions that it is important to integrate these decisions. The importance of order acceptance is often underestimated in practice and the issue of 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 and it would utilise as much of the flexibility as possible, such as 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 workload 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. We assume that there is an unlimited storage capacity for inventories between the processing steps. In this simulation model, we can easily plug in and test various OA methods and resource capacity loading methods in a dynamic 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 several capacity based order acceptance methods, varying from simple to sophisticated methods. In Section 4 we present the simulation model and the experimental setting followed by the results of the simulation experiments in Section 5. 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 exact solution method 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 exact 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. The difference between an order acceptance *policy* (Wang et al., 1994; Mainegra Hing et al., 2002; Snoek, 2000) as opposed to an order acceptance *algorithm* (cf. Akkan, 1997; Wester et al., 1992; Ten Kate, 1995) is that a policy has a given/predetermined accept/reject decision based on the state of the system. In case of an order acceptance algorithm, new computations are made every time an order arrives to make the acceptance decision, 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 (1995). 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 algorithms for order acceptance; an algorithm based on detailed scheduling (monolithic approach) and two workload based hierarchical approaches. Simulation results show that the monolithic approach performs better than the hierarchical approaches if the set-up

times are large and the due dates are tight. In other cases they perform similarly. Ten Kate (1995) 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 approach with a workload-based approach 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 approach can compete with the scheduling approach 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 algorithms, 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 their 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. The jobs can be regarded as an aggregation of operations. Thus, at the scheduling level, the jobs can be further disaggregated into operations. Detailed information on these operations is not available at the order acceptance planning level, since micro process planning has not been performed yet (cf. Zijm, 2000). Instead, through macro process planning only rough information on the jobs is available, such as precedence relations between the jobs. 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. 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 workload 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_{bjm}: 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 target due date of order j.
M: the number of resources in the job shop.
C_{tm}: the maximum available capacity on resource m in time period t.
C_t: the total maximum available capacity (sum of C_{tm})
U_{tm}: the reserved capacity on resource m in time period t, for example for already accepted orders.
U_t: the reserved capacity on all resources (sum of U_{tm})
α: 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. Similar approaches can be found in for example Wester et al. (1992) and Ten Kate (1995). 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).

$$\sum_{m=1}^{M} \sum_{b=1}^{n_j} p_{bjm} \leq \sum_{m=1}^{M} \sum_{t=r_j}^{d_j} (\alpha C_{tm} - U_{tm}) \tag{1}$$

Note that this method does not take the jobs of an order and the precedence relations between these jobs into account. It also does not distinguish between the various resources. Therefore, this method overestimates the utilisation of the shop. To compensate for this we introduce the safety factor α, which for this method can be seen as the maximum utilisation rate that can be achieved given the order and shop characteristics. A similar parameter is introduced by e.g. Wester et al. (1992), who call αC_t the critical work content level. The lower the value of α, the smaller the possibility that an order will be accepted that can not be produced before its due date. However, when α is low, the maximum utilisation rate is also low. Thus, α 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%. Simulation experiments can be used to find the value of that α corresponds with a given service level.

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 $\alpha C_t - U_t$. When the processing time of order j is larger than the remaining capacity, U_t becomes equal to αC_t 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)

Instead of looking at the aggregate capacity over all resources, we can look at the resource level. For example, Raaymakers et al. (2000b) looked at the capacity per workcenter. 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 (r_{bj} and d_{bj}) can be calculated from the order release and due date, the job precedence relations. The internal release date is the earliest possible time a job k can start, i.e. the earliest possible time all its predecessors can have been finished. The internal due date is the latest time a job k can finish such that there is still time for the successors to be processed before the due time:

$$r_{bj} = r_j + \sum_{k=1}^{b-1} p_{kjm}; \quad d_{bj} = d_j - \sum_{k=b+1}^{n_j} p_{kjm} \tag{2}$$

The time windows ([r_{bj},d_{bj}]) overlap when the order has slack. This follows from:

$$d_{bj} = r_{bj} + p_{bj} + s \overset{s>0}{>} r_{bj} + p_{bj} = r_{b+1,j} \tag{3}$$

An order is accepted when for each job capacity is available on the applicable resource within the time window of this job:

$$p_{bjm} \leq \sum_{t=r_{bj}}^{d_{bj}} (\alpha C_{tm} - U_{tm}) \tag{4}$$

Distinguishing between the resource groups describes the situation in the shop more accurately than the previous method. Therefore, we expect that for this method the value of α will be higher than for the previous one, where α still represents the maximum achievable utilization given a specified service level. 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 based order acceptance (EDD)

Another approach is making a new schedule every time an order arrives. Similar approaches for the single resource case can be found in Wester et al. (1992), Ten Kate (1995) and Akkan (1997). Raaymakers et al. (2000b) and Ivanescu et al. (2002) use a simulated annealing approach to make a detailed schedule. We use the Earliest Due Date dispatching rule to construct a detailed schedule. 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 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 predecessors 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.

This order acceptance method imitates an EDD dispatching procedure, which can be used for the operational planning. This only holds when the input data is deterministic. When the input data is stochastic, or in case of processing disruptions, the EDD schedule only gives an approximation of the operational plan. In this case, we may reserve capacity to deal with these uncertainties by using a parameter α in a similar way as in the previous methods. Note that the reason to introduce this parameter is different than for the previous methods: uncertainty instead of inaccuracy.

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. This problem is concerned with loading a set of orders onto a job shop consisting of several resources. 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 and precedence relations, the corresponding order schedule is also feasible to these restrictions. Also, since the possible 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 truncated branch-and-price approach. Hans (2001) shows that typically, this method finds 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 8.1.0 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 the order schedules of a BPRL solution allow job pre-emption. As a result, a job may not be executed in consecutive periods. Unfortunately, the BPRL approach cannot forbid pre-emption; we use the order schedules also 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

Table 1. Levels of the experimental factors

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

to order and shop characteristics. For ARL and RLR we also have to find the value of α that leads to the best performance. When processing times are stochastic we introduce a parameter α 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.

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 maximum achievable utilisation rate. Due to precedence relations and routing diversity it is impossible to achieve a utilisation of 100%. Therefore, orders have to be rejected in our settings.

For the dispatching of accepted orders in the shop we use two approaches. First, we use a simple EDD priority rule per resource for all order acceptance methods. When the resource becomes available the job with earliest (internal) due date that is waiting in the buffer for this resource is selected. Second, since the BPRL loading method allows pre-emption (see Sect. 3.4), we use the pre-emptive order schedules provided by this method to dispatch the jobs (Sect. 5.5). The order schedules indicate which part of which jobs has to be processed in a specific time period. These jobs are executed in a random order within this time period.

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 by constructing a paired 95% confidence interval and determining probability that the two settings do not have the same mean. When this probability (p-value) is smaller than 0.05 we assume the difference is significant. 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 α for the various OA methods, starting at 1 and decreasing in steps of 0.1 until we have found a solution that complies with the service level requirements. We select the results of the experiment with the highest α, for which the percentage of tardy orders is below 5%. Thus, we demand a service level of at least 95%.

5 Computational results

5.1 The value of the parameter α

The parameter α, which accounts for inaccuracy in the methods and uncertainty in processing times, has a serious impact on the results of the different methods. An advantage of EDD and BPRL in the deterministic case is that we do not need to set the parameter α ($\alpha = 1$). The shop and order characteristics, such as precedence relations, are taken into account by these methods. The setting of α 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 rather low, between 0.3 and 0.7, which can be explained by the fact that this method aggregates all resources and does not take precedence relations into account. 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 1 and Figure 2 we show the relation between α and the utilisation rate and between α and the service level for the various OA methods. In these experiments there is little slack, while all other experimental factors are set at the high level (cf. Table 4). Since ARL aggregates all capacity, the utilisation rate is close to α, as can be seen in Figure 1.

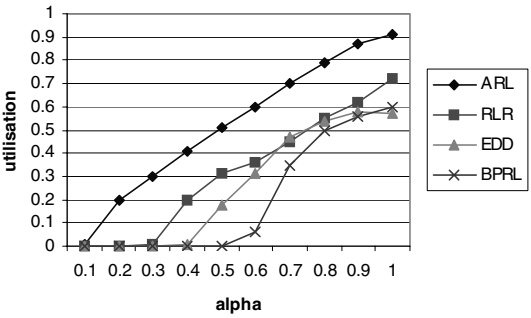

Fig. 1. Relation between α and the utilisation rate for the various OA methods

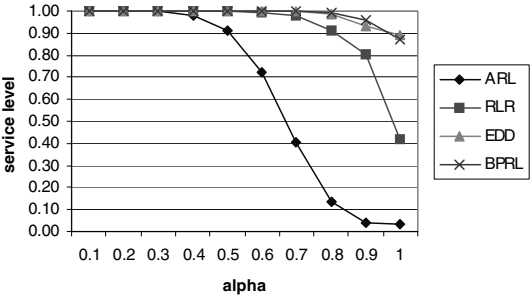

Fig. 2. Relation between α and the service level for the various OA methods

The utilisation rate for RLR, EDD and BPRL is very low with a low α. A reason is that these methods construct schedules in which the arrived orders cannot be completed before the due time, because with an α equal to 0.5 these methods assume that twice as much time is needed to complete a job. Figure 2 shows that for ARL and RLR the service level drops sharply with increasing α, while the results of EDD and BPRL are less sensitive to the value of α. For all these experiments α is between 0.8 and 1. Note that in case of EDD and BPRL the parameter α only accounts for the uncertainty in processing times while in case of ARL and RLR it also accounts for the inaccuracy of the methods. This explains the difference in sensitiveness.

It is difficult to give the best value of α for a particular service level requirement, but there are some general observations. When the slack of the orders increases, α 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 α decreases. In this case, capacity has to be reserved for the uncertainties. Figure 3 gives an overview of the feasible combinations of utilisation and service level. These values form an efficient frontier for each method.

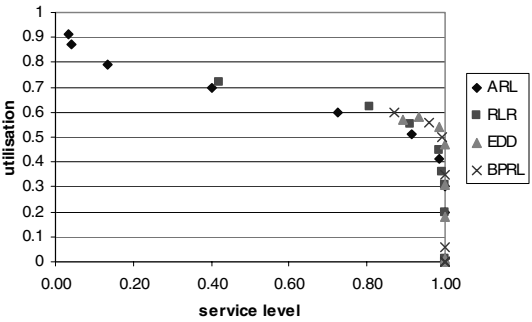

Fig. 3. Relation between utilisation and the service level

Table 2. Average effect of the experimental factors on the utilisation (absolute difference, n.s = not significant, $*p \leq 0.05$ and $**p \leq 0.01$)

OA method	#jobs/order	Processing time	Uncertainty	Workload	Slack
ARL	$0.01^{n.s}$	$-0.01^{n.s.}$	-0.02^{*}	0.02^{*}	0.24^{**}
RLR	-0.04^{**}	-0.02^{**}	-0.03^{**}	0.05^{**}	0.23^{**}
EDD	$0^{n.s.}$	-0.01^{*}	-0.03^{**}	0.06^{**}	0.15^{**}
BPRL	-0.03^{**}	-0.01^{**}	-0.02^{**}	0.07^{**}	0.12^{**}

5.2 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.

The *number of jobs per order* has moderate impact on the utilisation rate; it is not significant for ARL and EDD. The RLR and BPRL method accept less work in case of a high number of jobs per order. The OA methods are only slightly sensitive to the *variance in processing times*, where this effect is not significant for ARL. When we look at the results in more detail it turns out that 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 Fig. 4). 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

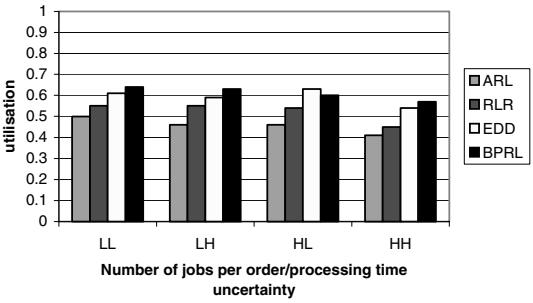

Fig. 4. Relation between "number of jobs per order / processing time uncertainty" (LH=Low/High) and the utilisation rate given a high variance in processing times, high workload and little slack

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.

5.3 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 various 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 various order acceptance methods. Only the pairwise comparison between EDD and BPRL over all experiments does not show a significant difference (p=0,06). All other differences are significant; $p < 0.05$.

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 Fig. 5). The advantages of EDD emerge when the workload is high and there is little slack (last column Table 3). In that case,

Table 3. Average utilisation for the various order acceptance methods

OA method	Average utilisation rate over all experiments	Average utilisation rate for low slack and high demand experiments
ARL	0.56	0.46
RLR	0.62	0.53
EDD	0.64	0.60
BPRL	0.65	0.62

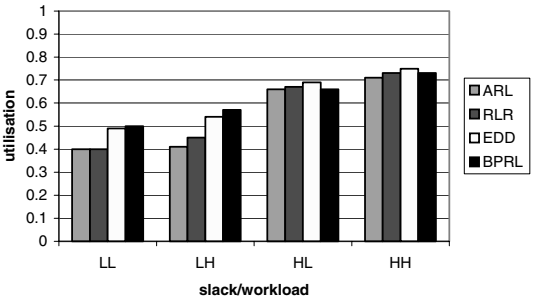

Fig. 5. Relation between "slack/workload" (LH=Low/High) and the utilisation rate given that the other factors are at high level

EDD achieves a utilisation rate of up to 0.14 higher than that of ARL, which is an improvement of 30% (Fig. 5). 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. In Figure 5 it appears that BPRL performs slightly better than EDD in case of little slack, but in case of high slack BPRL shows slightly worse results, possibly because of the restricted computation time. The reason that there is no significant difference might 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.

5.4 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 5 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.

5.5 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

with p=0.03) 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 compared different workload based rules for order acceptance, varying from rules based on aggregate information to a detailed scheduling method. Where most previous work has focused on the deterministic single resource case, we investigated a job shop with stochastic processing times. We found that in cases with little slack and a high workload it is worthwhile to construct a detailed schedule. This is in line with the existing literature (cf. Wester et al., 1992; Ivanescu et al., 2002), in which the scheduling approach for tight settings also shows better results than rules based on aggregate information. In our experiments we find an increase in the utilisation rate of up to 0.14 (30% improvement) can be realized in cases with little slack and high workload, given a required service level of 95%. Constructing a detailed schedule in cases with a lot of slack only leads to a slight increase in utilization rate. Overall, the EDD rule might be preferred when looking at performance, the setting of α, and the computation times.

From the two detailed scheduling methods the Branch&Price approach did not outperform the Earliest Due Date dispatching rule. One reason may be that we gave the Branch&Price method 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 the Branch&Price method, 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 various 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

Akkan C (1997) Finite-capacity scheduling-based planning for revenue-based capacity managementEuropean Journal of Operational Research 100: 170–179

Hans EW (2001) Resource loading by branch-and-price techniques. Ph.D. thesis, University of Twente, The Netherlands

Ivanescu CV, Fransoo JC, Bertrand JWM (2002) Makespan estimation and order acceptance in batch process industries when processing times are uncertain. OR Spectrum 24: 467–495

Lewis HF, Slotnick SA (2002) Multi-period job selection: planning work loads to maximize profit. Computers & Operations Research 29: 1081–1098

Mainegra Hing M, Harten A van, Schuur P (2002) Order acceptance with reinforcement learning. Working paper WP-66, University of Twente, The Netherlands

Raaymakers WHM, Bertrand JWM, Fransoo JC (2000a) The performance of workload rules for order acceptance in batch chemical manufacturing. Journal of Intelligent Manufacturing 11: 217–228

Raaymakers WHM, Bertrand JWM, Fransoo JC (2000b) Using aggregate estimation models for order acceptance in a decentralized production control structure for batch chemical manufacturing. IIE Transactions 32: 989–998

Slotnick SA, Morton TE (1996) Selecting jobs for a heavily loaded shop with lateness penalties. Computers & Operations Research 23: 131–140

Snoek M (2000) Neuro-genetic order acceptance in a job shop setting. Proceedings of the 7th International Conference on Neural Information Processing, Taejon, Korea, pp 815–819

Ten Kate H (1995) Order acceptance and production control. Ph.D. thesis, University of Groningen, The Netherlands

Wang J, Yang J-Q, Lee H (1994) Multi order acceptance decision support in over-demanded job shops: A neural network approach. Mathematical Computated Modeling 19: 1–19

Wester FAW, Wijngaard J, Zijm WHM (1992) Order acceptance strategies in a production-to-order environment with setup times and due dates. International Journal of Production Research 30: 1313–1326

Zijm WHM, (2000) Towards intelligent manufacturing planning and control systems. OR Spectrum 22: 313–345