A scenario based approach for flexible resource loading under uncertainty

G. Wullink A.J.R.M. Gademann E.W. Hans

A. van Harten

May 16, 2003

Abstract

Order acceptance decisions in manufacture-to-order environments are often made based on incomplete or uncertain information. To promise reliable due dates and to manage resource capacity adequately, resource capacity loading is an indispensable supporting tool. We propose a scenario based approach for resource loading under uncertainty that minimises the expected costs. The approach uses an MILP to find a plan that has minimum expected costs over all relevant scenarios. We propose an exact and a heuristic solution approach to solve this MILP. A disadvantage of this approach is that the MILP may become too large to solve in reasonable time. We therefore propose another approach that uses an MILP with a sample of all scenarios. We use the same exact and heuristic methods to solve this MILP.

Computational experiments show that, especially for instances with much slack, solutions obtained with deterministic techniques for a expected scenario can be improved with respect to their expected costs. We also show that for large instances the heuristic outperforms the exact approach given a computation time as a stopping criterion.

Keywords: multi resource loading, stochastic optimisation, modeling uncertainty, scenario planning

1 Introduction

Many manufacture-to-order companies face uncertainties at the order acceptance stage. Orders may vary significantly with respect to routings, material, tool requirements, etc. In spite of the uncertain order characteristics, order accept/reject decisions must be made. It is common practice that companies accept as many orders as they can get, although it is very difficult to estimate the impact on the operational performance of the production system. This may lead to serious overload of resources, which has a negative impact on order completion times. On the other hand, customers require reliable order due dates as part of the service mix offered by the company during order negotiation. Being able to quote reliable due dates is a competitive advantage during order acquisition. Therefore, at the order negotiation and acceptance stage, adequate planning methods that assess the consequences of order acceptance decisions for the production system are essential. Planning methods that are designed for the operational planning level are not suitable for this purpose. Operational planning / scheduling methods require detailed information about, e.g. processing times, precedence relations between operations, and resource requirements. This detailed information is often incomplete or not available during the order acceptance stage. This incorporates a lot of uncertainty. Nevertheless, at the tactical planning stage, there is more capacity flexibility (e.g. working in overtime) than at the operational planning level. Tactical planning requires methods that operate at a higher planning level, the tactical planning level, that use more aggregate data, and that can utilise capacity flexibility.

Capacity flexibility is an important characteristic of the tactical planning level. Ideally, methods at this planning stage should utilise this flexibility to support a planner in making a trade-off between the expected delivery performance and the expected costs of using nonregular capacity. In the remainder of this paper we will refer to such methods as Flexible Resource Loading (FRL) methods. Several exact and approximate FRL methods have been proposed, which use deterministic input data and assume deterministic behaviour of the production system (see e.g. Hans, 2001). These deterministic approaches generally either focus on optimisation of time criteria (tardiness, lateness, makespan, etc.), or money criteria (costs, profit), since these are easily quantifiable. Unfortunately these methods hardly deal with uncertainty. As argued before, data may be highly uncertain, especially in the order acceptance stage. Deterministic planning may result in unreliable and nervous plans.

In this paper we propose a model that minimises expected costs of Flexible Resource Loading problems under Uncertainty (FRLU). This model is a generalisation of the FRL model. We use scenarios to model uncertainties that are typical for the tactical planning level. We propose an exact and a heuristic algorithm to solve this FRLU model over all scenarios, or over a sample of all scenarios.

The paper is organised as follows. In section 2 we position our research, and discuss related work about planning and optimisation under uncertainty. In section 3 we describe the FRLU problem. In section 4 we propose a scenario based MILP model. In section 5 we discuss two solution approaches for the scenario based model. In section 6 we present computational results. Finally, we draw conclusions in section 7, and give directions for further research.

2 Research positioning and literature

We position our research and discuss related work about planning under uncertainty using a hierarchical planning framework with 3 levels, which are generally distinguished in the literature (Bitran and Tirupati, 1993):

- Strategical planning
- Tactical planning
- Operational planning

To position FRLU at the tactical planning level we discuss the major differences with the other planning levels in this section.

Strategical planning involves long-term decisions made at the company management level. It addresses problems like facility location planning, workforce planning and product mix planning. Strategical planning problems are often solved using LP techniques, see Hopp and Spearman (1996). They typically use demand forecasts as planning input. These forecasts are a considerable source of uncertainty. An example of a strategical planning technique that accounts for these uncertainties is, for instance, the multi stage LP technique proposed by Eppen, Martin and L. (1989). Escudero et al. (1993) propose scenario based LP techniques for production planning problems with unknown product demands. The main characteristic of strategic planning is that it does not assume any information about customer orders, but instead uses demand forecasts that yield aggregate data about the future demands.

Tactical planning is concerned with allocating available resources to arriving/accepted customer orders as efficiently as possible and determining reliable due dates for these orders. At this medium term planning stage, generally only rough order data is available, such as estimated processing times of jobs and some *a priori* precedence relations. Especially in a manufacture-to-order environment uncertainty at the order acceptance stage is still considerably large. The FRLU problem concerns assigning jobs to resource capacity buckets in time periods. Regular capacity levels are fixed, and nonregular capacity (outsourcing, working overtime and hiring additional personnel) provides capacity flexibility. Using this nonregular capacity invokes additional costs.

Literature on deterministic tactical planning / FRL is rather scarce. The aforementioned LP based strategic planning methods are not suitable for FRL, because they do not account for precedence relations between jobs. Scheduling techniques are not suitable for the tactical planning level either. While they do account for precedence relations, too much detailed data is required, which is generally not readily available at the tactical planning stage. Moreover, they generally can not utilise capacity flexibility. Especially in manufacture-to-order environments detailed order information becomes available during detailed design and process planning, which is generally performed *after* order acceptance.

Hans (2001) proposes a branch-and-price approach to solve the FRL problem. Gademann and Schutten (2001) propose an LP based heuristic, and De Boer (1998) propose an adaptive search approach for the tactical planning problem. All these methods use deterministic input data.

At the tactical planning stage, the lack of detailed information about jobs forces a planner to deal with uncertainties to be able to quote, for instance, reliable due dates. Models that anticipate and deal with uncertainty on the tactical level have not been developed yet.

Operational planning concerns the short term scheduling of operations on resources. The scheduling objective is generally time related. At this planning stage resource capacity is generally considered as fixed, which means that there is hardly flexibility to absorb disruptions. The only possibilities are perhaps to plan slack, or to reschedule. Consequently, uncertainties may result in nervousness of the schedules created with deterministic input data. Hence, dealing with uncertainty in scheduling has gained the interest of researchers in the past decades. Herroelen and Leus (2002) distinguish five main approaches of scheduling under uncertainty: *reactive scheduling*, *stochastic project scheduling, stochastic project networks, fuzzy project scheduling and proactive/robust scheduling.* The first and the second approach are online scheduling techniques that respectively reoptimise the schedule after a disturbance, or develop an optimal policy (e.g. Moehring, 2000) for situations of disturbances. In both approaches decisions are made whenever information about disturbances becomes available. Stochastic project networks deal with projects with a stochastic evolution structure of the activity network. This means that it is unknown in advance which activities are going be executed, and for how many times. Because of the high computational requirements of the methods, analysis of stochastic project networks is often performed by simulation. For more details about stochastic project networks we refer to Neumann and Zimmermann (1979). Fuzzy project scheduling is based on the assumption that activity durations rely on human estimations. Hapke and Slowinski (1996) propose a priority based scheduling heuristic using fuzzy number theory . Finally, Herroelen and Leus (2002) distinguish proactive/robust scheduling as an approach for scheduling under uncertainty. The main goal of proactive/robust scheduling is to generate a robust schedule. Herroelen and Leus propose a pairwise float model, which is a mathematical programming technique to develop stable (robust) baseline schedules. They aim to minimise the difference between the start times of the realisation and the initial schedule. Other proactive scheduling techniques are often based on the insertion of idle times, slack or buffers. An example of such an approach is the critical chain approach proposed by Goldratt (1997).

Summarising, we observe that research on production planning under uncertainty mainly focusses on the strategical or the operational level. As far as we know, no stochastic optimisation method for the tactical planning level have been proposed in the literature. Tactical planning methods that have been found in literature assume deterministic input data, which we believe is a highly questionable assumption, especially in environments with much uncertainty.

3 Problem description

Flexible Resource Loading under Uncertainty (FRLU) addresses the problem of assigning a set of jobs, generated by a list of customer orders considered for acceptance, to a number of resource groups. Capacity levels are flexible because of the possibility to plan nonregular capacity against additional costs. Problem parameters like processing times or capacity levels can be uncertain. The objective is to plan the orders in such a way that available resources are used as efficiently as possible, customer order due dates are met, and the plans are insensitive to uncertainty.

There are n orders that consist of jobs or work packages, which must be processed for a given processing time on one or more resource groups. We consider m independent resources. Each order j consists of n_j jobs (index b) with generic precedence relations. An order can start from its release date (r_j) and must be completed before its due date (d_j). Job (b, j) is performed on resource group μ_{bj} for a given positive processing time p_{bj}. The parameter ω_{bj} indicates the minimum duration for each job (b, j) in periods.

The basic idea of our scenario based approach is as follows. At the tactical planning stage, many order characteristics and resource characteristics, such as processing times, precedence relations, occurrence of a job, resource capacity, or material availability, etc., can be uncertain. In this paper we deal with uncertainty in processing times. To model this, we assume that a limited number of processing times per job may actually occur, which we call *modes*. We define a *scenario* of a problem instance as a case in which each job occurs in a specific mode. We assume that we have no *a priori* information about which mode a job will occur in until we start the job. At that moment we know the mode for that job. Based on this limited knowledge, we want to present a good plan with respect to the expected costs over all scenarios. The plan must be causal, i.e., it can only use information about what scenarios may occur, but it is unknown what scenario will occur. Valls et al. (1998) refer to this causality condition as the nonanticipativity constraint of multiperiod stochastic problems. We define a scenario independent plan as follows: for each job we determine the fraction of the job that will be processed in a time period. Since the processing time is known at the start of the job, we can execute such a plan independent of the scenario. To find such a plan we present an approach to solve to *FRLU* problem by minimising the expected costs over all scenarios. The idea of this approach is that uncertain jobs will be planned in buckets with the largest amount of excess capacity, if multiple scenarios are taken into account. We illustrate this by the following example.

Example

Consider the following problem instance with one resource group and two orders. Each order has one job with a minimum duration. Job $(1, 1)$ is certain, and only occurs in one processing mode. Job $(2, 1)$ is uncertain, and can occur in three processing modes. Each mode has equal probability $1/3$. The resource groups have regular and nonregular capacity. Table 1 and table 2 present the instance data.

Order	Job	Resource	*min.* dur.	Proc. times			Probabilities		
1	1	1	2	–	60	–	–	1	–
2	1	1	1	5	10	15	1/3	1/3	1/3

Table 1: Order data

Resource	Reg. cap		Nonreg. cap	
i	$t = 1$	$t = 2$	$t = 1$	$t = 2$
1	40	40	10	10

Table 2: Resource data

Solving the problem using one scenario with processing time 10 for job $(2, 1)$ may yield the cost optimal solution displayed in figure 1 (note that alternative optimal solutions exist).

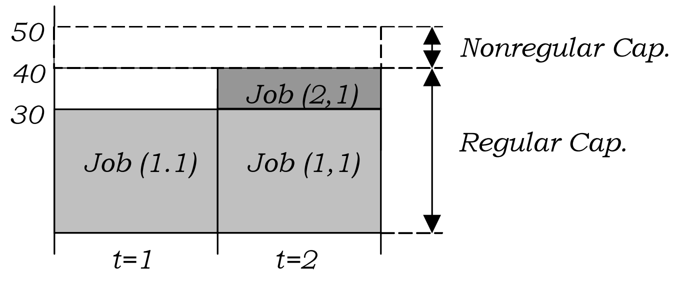

Figure 1: A solution for the expected scenario

This plan gives the following fractions: $\frac{1}{2}$ of job $(1, 1)$ is executed in period 1 and $\frac{1}{2}$ is executed in period 2. Job $(2, 1)$ is completely executed in period 2. If the processing time of job $(2, 1)$ happens to occur with processing time 15 this would require $\left(1 * 15 + \frac{1}{2} * 60 - 40 =\right)$ 5 hours of nonregular capacity. The expected costs of this solution are $(\frac{1}{3} * 0) + (\frac{1}{3} * 0) + (\frac{1}{3} * 5) = 1\frac{2}{3}$.

Preferably, we would have generated a cost equal solution for the using gives a plan that executes $\frac{2}{3}$ of job $(1, 1)$ in period 1 and $\frac{1}{3}$ in period 2 (see figure 2).

This solution does not require costs for nonregular capacity if job $(2, 1)$ occurs with processing time 15. The expected costs of this plan over all scenarios are $(\frac{1}{3} * 0) + (\frac{1}{3} * 0) + (\frac{1}{3} * 0) = 0$.

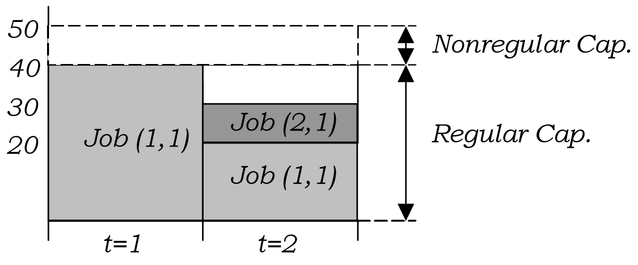

Figure 2: The preferred solution

4 Scenario based model

To formulate the model we introduce the following concepts.

Order plan and loading schedule

The *order plan* for order j specifies the time periods in which job (b, j) is allowed to be processed. An order plan π for an order j is represented by a vector a^{π}_{j}, which has binary elements a^{π}_{bjt} that specify whether job (b, j) is allowed to be executed in period t. We only consider order plans that are feasible with respect to the precedence relations, the order release date, and the minimum duration restrictions for jobs. For each order j the variable X^{π}_{j} indicates whether order plan π is selected. We implicitly generate order plans by a column generation technique (we come back to this in section 5.1). Working with order plans has tremendous advantages in terms of the size of an MILP model and the number of required integer variables.

The *loading schedule* of order j specifies the fraction Y_{bjt} of job (b, j) that is performed in period t. A loading schedule is represented by a vector (Y_{1j1},..., Y_{1jT},..., Y_{nj1},..., Y_{njT}). Since a loading schedule matches a selected order plan, the loading schedule is always feasible with respect to the precedence relations. By multiplying a loading schedule by the corresponding processing times p_{bj} we obtain the realisation $Y_{bjt} * p_{bj}$ in period t.

Tardiness

We define the *completion time* of an order according to order plan π as the last time period t in which processing of order j is allowed. Accordingly, we define the *allowed lateness* of an order as the difference between the allowed completion time CT_j^π and the due date d_j of order j measured in periods. The allowed completion time of order j is the last period in which the order is allowed to be processed in the order plan, i.e. $CT_j^\pi = max\left\{t|a_{bjt}^\pi = 1\right\}$. The *allowed tardiness* ∂_j^π of an order is zero if the allowed lateness is nonpositive, and equal to the allowed lateness if it is positive, or formally: $\partial_j^\pi = \max\left\{0, CT_j^\pi - d_j\right\}$. The total allowed tardiness is penalised in the objective function with a cost parameter θ. Notice that we penalise the allowed tardiness instead of the actual tardiness. A branch-and-price procedure (see section 5.1) that we use to solve this problem, however, always leads to a solution where the allowed tardiness matches the allowed completion time.

Scenarios

We model uncertainty as follows. For all u_j uncertain jobs we consider three processing modes ($m \in \{min, exp, max\}$) by drawing processing times from a uniform distribution. We use p_{bj}^m to indicate processing time of job (b, j) in mode m. The corresponding probability for each mode is q_{bj}^m. The modes constitute a total of $l = \Pi_j (3)^{u_j}$ scenarios. The mode in which an uncertain job (b, j) occurs in scenario σ is indicated by z_{bj}^σ. The scenario probability q^σ is then given by: $\Pi_{b,j,m|m=z_{bj}^\sigma} q_{bj}^m$. In the remainder we indicate the processing time of job (b, j) in scenario σ by p_{bj}^σ. Using scenario independent loading schedules automatically results in satisfying the causality condition.

4.1 Notation

We use the following notation:

Indices

t	period $(t = 0, ..., T)$
σ	scenarios $(\sigma = 1, ..., l)$
j	order $(n = 1, ..., n)$
b	job of order $j(b = 1, ..., n_j)$
i	resources $(i = 1, ..., m)$

Scenario dependent parameters

p_{bj}^{σ}	the processing time of job (b,j) in scenario σ
p_{bj}^{m}	the processing time of job (b,j) in mode m
q^{σ}	probability of scenario σ
q_{bj}^{m}	probability that job (b,j) to occurs in mode m
z_{bj}^{σ}	the mode in which job (b,j) occurs in scenario σ
s_{it}^{σ}	outsourcing capacity in period t in scenario σ
υ_{bji}^{σ}	fraction of job (b,j) that is performed on resource i in scenario σ
mc_{it}^{σ}	total regular capacity of resource i in period t in scenario σ
s_{it}^{σ}	outsourcing capacity on resource i in period t in scenario σ

Scenario independent parameters

u_j	number of uncertain jobs of order j
$\Pi\,(\Pi_j)$	the set of all feasible order plans (order plans for order j)
a_j^{π}	π-th order plan for order j
ζ	costs of one unit subcontracting
r_j, d_j	release date, due date for order j
θ	tardiness penalty
∂_j^{π}	allowed tardiness of order j in order plan a_j^{π}
ω_{bj}	minimum duration of job (b,j)

Decision variables

S_{it}^{σ}	outsourced production hours for resource i in period t in scenario σ
X_j^{π}	0-1 variable that assumes 1 when order plan $a_{j\pi}$ is selected for order j
Y_{bjt}	fraction of job (b,j) executed in period t

4.2 Model

The objective of the model is to minimise the expected costs over all scenarios:

$$z_{ILP}^{*} = \min \sum_{\sigma=1}^{l} q^{\sigma} \left(\zeta \sum_{t=\mathbf{0}}^{T} \sum_{i=1}^{m} S_{it}^{\sigma} + \theta \sum_{j=1}^{n} \sum_{\pi \in \mathbf{\Pi}_j} \partial_j^{\pi} X_j^{\pi} \right) \tag{1}$$

Subject to:

$$\sum_{\pi \in \mathbf{\Pi}_j} X_j^{\pi} = 1 \ (\forall j) \tag{2}$$

$$Y_{bjt} - \frac{\sum_{\pi \in \mathbf{\Pi}_j} a^{\pi}_{bjt} X^{\pi}_j}{\omega_{bj}} \leq 0 \ (\forall b, j, t) \tag{3}$$

$$\sum_{t=r_j}^{T} Y_{bjt} = 1 \ (\forall b, j) \tag{4}$$

$$\sum_{j=1}^{n} \sum_{b=1}^{n_j} p^{\sigma}_{bj} v^{\sigma}_{bji} Y_{bjt} \leq mc^{\sigma}_{it} + S^{\sigma}_{it} \ (\forall i, t, \sigma) \tag{5}$$

$$\sum_{i=1}^{m} S^{\sigma}_{it} \leq s^{\sigma}_{t} \ (\forall t, \sigma) \tag{6}$$

$$X^{\pi}_j \in \{0, 1\} \ (\forall j, \pi \in \Pi_j \subset \Pi) \tag{7}$$

$$\text{all variables} \geq 0 \tag{8}$$

Constraints (2) and (7) stipulate that exactly one order plan is selected for each order j. Constraints (3) stipulate that for each order j, the loading schedule Y^{π}_{bjt} is consistent with the selected order plan a^{π}_{j}. It also stipulates that if job (b, j) has a minimum duration of w_{bj} periods, no more than $\frac{1}{\omega_{bj}}$-part of the job can be done per period. Constraints (4) stipulate that all is done. Constraints (5) and (6) are the resource capacity and subcontracting capacity constraints for each scenario σ. An LP relaxation of this model, which we shall use later on, is obtained by relaxing constraints (7) to $X^{\pi}_j \leq 1 \ (\forall j, \pi \in \Pi_j \subset \Pi)$.

5 Solution approaches

In this section we discuss two solution approaches to solve the FRLU problem. In section 5.1 we discuss an exact branch-and-price algorithm. In section 5.2, we discuss a heuristic approach to solve the FRLU problem. As described in section 4.2, the number of scenarios is $l = \Pi_j \, (3)^{u_j}$ where u_j is the number of uncertain jobs. To deal with a large number of scenarios we also propose method that takes a sample of all scenarios in section 5.3.

5.1 Branch-and-Price

Solving an LP relaxation of the MILP model described in section 4.2 would require all order plans. The total number of feasible order plans is exponentially large. Therefore, we start with a restricted LP model (RLP) in which each order has at least one feasible order plan. We generate these order plans with a heuristic that is based on the Earliest Due Date (EDD) priority rule.

The FRLU model is a generalisation of the FRL model, proposed by Hans (2001). The main difference is in the added constraints to deal with scenarios. The branch-and-price algorithm proceeds roughly as follows: a pricing algorithm is applied to update the RLP. If order plans with negative reduced costs exist, they are added to the RLP. After this, the new RLP is reoptimised. This process is repeated until no order plans with negative reduced costs exist. The column generation algorithm then terminates. The obtained RLP solution at this stage is the optimal solution for the LP relaxation. If this solution happens to be integral, it is the optimal solution for the MILP model of the FLRU problem. Otherwise, we first apply several rounding heuristics to obtain good upper bounds and then a branch-and-price algorithm to obtain the optimal solution for the MILP problem. In a fractional solution more than one order plan may be selected fractionally. As a result, in the combined order plan precedence constraints may be violated. The branch-and-price algorithm branches on these violated precedence constraints. If no precedence constraint in a RLP solution in a node is violated we have found a solution for the MILP problem. By branching through all nodes we obtain the optimal solution. If no optimal solution is obtained within 30 minutes, we truncate the algorithm. We select the best solution that has been found until then.

5.2 LP based improvement heuristic

We use an improvement heuristic proposed by Gademann and Schutten (2001) to find approximate solutions for FRLU problems. As for the exact branch-and-price approach , this heuristic was also proposed for FRL problems. However, we may use it to solve FRLU problems with hardly any modifications.

The heuristic starts from a feasible solution from which it forms an RLP model with one order plan per order. This feasible solution is constructed with an earliest due date heuristic (EDD). In each iteration the algorithm evaluates the expected yield (based on the shadow prices) of all possible changes in the order plans. These changes are obtained by changing the start or completion time of a job in the order plans with one period. If necessary, the start or completion times of successors or predecessors of this job are updated. There are four possible types of changes resulting from decreasing or increasing the start time or the completion time of a job with one period. We discard the changes that lead to an infeasible solution. We order all changes according to the improvement in objective function. We accept the change that yields the best expected improvement, and then reoptimise the RLP. After optimisation of the RLP with the new order plan we evaluate all possible changes of the new order plan. We repeat this procedure until no change exists that leads to an expected improvement. Note that this procedure uses *one* order plan per order in the RLP at all times. As a result, reoptimising the RLP in each iteration requires relatively little computation time and each iteration yields a feasible solution. However, for some instances it may be necessary to limit the computation time. We therefore truncate the algorithm after 30 minutes.

5.3 Sampling

We proposed two algorithms to solve the FRLU problem. Observe that including all possible scenarios in the FRLU model may lead to a model that is too large to solve within reasonable computation time. We therefore propose a scenario sampling approach. This sampling approach takes a sample of size x from all scenarios.

We refer to the approach with the branch-and-price with all l scenarios as Monolithic Branch-and-Price Loading (*MBPL)*. We refer to the approach with the branch-and-price with a sample of size x form all scenarios as Sample

Branch-and-Price Loading (*SBPL*(x)).

We refer to the approach which incorporates all scenarios in combination with the heuristic solution algorithm as Monolithic Heuristic Loading (*MHL)*. We refer to the approach with a scenario sample of size x in combination with the heuristic solution approach as Sample Heuristic Loading (*SHL*(x)).

6 Computational experiments

We performed several experiments to test the proposed approaches. In section 4.2 we have formulated the model for the FRLU problem with scenario dependent processing times, resource requirements, resource capacity and outsourcing capacity. For the computational experiments we consider scenario dependent processing times. Hence, ν^{σ}_{bji}, mc^{σ}_{it} and s^{σ}_{it} are in the *exp* mode. Before presenting the computational results we describe the problem instance generation procedure.

We implemented and tested all methods in the Borland Delphi 7.0 programming language on a Pentium III 600 Mhz personal computer. The application interfaces with the ILOG CPLEX 8.0 callable library, which we use to optimise the linear programming models.

6.1 Test Instances

We use the network generation procedure proposed by Kolisch, Sprecher and Drexl (1995) to generate the order release and due dates, and job precedence relations in our test instances. In this procedure a network is characterised by the three parameters n (the number of jobs), K (the number of resources) and ϕ (average slack).

The network generation procedure is as follows. Given is a set of n jobs. The first step is to determine the start jobs (jobs that have no predecessor) and the finish jobs (jobs that have no successor). In step 2, one predecessor is randomly assigned to each non start job. In step 3, one successor is assigned randomly to each non finish job. Precedence relations are only added in step 2 and 3 if they are not redundant. A precedence relation between job i and j is redundant if and only if some path $P(i,j)$ exists. In step 4, nonredundant arcs are added until the desired average number of predecessors per node is reached. The desired average number of predecessors per node (i.e. the network complexity) is 2.

All jobs require at least 3 and at most 5 resources. Resource capacity (mc_{it}) and processing times $\left(p_{bj}^{\sigma}\right)$ are chosen such that the expected workload is 70% of the total capacity. The average slack is defined as:

$$\phi = \frac{\sum_j \sum_{b=1}^{n_j} (d_j - \omega_{bj} - r_{bj} + 1)}{\sum_j n_j} \tag{9}$$

where $(d_j - \omega_{bj} - r_{bj} + 1)$ is the slack of job (b,j). The minimum duration ω_{bj} of job (b,j) is an integer number drawn randomly from $\{1, ..., 5\}$. We now repeatedly draw d_j and r_{bj} (the release date of job (b,j)) until we satisfy the average slack condition in equation (9). The regular capacity for each resource mc_{it} is randomly drawn from $[0, 20]$. Each job requires a random number of resources drawn from $\{1, ..., 5\}$. The processing times p_{bj} are now randomly drawn from the interval $\left[1, 0.8\left(2\frac{\sum_{i=1}^{m}\sum_{t=1}^{T} mc_{it}}{n_j \frac{\min\{m,5\}+1}{2}} - 1\right)\right]$. If a resource is not selected to process job (b,j), its processing time p_{bj} is set to 0.

The number of uncertain jobs $u_j = 4$. We draw these uncertain jobs randomly from all n_j jobs. By applying $l = \Pi_j (3)^{u_j}$, the number of scenarios is 81. The processing modes are determined as follows: $p_{bj}^{min} = \alpha * p_{bj}$, $p_{bj}^{max} = \beta * p_{bj}$, and $p_{bj}^{exp} = \frac{\alpha+\beta}{2} * p_{bj}$, α is randomly drawn form $[0.5, 1]$ and β is randomly drawn from $[1, 2]$. In our experiments with *MBPL*, *SBPL*(x), *MHL*, and *SHL*(x) the probabilities q_{bj}^{m} are 1/3 for each mode.

We generate 90 instances with 10 jobs $(n_j = 10)$. This set consists of 3 sets of 30 instances. The average slack in the three sets are 0.2, 0.5, and 1 respectively. For the sampling methods we use a sample of 10 scenarios that are uniformly drawn from all scenarios (*SBPL*$(rand10)$ and *SHL*$(rand10)$).

6.2 Test results

In this section we present the computational results. In table 3 we present the results for the complete set of 90 instances. The results in this table are all relative to the performance of the (deterministic) *FRL* method. The *FRL* method determines a solution based on the *exp* scenario. This solution is then evaluated over all scenarios, which yields the costs of the solution in each scenario. From this we can calculate the expected costs (of the '*exp* solution', over all scenarios) and the standard deviation of the costs over all scenarios. The column 'exp costs' in table 3 represents the relative average improvement of the expected costs (over all scenarios) for the other methods, with respect to the expected costs of the *FRL* solution (the range of the costs over all instances is given between square brackets). The column 'stddev' does the same for the standard deviation in the costs.

The results in the '*min* scen' column in table 3 give the relative average improvement of the costs of the solutions of the other methods in the *min* scenario, with respect to the costs of the *FRL* solution in the *min* scenario (a positive number is a decrease of the costs). The column '*max* scen' does the same for the *max* scenario. The column '# trunc instances' shows the number of instances that were truncated after 30 minutes. The column 'Average comp time (in sec)' shows the average computation time in seconds.

	Average improvement of (in %):					Average
	exp costs		*min*	*max*	# trunc	comp time
	[range]	stddev	scen	scen	instances	(in sec)
FRL	–	–	–	–	3	102
MBPL	5 $[-71.2, 49.3]$	0.4	8.8	4.6	62	1525
MHL	10.5 $[-7.3, 57.3]$	2.2	16.2	8.9	23	869
SBPL($rand10$)	9.1 $[-20.1, 57.8]$	-0.1	15.0	7.0	21	607
SHL($rand10$)	11.2 $[-7.3, 57.3]$	2.1	16.7	9.3	0	24

Table 3: Results of the complete batch

MBPL yields smaller improvements than *MHL* and the sampling approaches *SBPL*($rand10$) and *SHL*($rand10$). The main explanation for this is that *MBPL* only solves 28 out of the 90 instances to optimality. We have performed experiments in which we gave *MBPL* 60 minutes of computation time (instead of 30 minutes), but this did not yield much improvement. The range of the solutions, especially for the exact methods, is considerably wide. This can be also be explained by the fact that for some problem instances the truncated exact algorithm yields a far from optimal solution.

In Table 4 we compare the results for all instances that could be solved to optimality. We compare the results of the instances that were solved to optimality by the exact approaches (*MBPL*) with the results for the same instances solved by *SBPL*($rand10$), *MHL,* or *SHL*($rand10$). Note that we only consider 28 of the 90 instances.

	Average improvement of (in %):				Average
	exp costs		*min*	*max*	comp time
	[range]	stddev	scen	scen	(in sec)
MBPL	6.9 $[0.0, 43.4]$	2.7	9.1	6.4	640
MHL	6.8 $[-0.2, 43.0]$	2.5	9.0	6.3	283
SBPL($rand10$)	6.4 $[0.0, 36.9]$	1.3	8.6	5.5	13
SHL($rand10$)	6.6 $[-0.3, 41.2]$	2.1	8.6.0	5.9	7

Table 4: Results for the instances that could be solved to optimality

Here we see that the differences between the heuristic and the exact method are only small. The additional computational effort required for the exact solution approach appears to yield little improvement. The solutions for the problem instances for the monolithic approaches (*MBPL* and *MHL*) range from 0% to 43% improvement. The left side of the ranges of 0% can be explained by instances for which no lower expected costs could be found using a scenario approach.

Note that the number of solutions to an instance increases with the average slack that the orders in the instance have. Intuitively, these cases with more slack are easier to solve. However, since the solution space also increases with the slack, *MBPL* will require more computation time to find an optimal solution.

To analyse the impact of the average slack we performed the experiments for the three sets with $\phi = 0.2$, $\phi = 0.5$, and $\phi = 1$ separately. Tables 5, 6, and 7 show the results for these sets.

	Average improvement of (in %): exp costs [range]	stddev	*min* scen	*max* scen	# trunc instances	Average comp time (in sec)
FRL	–	–	–	–	0	2
MBPL	5.0 $[0.1, 34.3]$	-0.2	8.1	3.8	3	751
MHL	5.0 $[-0.2, 34.3]$	-0.3	8.1	3.8	0	335
SBPL($rand$10)	4.8 $[0.1, 34.3]$	-0.7	7.7	3.5	0	18
SHL($rand$10)	4.9 $[-0.3, 34.3]$	-0.5	7.6	3.6	0	8

Table 5: Results of the batch with $\phi = 0.2$

	Average improvement of (in %): exp costs [range]	stddev	*min* scen	*max* scen	# trunc instances	Average comp time (in sec)
FRL	–	–	–	–	0	65
MBPL	6.4 $[-17.6, 47.3]$	0.7	9.3	6.1	29	1766
MHL	11.0 $[-1.8, 50.4]$	2.7	15.1	9.8	5	901
SBPL($rand$10)	10.7 $[-4.1, 53.7]$	2.2	14.5	8.8	4	585
SHL($rand$10)	10.8 $[-3.8, 50.2]$	2.2	14.5	9.1	0	24

Table 6: Results of the batch with $\phi = 0.5$

The average slack has a negative impact on the solution performance of the exact approaches. If the slack is large the instance are more difficult to solve. The explanation for this is that more average slack increases the solution space of the problem. Hence, *MBPL* results in small improvements for the problem instances with an average slack of 1, because no instances could be solved to optimality. Hence, the heuristic approaches *SHL*($rand$10) and *MHL* yield the best results of 15.6% and 18.1% respectively (see table 5, table 6 and table 7).

Observe that the improvement heuristic in combination with a sample yields the best result for the expected costs for almost all experiments. To investigate the impact of the way of selecting scenarios on the result of the experiments we now compare the results of three different scenario selection approaches in combination with the improvement heuristic. The first approach is to select the scenarios randomly, like for the previous experiments. We call this approach *SHL*($rand$10), like in the previous experiments. For the second sampling approach we use a sample that consists of the *exp* scenario, the *max* scenario, all four scenarios with one of the four uncertain jobs in the *max* mode, and all four scenarios with one of the four uncertain jobs in the *min* mode. We refer to

	Average improvement of (in %):					Average
	exp costs		*min*	*max*	# trunc	comp time
	[range]	stddev	scen	scen	instances	(in sec)
FRL	–	–	–	–	3	238
MBPL	3.6 $[-71.2, 49.3]$	-1.7	9.0	3.8	30	1800
MHL	15.6 $[-7.3, 57.3]$	4.1	25.2	13.0	18	1365
SBPL($rand10$)	12.0 $[-17.5, 57.8]$	-1.7	22.7	8.6	17	1219
SHL($rand10$)	18.1 $[-7.3, 57.3]$	4.5	27.8	15.0	0	41

Table 7: Results of the batch with $\phi = 1$

this sampling approach as *SHL*($Sel10$). Finally, we test a sampling approach which uses a sample consisting of the *min* scenario (i.e., all jobs in the *min* mode), the *exp* scenario (i.e., all jobs in the *exp* mode), and the *max* scenario (i.e., all jobs in the *max* mode). We call this approach *SHL*($MEM3$). Finally, we test an approach which only uses two scenarios as input for the model. For this approach (SHL($EM2$)) we use the *max* scenario and the *exp* scenario.

	Average improvement of (in %):					Average
	exp costs		*min*	*max*	# trunc	comp time
	[range]	stddev	scen	scen	instances	(in sec)
FRL	–	–	–	–	3	106
SHL($rand10$)	11.2 $[-7.3, 57.3]$	2.1	16.7	9.3	0	25
SHL($Sel10$)	10.9 $[-7.6, 57.2]$	3.1	15.6	9.2	0	25
SHL($MEM3$)	11.0 $[-7.3, 57.8]$	2.7	16.6	9.4	0	5
SHL($EM2$)	10.2 $[-8.3, 57.4]$	6.9	11.6	9.7	0	4

Table 8: Results for various samples of scenarios

Table 8 shows that using a controlled sample (*SHL*($Sel10$)) does not improve the average expected costs. From table 8 we can also conclude that a sample of three scenarios (*SHL*($MEM3$)) yields almost the same improvement as a sample of 10 scenarios. Moreover, a sample of two scenarios (*SHL*($EM2$)) still yields considerably better results than the *FRL* approach. Finally, we conclude that the approach with three scenarios (SHL($MEM3$)) has a good performance considering the improvement of the expected costs and the computation time.

7 Conclusions and further research

We have presented a scenario based model for Flexible Resource Loading under Uncertainty (*FRLU*), which aims to obtain plans with minimum expected costs. We discuss several exact and approximation algorithms to solve this model. Computational experiments show that improvement of the expected costs can be obtained by using the *FRLU* approach, as opposed to using a deterministic approach. We have shown that the exact approaches often cannot solve instances to optimality within reasonable time, even when only a sample of the scenarios is considered. An LP based improvement heuristic in combination with scenario sampling is to be the most promising approach. Moreover, a small sample (of for instance three or two scenarios) appears to be enough to achieve a considerable improvement with respect to the expected costs.

The model proposed in this paper can account for several uncertainties in a manufacturing environment. However, uncertainties like uncertain release or due dates can not be accounted for in the model. In future research we will look for ways that allow us to account for these uncertainties. We also plan to examine an approach which generates a number of alternative plans with (almost) equal costs. We will develop a robustness indicator to measure the robustness of a plan. With such an indicator, the alternative plans can be compared with respect to their robustness.

References

Bitran, G.R. and D. Tirupati (1993). *Hierarchical Production Planning*, Volume 4 of *Logistics of Production and Inventory, Handbooks in Operations Research and Management Science.* North-Holland, Amsterdam.

De Boer, R. (1998). *Resource-Constrained Multi-Project Management - A Hierarchical Decision Support System.* Ph. D. thesis, University Of Twente.

Eppen, G.D., R.K. Martin, and Schrage L. (1989). A scenario approach to capacity planning. *Operations Research 37*, 517–527.

Escudero, L.F., P.V. Kamesam, A.J. King, and R.J-B. Wes (1993). Production planning via scenario modelling. *Annals of Operations Research 43*, 311–335.

Gademann, A.J.R.M. and J.M.J. Schutten (2001). Linear programming based heuristics for multi-project capacity planning. *Working paper TBK01W-004 OMST-002, University of Twente, Enschede.*

Goldratt, E.M. (1997). *Critical Chain.* The North River Press Publishing Corporation, Great Barrington.

Hans, E.W. (2001). *Resource Loading by Branch-and-Price Techniques.* Ph. D. thesis, University of Twente.

Hapke, M. and R. Slowinski (1996). Fuzzy priority rules for project scheduling. *Fuzzy sets and systems 83*, 291–299.

Herroelen, W. and R. Leus (2002). Project scheduling under uncertainty; survey and research potentials. *To appear in: European Journal of Operation Research.*

Hopp, W.J. and M.L. Spearman (1996). *Factory Physics - Foundations of Manufacturing Management.* IRWIN, Boston.

Kolisch, R., A. Sprecher, and A. Drexl (1995). Characterization and generation of a general class of resource constrained project scheduling problems. *Management science 10*, 1693–1703.

Moehring, R.H. (2000). Scheduling under uncertainty: Bounding the makespan distribution. *Working paper 700.*

Neumann, K. and J. Zimmermann (1979). *GERT-Networks and the Time-Oriented Evaluation of Projects*, Volume 172. Berlin: Springer Verlag.

Valls, V., M. Laguna, P. Lino, A. Perez, and S. Quintanilla (1998). *Project Scheduling with Stochastic Activity Interruptions.* 14. Kluwer Academic Publishers.