European Journal of Operational Research 213 (2011) 166–179

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Stochastics and Statistics

A multi-stage stochastic programming approach in master production scheduling

Ersin Körpeoğlu [a], Hande Yaman [b], M. Selim Aktürk [b,*]

[a] Tepper School of Business, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, USA
[b] Department of Industrial Engineering, Bilkent University, 06800 Ankara, Turkey

ARTICLE INFO

Article history:
Received 26 April 2010
Accepted 27 February 2011
Available online 4 March 2011

Keywords:
Stochastic programming
Master production scheduling
Flexible manufacturing
Controllable processing times

ABSTRACT

Master Production Schedules (MPS) are widely used in industry, especially within Enterprise Resource Planning (ERP) software. The classical approach for generating MPS assumes infinite capacity, fixed processing times, and a single scenario for demand forecasts. In this paper, we question these assumptions and consider a problem with finite capacity, controllable processing times, and several demand scenarios instead of just one. We use a multi-stage stochastic programming approach in order to come up with the maximum expected profit given the demand scenarios. Controllable processing times enlarge the solution space so that the limited capacity of production resources are utilized more effectively. We propose an effective formulation that enables an extensive computational study. Our computational results clearly indicate that instead of relying on relatively simple heuristic methods, multi-stage stochastic programming can be used effectively to solve MPS problems, and that controllability increases the performance of multi-stage solutions.

1. Introduction

Master Production Schedules (MPS) are widely used by manufacturing facilities to handle production and scheduling decisions. In current industry practice, the MPS produces production schedules in a finite planning horizon, assuming infinite capacity, fixed processing times, and deterministic demand.

Our study is motivated by the following application. The largest auto manufacturer in Turkey recently introduced a new multi-purpose vehicle to the market. The company installed a single production line with a limited production capacity and dedicated it to this particular model. Since the production facilities are flexible, the processing times could be altered or controlled (albeit at higher manufacturing cost) by changing the machining conditions in response to demand changes. As this model is new, the company generated different demand scenarios for each time period. One of the important planning problems was to develop a master production schedule to determine how many units of this new model would be produced in each time period along with the desired cycle time (or equivalently, the optimal processing times) to satisfy the demand and available capacity constraints, with the aim of maximizing the total profit. This plan will be used in their Enterprise Resource Planning (ERP) system as an important input to the materials management module to explode the component requirements and generate the required purchase and shop floor orders for the lower level components.

Motivated by this application, we consider the following problem setting. We have a single work center with controllable processing times. The work center produces a single product type with a given price, manufacturing cost function, processing time upper bound, i.e., processing time with minimum cost, and maximum compressibility value. As in the case of MPS, we have a finite planning horizon. The orders arrive at the beginning of each period and the products are replenished at the end of the period. There is an additional cost of postponement if the replenishment cannot be done by the end of the period.

The demand of the first period is assumed to be known with certainty prior to scheduling. However, the demand of the other periods are uncertain; possible scenarios for demand realizations and their associated probabilities are known. In our MPS calculations, the number of units of demand is defined in terms of the multiples of a base unit. Therefore, a job represents the amount of one base unit. Our objective is to maximize the total expected profit by deciding how many units to produce, when to produce, and how to produce them, i.e., the required processing times.

Our aim in this paper is to question the basic assumptions of MPS regarding infinite capacity, fixed processing times, and deterministic demand, and to propose a new approach that overcomes to an extent the disadvantages caused by these assumptions and is computationally efficient. In the remaining part of this section, we briefly summarize the existing work on MPS, scheduling with controllable processing times, and multi-stage stochastic programming. We conclude the section with an example that motivates our study.

* Corresponding author. Tel.: +90 3122901360; fax: +90 3122664054.
E-mail addresses: ekorpeog@andrew.cmu.edu (E. Körpeoğlu), hyaman@bilkent.edu.tr (H. Yaman), akturk@bilkent.edu.tr (M. Selim Aktürk).

0377-2217/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.ejor.2011.02.032

The classical approach for generating Master Production Schedules assumes known demands, infinite capacity, and fixed processing times. In the current literature on MPS, the demand uncertainty is ignored during the schedule generation. As a result, the main research focuses on the length of the frozen time period, i.e., the number of periods in which production scheduling decisions are not altered even when demand realizations turn out to be different than the estimates. A longer frozen time period is less responsive to demand changes, but creates less nervousness, while a shorter one acts oppositely. Studies by Sridharan et al. (1987) and Tang and Grubbström (2002) are examples that consider the effect of the length of the frozen zone on production and inventory costs. Based on his industry experience, Vieira (2006) points out that the real complexity involved in making a master plan arises when capacity is limited and when products have the flexibility of being produced at different settings. As opposed to the current literature, we consider different demand scenarios with given probabilities along with the controllable processing times and finite capacity of the available production resources while generating the schedule.

There are several instruments that can be used to control processing times. For example, in computer numerical control (CNC) machining operations, the processing time can be controlled by changing the feed rate and the cutting speed. As the cutting speed and/or the feed rate increases, the processing time of the operation compresses at an additional cost that arises due to increased tooling costs, as discussed in Gurel and Akturk (2007). This scenario results in a strictly convex cost function for compression. Cheng et al. (2006) study a single machine scheduling problem with controllable processing times and release dates. They assume that the cost of compression is a linear function of the compression amounts. Leyvand et al. (2010) provide a unified model for solving single-machine scheduling problems with due date assignment and controllable job-processing times. They assume that the job-processing times are either a linear or a convex function of the amount of a continuous and nonrenewable resource that is to be allocated to the processing operations. In our study, we define the compression cost function $f(y) = \kappa \cdot y^{a/b}$ as discussed in Kayan and Akturk (2005), where y is the amount of compression, a and b are two positive integers such that $a > b > 0$, and κ is a positive real number. We use a nonlinear compression cost function as opposed to a linear cost function as widely used in the literature, since it reflects the law of diminishing marginal returns.

A review of scheduling with controllable processing times can be found in Shabtay and Steiner (2007), in which they also summarize possible applications in a steel mill and in an automated manufacturing environment in addition to the automotive industry example that we have discussed above. As far as our problem is concerned, controllable processing times may constitute a flexibility in capacity since the maximum production amount can be increased by compressing the processing times of jobs with, of course, an additional cost. Thus, this scenario brings up the trade-off between the revenue gained by satisfying an additional demand and the amount of compression cost. The value of controllable processing times becomes even more evident during economic crises, since they allow companies to adjust their production quantities to meet the immediate demand that varies significantly during the planning horizon more effectively.

Stochastic programming uses mathematical programming to handle uncertainty. Although deterministic optimization problems are formulated with parameters that are known with certainty, in real life it is difficult to know the exact value of every parameter during planning. Stochastic programming handles uncertainty assuming that probability distributions governing the data are known or can be estimated. The goal here is to maximize the expectation of some function of the decisions and random variables. Such models are formulated, analytically or numerically solved, and then analyzed in order to provide useful information to a decision-maker.

Two-stage stochastic programs are the most widely used versions of stochastic programs. The decision maker takes some action in the first stage, after which a random event occurs that affects the outcome of the first-stage decision. A recourse decision can then be made in the second stage to compensate for any negative effect that might have been experienced as a result of the first-stage decision. A detailed explanation of stochastic programming, its applications, and solution techniques can be found in Birge and Louveaux (1997) and a survey of two-stage stochastic programming is given in Schultz et al. (1996). Using more than one stage in decision making is also utilized in robust optimization. Atamturk and Zhang (2007) apply two-stage robust optimization to network flow and design problems. They give a numerical example that explains the benefit of using two stages instead of a single one.

In multi-stage stochastic programming, decisions are made in several decision stages instead of two. At each stage, a different decision is made or recourse action is taken. Multi-stage stochastic programming models may yield better results than two-stage models since they incorporate data as they become available, and hence enable a more certain environment for decision making. On the other hand, they are generally more difficult to solve than their two-stage counterparts, therefore, their applications are rare.

In the context of production planning, the early work of Holt et al. (1956) explicitly considers uncertain demand and flexible workforce capacity, whereas Charnes et al. (1958), Bookbinder and Tan (1988) and Orcun et al. (2009) use chance constraints to address problems with uncertain demand. Furthermore, Peters et al. (1977), Escudero et al. (1993), Voss and Woodruff (2006), Karabuk (2008) and Higle and Kempf (2011) apply multi-stage stochastic programming to production planning. Balibek and Koksalan (2010) apply a multi-objective multi-stage stochastic programming approach for the public-debt management problem. Guan et al. (2006) study the uncapacitated lot-sizing problem and Ahmed et al. (2003) study the capacity expansion problem with uncertain demand and cost parameters. Huang and Ahmed (2009) provide analytical bounds for the value of multi-stage stochastic programming over the two-stage approach for a general class of capacity planning problems under uncertainty. To the best of our knowledge, there is no study in the literature that applies multi-stage stochastic programming to master production scheduling. Stochastic programming problems are generally considered difficult (Dyer and Leen, 2006).

When the uncertain parameters evolve as a discrete-time stochastic process with finite probability space, the uncertainty can be represented with a scenario tree; Fig. 1.1 depicts an example.

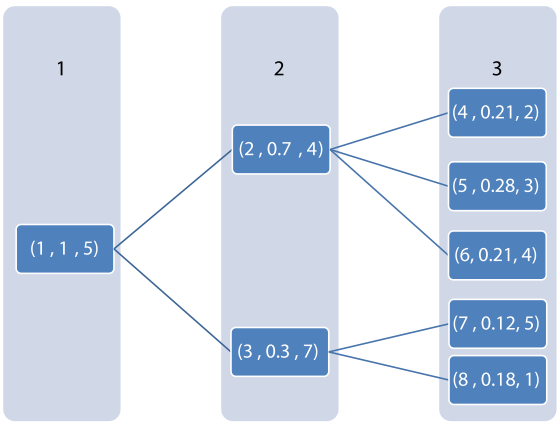

Fig. 1.1. A scenario tree for three periods.

The nodes of the tree represent demand scenarios for periods. For each node, we give in parentheses, the node number, the probability (not the conditional but the actual probability) of realization of that node, and the corresponding demand realization. For instance, node 2 corresponds to the scenario in which a demand of four is realized at period 2 and its probability is 0.7. A path starting from the root node and ending at a leaf node represents a scenario in the decision tree and each scenario path can be uniquely defined by a leaf node. For instance, 1–2–6 is a path that can uniquely be represented by node 6.

In our master production scheduling problem, we use a multi-stage stochastic programming approach and a scenario tree in order to handle the uncertainty in demand. Since information on the demand of each period becomes available at the beginning of the period, our decision stages correspond to periods. We use the following example to highlight the main ideas behind the proposed study.

Example 1. Consider the scenario tree in Fig. 1.1. In the classical MPS, the planner needs to define fixed values for demand realizations. There are several strategies available to choose this single scenario:

(1) choosing the most likely scenario, which is 1–2–5,
(2) choosing the most optimistic scenario, which is 1–3–7,
(3) choosing the most pessimistic scenario, which is 1–2–4, and
(4) using rounded expected demand values; in our example, this corresponds to the scenario in which the demands are 5, 5, and 3 for the first three periods, respectively.

The fifth option is to use multi-stage stochastic programming. Suppose that the compression cost function is $f(y) = y^{\frac{3}{2}}$, the net unit revenue is 60, the time required to process a job at minimum cost is 10 time units, the maximum compression amount is four time units, and the capacity is 36 time units. For simplicity, we assume that the postponement and the shortage costs are zero. The cost incurred due to excess production is ξ per item. We do not assign a value to ξ at this point since we do not want this assumption to affect the overall results.

In Table 1.1, we report the maximum profits for each strategy and scenario realization. Clearly, the solutions based on single scenarios have the best performance for their own scenarios. However, we observe that they have very poor results if the realized scenario is different. The multi-stage stochastic programming solution has the best or second-best performance in all scenarios and has the maximum expected profit.

Another measure that can be used to evaluate the strategy performance is relative regret. The relative regret of a solution at a given scenario is the percentage difference between the profit of this solution and the optimal profit in that scenario. To calculate the relative regrets, we need to assign a value to ξ. We consider a somewhat small $\xi = 10$, and the results are given in Table 1.2.

Here we see that the relative regret of the multi-stage stochastic programming solution is very small compared to those of the other solutions when the solution is not optimal for the scenario in consideration. In all cases, the profit of the multi-stage stochastic programming solution is within 5% of the actual optimal profit, while the profits of the other solutions may deviate up to 32%.

Therefore, we conclude that, in this example, using a multi-stage stochastic programming approach instead of using fixed demand estimates significantly improves the outcomes.

Using controllable processing times instead of fixed processing times also increases schedule performance. For instance, in this example, the profit of the multi-stage stochastic programming solution decreases to 540 in every scenario if the processing times are fixed, which clearly indicates that the controllability of processing times enlarges our solution space and enables us to utilize the limited capacity of the production resources more effectively.

The structure of the paper is as follows: In Section 2, we present the notation and a nonlinear and a linear integer programming formulation. Then we study two subproblems and use their outcomes to derive an alternative linear integer programming formulation, which turns out to be quite efficient. In Section 3, we present and discuss the results of our computational study, with emphasis on the assumptions of the traditional MPS on infinite capacity, fixed processing times, and deterministic demand. We conclude the paper in Section 4.

2. Multi-stage stochastic programming

As explained above, we consider a capacitated version of the MPS where demand is uncertain and processing times are controllable. Thus, the decisions involved in this problem are how much to produce, when to produce, and the required processing times.

Table 1.1
Maximum profits of different strategies.

Realized scenario	Prob	Possible strategies				
		Pessimistic	Most likely	Optimistic	Expected demand	Multi-stage
1–2–4	0.21	628.4	$568.4 - \xi$	$485.8 - 6\xi$	$588.4 - 2\xi$	604.2
1–2–5	0.28	628.4	672.6	$545.8 - 5\xi$	$648.4 - \xi$	664.2
1–2–6	0.21	628.4	672.6	$605.8 - 4\xi$	708.4	708.4
1–3–7	0.12	628.4	672.6	845.8	708.4	845.8
1–3–8	0.18	628.4	672.6	$605.8 - 4\xi$	708.4	672.9
Expected profit		628.4	$650.7 - 0.2\xi$	$592.6 - 4.2\xi$	$666.4 - 0.7\xi$	684.2

Table 1.2
Relative regrets of different strategies.

Realized scenario	Prob	Possible strategies				
		Pessimistic	Most likely	Optimistic	Expected demand	Multi-stage
1–2–4	0.21	0.0	11.1	32.2	9.5	3.9
1–2–5	0.28	6.6	0.0	26.3	5.1	1.2
1–2–6	0.21	11.3	5.1	20.1	0.0	0.0
1–3–7	0.12	25.7	20.5	0.0	16.2	0.0
1–3–8	0.18	11.3	5.1	20.1	0.0	5.0
Expected regret		10.0	7.1	21.2	5.5	2.0

In this section, we first give a nonlinear formulation that determines when to produce and how much to produce, assuming that the profit of producing a certain number of jobs is given. After that, we introduce an equivalent linear integer programming formulation.

Next, we study two subproblems. The outcomes of our study of the first subproblem is used to decide on the optimal processing times and to compute the profit of producing a given number of jobs. We use the results of the second subproblem to reduce the size of our formulation. Finally, we give an alternative linear formulation using these results.

2.1. Notation and problem definition

Let T be the number of periods in the planning horizon. Let N be the set of nodes of the scenario tree and N_t be the set of the nodes of period $t = 1,2,\ldots,T$. For node $i \in N$, let d_i be the demand estimate in the corresponding scenario, D_i be the set of descendants of i including i, B_i be the set of predecessors of i including i, γ_i be the probability of realizing node i with $\gamma_1 = 1$, and finally s_i be the period of node i. For $i \in N$ and $j \in D_i$, let P_{ij} be the set of the nodes on the path from i to j in the scenario tree.

We define the net unit revenue h as the difference between the unit price and the sum of all unit costs, except compression and postponement costs. We denote the processing time of a job with minimum compression cost by p, the maximum compression amount by u, and the capacity by C. We assume that h, p, and C are positive, and u is non-negative. Let k_{max} be the maximum number of jobs that can be produced in a period without violating the capacity constraint, i.e., $k_{max} = \left\lfloor \frac{C}{p-u} \right\rfloor$. We denote the cost of postponing one job for t periods with $b(t)$ and assume that $b(t)$ is a convex function with $b(0) = 0$.

Let $\Pi(k)$ be the maximum profit excluding the cost of postponement when k jobs are produced in a period. For the time being, we assume that $\Pi(k)$ is given for all possible values of k. Later, we explain how this value is calculated.

Given the parameters above, the problem is to decide how many units of the demand of each period to satisfy, and when and with what processing time to produce it in each scenario so that the capacities are respected and the expected profit is maximized. We refer to this problem as multi-stage master production scheduling and abbreviate it as *MMPS*.

2.2. A nonlinear and a linear integer programming model

In this section, we first present a nonlinear formulation for problem *MMPS*. We use the following decision variables: For node $j \in N$, we define y_j to be the number of jobs produced at node j, and z_j to be the amount of demand of node j that is satisfied within the planning horizon. For node $i \in N$ and for $j \in B_i$, we define x_{ij} to be the amount of demand of node j that is produced at node i.

Our first formulation for *MMPS*, referred to as *MMPS-N*, is as follows:

$$(\textit{MMPS-N}) \quad \max \quad \sum_{i\in N} \gamma_i \cdot (\Pi(y_i) - \sum_{j\in B_i} b(s_i - s_j)\cdot x_{ij}) \tag{2.1}$$

$$\text{s.t.} \quad \sum_{j\in B_i} x_{ij} = y_i \quad \forall i \in N, \tag{2.2}$$

$$\sum_{i\in P_{jm}} x_{ij} = z_j \quad \forall m \in N_T \cap D_j,\ j \in N, \tag{2.3}$$

$$z_j \leqslant d_j \quad \forall j \in N \tag{2.4}$$

$$y_j \leqslant k_{max} \quad \forall j \in N, \tag{2.5}$$

$$x_{ij} \in \mathbb{Z}_+ \quad \forall i \in N,\ j \in B_i, \tag{2.6}$$

$$z_i \in \mathbb{Z}_+ \quad \forall i \in N, \tag{2.7}$$

$$y_i \in \mathbb{Z}_+ \quad \forall i \in N. \tag{2.8}$$

The objective function (2.1) is equal to the total expected profit. Constraints (2.2) link the variables x_{ij}'s and y_i's. The amount of production at a given node i is equal to the sum of the amounts of production done at node i to satisfy the demand of its preceding nodes. Constraints (2.3) ensure that the amount of the demand satisfied for a given node j is equal in all scenarios that include node j. To this end, these constraints impose the requirement that z_j, which is the amount of demand of node j that is satisfied, is equal to the sum of the amounts of production done to satisfy the demand of node j over each path that starts at node j and ends at a descendant leaf node. Constraints (2.4) ensure that the amount of demand of node j that is satisfied within the planning horizon is no more than the demand at node j. Capacity restrictions are imposed through constraints (2.5). Finally, the integrality and nonnegativity of variables are given in constraints (2.6)–(2.8).

The model *MMPS-N* has a nonlinear objective function. Next, we propose a linear integer programming formulation for problem *MMPS*. To obtain this formulation, we rewrite the integer variables y_i's as weighted sums of binary variables. We define w_{ik} to be 1 if k jobs are produced at node i and 0 otherwise for all $i \in N$ and $k \in \{0,1,\ldots,k_{max}\}$. Clearly, we need $\sum_{k=0}^{k_{max}} w_{ik} = 1$ for all $i \in N$. Now, $y_i = \sum_{k=0}^{k_{max}} k \cdot w_{ik}$ and $\Pi(y_i) = \sum_{k=0}^{k_{max}} \Pi(k) \cdot w_{ik}$ for all $i \in N$. Substituting in the above formulation, and adding the constraints that ensure that for each node $i \in N$, exactly one k value in $\{0,1,\ldots,k_{max}\}$ is picked as the production amount, we obtain the following linear integer programming formulation, referred to as *MMPS-L1*.

$$(\textit{MMPS-L1}) \quad \max \quad \sum_{i\in N} \gamma_i \cdot \left(\sum_{k=0}^{k_{max}} \Pi(k)\cdot w_{ik} - \sum_{j\in B_i} b(s_i - s_j)\cdot x_{ij} \right)$$

$$\text{s.t.} \quad (2.3), (2.4), (2.6), (2.7)$$

$$\sum_{k=0}^{k_{max}} w_{ik} = 1 \quad \forall i \in N,$$

$$\sum_{j\in B_i} x_{ij} = \sum_{k=0}^{k_{max}} k \cdot w_{ik} \quad \forall i \in N,$$

$$w_{ik} \in \{0,1\} \quad \forall i \in N,\ k \in \{0,1,\ldots,k_{max}\}.$$

In this formulation, since w_{ik} values are defined only for feasible production amounts, there is no need for capacity constraints (2.5).

In formulations *MMPS-N* and *MMPS-L1*, the maximum number of jobs that can be produced in a period is computed using the capacity restrictions and is equal to k_{max}. Moreover, we assume that the values of the profit function $\Pi(k)$ for k in $\{0,1,\ldots,k_{max}\}$ are given. As the total profit is equal to the number of jobs times unit revenue minus the manufacturing costs, this implies that the optimal compression amounts have to be computed for each k value. Next, we introduce two subproblems which are used to reduce the possible number of jobs produced in a given period and to calculate the optimal compression amounts and maximum profits for a given number of jobs.

2.3. The single-period capacitated deterministic scheduling problem with a cost minimization objective

In this section, we introduce and study our first subproblem, which is the single-period capacitated deterministic scheduling problem with a cost minimization objective. The results that we obtain for this problem are used to define the optimal compression costs and the $\Pi(k)$ values.

In this problem, we have a single work center and identical products. The work center has a finite capacity of C. Suppose that the processing time of a job with the minimum compression cost is p and the maximum compression amount is u. There are $n \leqslant k_{max}$ jobs in the work center. The compression cost function is

$f:\mathbb{R}_+ \to \mathbb{R}_+$ and is strictly convex. The problem is to decide on the compression amounts of the n jobs with the aim of minimizing the total compression costs. We define the variable c_j to be the compression amount of job j in $\{1,\ldots,n\}$. Now, this problem can be formulated as follows:

$$\min \quad \sum_{j=1}^{n} f(c_j) \tag{2.9}$$

$$\text{s.t.} \quad c_j \leqslant u \quad \forall j \in \{1,\ldots,n\}, \tag{2.10}$$

$$\sum_{j=1}^{n} (p - c_j) \leqslant C, \tag{2.11}$$

$$c_j \in \mathbb{R}_+ \quad \forall\, j \in \{1,\ldots,n\}. \tag{2.12}$$

The objective function (2.9) is equal to the sum of the compression costs. Constraints (2.10) ensure that the compression amounts do not exceed the maximum amount u and constraint (2.11) ensures that the sum of processing times does not exceed the capacity. Constraints (2.12) are nonnegativity constraints.

In the following proposition, we characterize the optimal solution to this problem.

Proposition 2.1. *Let n be a positive integer with $n\cdot(p-u)\leqslant C$. If n jobs are to be produced in a work center, then the solution with $c_j = \max\{p-\frac{C}{n},0\}$ for all $j=1,\ldots,n$ is the unique optimal solution to the above problem.*

Proof. First, we show that in the optimal solution, the compression amount is equal for all the jobs in the work center. Let c be an optimal solution. Suppose to the contrary that there exist jobs i and j such that $c_i > c_j$. Let $\bar{c}$ be the same as c except $\overline{c_i} = \overline{c_j} = \frac{c_i+c_j}{2}$. The solution $\bar{c}$ is feasible and by strict convexity of the cost function, $f(\overline{c_i})+f(\overline{c_j}) < f(c_i)+f(c_j)$. This contradicts the optimality of the initial solution c. Now, it follows immediately that $c_j = \max\{p-\frac{C}{n},0\}$ for all $j=1,\ldots,n$ is an optimal solution. □

Proposition 2.1 is intuitive. As the compression cost function is strictly convex, the greater the compression amount, the more the marginal compression cost is incurred. Thus, in order to minimize the total compression cost, the necessary compression amount $\max\{n\cdot p - C, 0\}$ is evenly distributed among all jobs. Using Proposition 2.1, it is possible to find the optimal compression amounts for jobs, given the optimal allocation of jobs to the nodes. Moreover, we can compute the $\Pi(k)$ values using our compression cost function $f(y)=\kappa\cdot y^{a/b}$.

For $x\in\mathbb{R}_+$, we define $\Pi(x) = \begin{cases} x\cdot h - x\cdot\kappa\cdot\left(p-\frac{C}{x}\right)^{\frac{a}{b}} & \text{if } x > \frac{C}{p}, \\ x\cdot h & \text{otherwise.}\end{cases}$

Corollary 2.2. *Let n be a positive integer with $n\cdot\ (p-u)\leqslant C$. If n jobs are to be produced at a work center in a period, then the maximum profit at the work center is $\Pi(n)$.*

Using Corollary 2.2, the profit function is calculated for all possible values of job numbers at a node and is given as an input to formulations *MMPS-N* and *MMPS-L1*.

2.4. The single-period capacitated deterministic problem with a profit maximization objective

In the previous sections, we make use of the fact that at most k_{max} units can be produced due to capacity constraint. In this section, we propose a tighter upper bound for the maximum possible production using the concavity of the profit function and reduce the size of formulation *MMPS-L1* by reducing the number of w_{ij} variables significantly. The following example illustrates the scope of this reduction.

Example 2. Suppose that $h=200$, $C=30$, $p=10$, $u=8$, and $f(y)=y^3$. Consequently, $k_{max}=15$ meaning that *MMPS-L1* requires $15|N|$ binary variables (w_{ij}'s). However, as we propose in this section, the maximum production in an optimal solution could be at most 4, so that the required number of binary variables can be reduced to $4|N|$.

To obtain the tight bound illustrated in Example 2, we consider a subproblem in which we have a single work center with finite capacity C and a single period with infinite demand. The objective is to decide on the number of jobs to produce to maximize the total profit.

We define the threshold value, denoted by τ, to be the optimal number of jobs to be produced at the work center so that the total profit is maximized. Hence, the problem is:

$$\begin{aligned} \max \quad & \Pi(n) \\ \text{s.t.} \quad & n \leqslant k_{max}, \\ & n \in \mathbb{Z}_+. \end{aligned}$$

The value of τ depends on both the available capacity and the relative profit gain of producing one more job. Although we could have used an enumerative approach to compute τ in $O(k_{max})$ time, we use the following lemma to compute τ analytically.

Lemma 2.3. *The profit function Π satisfies the following properties:*

i. $\Pi(x)$ is continuously differentiable on $\mathbb{R}_{++}$,
ii. $\Pi(x)$ is concave,
iii. if $h<\kappa\cdot p^{\frac{a}{b}}$, there exists x^* in $\left(\frac{C}{p},+\infty\right)$ such that $\frac{d\Pi}{dx}(x^*)=0$.

Proof. Let $\Pi^c:(\frac{C}{p},+\infty)\to\mathbb{R}$ be defined as $\Pi^c(x) = x\cdot h - x\cdot\kappa\cdot(p-\frac{C}{x})^{\frac{a}{b}}$. Then,

$$\Pi(x) = \begin{cases} \Pi^c(x) & \text{if } x > \frac{C}{p}, \\ x\cdot h & \text{otherwise.}\end{cases}$$

When $x<\frac{C}{p}$, $\Pi(x)$ is linear. When $x>\frac{C}{p}$, $\Pi(x)=\Pi^c(x)$ is a smooth function since $x\neq 0$. Therefore, the only point that needs consideration is $x=\frac{C}{p}$. The first derivative of the Π^c function with respect to x is:

$$\frac{d\Pi^c}{dx}(x) = h - \kappa\cdot\left(p-\frac{C}{x}\right)^{\frac{a}{b}} - \frac{a}{b}\cdot\kappa\cdot\left(p-\frac{C}{x}\right)^{\frac{a}{b}-1}\cdot\frac{C}{x}.$$

Therefore, the right limit of $\frac{d\Pi}{dx}(x)$ at $x=\frac{C}{p}$ is h. The derivative of hx with respect to x is h so the left limit of $\frac{d\Pi}{dx}(x)$ at $x=\frac{C}{p}$ is also h. Therefore, $\frac{d\Pi}{dx}(x)$ is continuous on $\mathbb{R}_{++}$, hence $\Pi(x)$ is continuously differentiable on $\mathbb{R}_{++}$.

The second derivative of $\Pi^c(x)$ with respect to x is:

$$\begin{aligned} \frac{d^2\Pi^c}{dx^2}(x) &= -\frac{a}{b}\cdot\kappa\cdot\left(p-\frac{C}{x}\right)^{\frac{a}{b}-1}\cdot\frac{C}{x^2} - \frac{a}{b}\cdot\left(\frac{a}{b}-1\right)\cdot\kappa\cdot\left(p-\frac{C}{x}\right)^{\frac{a}{b}-2} \\ &\quad\cdot\frac{C^2}{x^3} + \frac{a}{b}\cdot\kappa\cdot\left(p-\frac{C}{x}\right)^{\frac{a}{b}-1}\cdot\frac{C}{x^2} \\ &= -\frac{a}{b}\cdot\left(\frac{a}{b}-1\right)\cdot\kappa\cdot\left(p-\frac{C}{x}\right)^{\frac{a}{b}-2}\cdot\frac{C^2}{x^3} \leqslant 0 \end{aligned}$$

since $a>b$ and $x\geqslant\frac{C}{p}$, $\frac{d\Pi^c}{dx}(x)$ is monotonically decreasing. Moreover, h is monotonically nonincreasing. In addition to those, the derivative function of $\Pi(x)$ is continuous. Thus, $\frac{d\Pi}{dx}(x)$ is monotonically non-increasing and continuous, hence $\Pi(x)$ is concave.

Now suppose that $h < \kappa \cdot p^{\frac{a}{b}}$. When x tends to $\frac{C}{p}$, $\frac{d\Pi^c}{dx}(x) > 0$ and when x tends to infinity, $\frac{d\Pi^c(x)}{dx} < 0$ since $h < \kappa \cdot p^{\frac{a}{b}}$. In addition to that, the derivative function is continuous. Then, by the intermediate value theorem, there exists x^* in $\left(\frac{C}{p}, +\infty\right)$ such that $\frac{d\Pi^c}{dx}(x^*) = 0$. Since $\Pi(x)$ has the same values as $\Pi^c(x)$ on the domain $\left(\frac{C}{p}, +\infty\right)$, then $\frac{d\Pi}{dx}(x^*) = 0$. □

By Lemma 2.3, we know that the profit function is concave and has a critical point within its domain if $h < \kappa \cdot p^{\frac{a}{b}}$. Thus, one can find this critical point and by concavity this critical point is the maximizing point within a continuous domain if $h < \kappa \cdot p^{\frac{a}{b}}$. Obviously, this does not immediately tell what the τ value is since τ is the maximizing value among only integer points. Moreover, the critical point does not take into account the capacity constraint. Proposition 2.4 uses Lemma 2.3 to compute the actual threshold value.

Proposition 2.4. *Suppose that $h < \kappa \cdot p^{\frac{a}{b}}$. Let x^* be the critical point of $\Pi^c(x)$. Then,*

$$\tau = \begin{cases} k_{max} & \text{if } x^* > k_{max}, \\ \Pi\lceil x^* \rceil & \text{if } x^* \leqslant k_{max} \text{ and } \Pi\lceil x^* \rceil > \Pi\lfloor x^* \rfloor, \\ \Pi\lfloor x^* \rfloor & \text{otherwise.} \end{cases}$$

On the other hand, if $h \geqslant \kappa \cdot p^{\frac{a}{b}}$, then $\tau = k_{max}$.

Proof. Follows from the concavity of $\Pi(x)$. □

Using the threshold value, it is possible to reduce the size of formulation *MMPS-L1* such that k_{max} in the main formulation can be replaced by τ, as stated below. We omit the proof as it is easy.

Proposition 2.5. *At an optimal solution to MMPS-N, the production amounts of all nodes are less than or equal to the threshold value, τ.*

2.5. An alternative linear integer programming formulation

In this section, we give an alternative linear integer programming formulation for *MMPS*. This formulation uses the results of the previous sections on the concavity of the profit function and the threshold value. In this formulation, we rewrite the integer variables y_i's as the sum of binary variables. We define v_{ik} to be 1 if at least k jobs are produced at node i and 0 otherwise for all $i \in N$ and $k \in \{1, \ldots, \tau\}$. Then for $i \in N$, $y_i = \sum_{k=1}^{\tau} v_{ik}$ and $\Pi(y_i) = \sum_{k=1}^{\tau} (\Pi(k) - \Pi(k-1)) \cdot v_{ik}$ with $v_{ik} \geqslant v_{i(k+1)}$ for all $k \in \{1, \ldots, \tau - 1\}$. This formulation, referred to as *MMPS-L2*, is as follows:

$$(\textit{MMPS-L2}) \quad \max \sum_{i \in N} \gamma_i \cdot \left(\sum_{k=1}^{\tau} (\Pi(k) - \Pi(k-1)) \cdot v_{ik} - \sum_{j \in B_i} b(s_i - s_j) \cdot x_{ij} \right) \tag{2.13}$$

$$\text{s.t.} \quad (2.3), (2.4), (2.6), (2.7),$$

$$v_{ik} \geqslant v_{i(k+1)} \quad \forall i \in N,\ k \in \{1, \ldots, \tau - 1\}, \tag{2.14}$$

$$\sum_{j \in B_i} x_{ij} = \sum_{k=1}^{\tau} v_{ik} \quad \forall i \in N,$$

$$v_{ik} \in \{0, 1\} \quad \forall i \in N,\ k \in \{1, \ldots, \tau\}.$$

Constraints (2.14) of *MMPS-L2* ensure that if at least $k+1$ jobs are produced at a node, then clearly, at least k jobs are produced. These constraints can be removed without changing the optimal value since $\Pi(x)$ is concave. Let *MMPS-L3* be the resulting formulation.

To conclude this section, we compute the number of variables and constraints in formulations *MMPS-L1* and *MMPS-L3*. The number of variables x_{ij}'s is equal to $\sum_{t=1}^{T} |N_t| t$ since a node in set N_t has t predecessors including itself. The number of variables in formulation *MMPS-L1* is equal to $\sum_{t=1}^{T} |N_t| t + |N| + |N|(k_{max} + 1)$ and the number of variables in formulation *MMPS-L3* is equal to $\sum_{t=1}^{T} |N_t| t + |N| + |N|\tau$. The number of constraints (2.3) is equal to $T|N_T|$. Consequently, the formulation *MMPS-L1* has $T|N_T| + 3|N|$ constraints, whereas the formulation *MMPS-L3* has $T|N_T| + 2|N|$ constraints. It can be seen that the sizes of both formulations depend on the number of nodes in the scenario tree, which grows exponentially with the degrees of the nodes and the number of periods.

In Appendix A, we introduce two trivial cases and analyze two special cases of the problem that are polynomially solvable.

3. Computational results

The computational study consists of three stages. In stage one, we test the CPU time performance of the linear integer programming formulations *MMPS-L1* and *MMPS-L3*. We find that *MMPS-L3* proves to be very efficient in terms of CPU time, solving all the test problems in at most four seconds. In the second stage, we compare the performances of solutions obtained from single-scenario strategies utilizing different production adjustment policies with the multi-stage stochastic programming solution. We also make a thorough analysis on the significance of capacity on solution quality. The results show that multi-stage stochastic programming outperforms single-scenario strategies, and that capacity has a statistically significant effect on solution quality, regardless of the utilized strategy. Finally, in the third stage, we investigate the effect of controllability, and our computational results clearly indicate that adding controllability in a capacitated environment provides a notable improvement in the solution quality of multi-stage stochastic programming.

3.1. Experimental design

Although our study was initially motivated by an industrial application, the master production scheduling setting in the paper is quite general, i.e., can be applied to different production systems. Therefore, we randomly generated data to test the proposed model in different computational settings. In our test problems, we take the number of periods T to be four, as weekly periods in a monthly planning horizon. We set the coefficient of the compression cost function κ to one, and the net unit revenue h to 200. The processing time with minimum cost p is Uniform[10, 15] and the maximum compression amount u is $p \times$ Uniform[0.5, 0.9].

In order to prevent possible parameter selection bias, we take different settings for each parameter, which are determined based on an intensive pre-experimental study. In particular, we parameterize the effect of the magnitude of the compression cost exponent, the capacity tightness, the relative magnitude of postponement costs, distributions that the node probabilities are generated from, the variability of demand over scenarios, and the number of possible scenarios. We also investigate the effect of the ratio between inventory holding and postponement costs in our analysis of single scenario strategies. Table 3.1 summarizes the factors that we find to be significant and the values that they take throughout the study. We take five replications for each of the 384 experimental settings, resulting in 1920 randomly generated scenario trees. All runs are performed using ILOG Cplex Version 11.2 on a 2×2.83 gigahertz Intel Xeon CPU and 8 gigabytes memory workstation HP with the operating system Ubuntu 8.04.

We use two alternatives for the compression cost function exponent a/b, which are 2 and 3. In Fig. 3.1a, we depict the profit function for $a/b = 3$ as an example. In this case, $\tau = 3 < k_{max} = 10$.

Table 3.1
Factors used in the experiments.

Factor	Name	Number of levels	Factor combinations			
			1	2	3	4
A	Compression cost exponent a/b	2	2	3	–	–
B	Capacity scaling factor ζ	4	0.2	0.4	0.6	0.8
C	Postponement cost scaling factor β	2	0.15	0.3	–	–
D	Probability type (node probabilities)	2	Equal	Normal Dist.	–	–
E	Demand variability $[b_{low}, b_{high}]$	3	[0,20]	[10,30]	[0,40]	–
F	Degree factor	2	[0,14]	[7,14]	–	–
G	Inventory cost/ postponement cost	2	0.2	0.8	–	–

The profit function for $a/b = 2$ is given in Fig. 3.1b and here $\tau = k_{max} = 10$. Consequently, with these two values for the compression cost function exponent, we capture both cases, where $\tau < k_{max}$ and $\tau = k_{max}$.

A scenario tree is defined by its nodes and their associated demand scenarios and probabilities. We generate the nodes as follows. To determine the number of immediate descendants of a node, we use the concept of 'degree factor'. The number of immediate descendants of a node is generated from either Uniform[0,14] or Uniform[7,14].

The demand realization at a node is generated by rounding the random variate from Uniform$[b_{low}, b_{high}]$. We refer to this factor as the 'demand variability'. We use three alternative distributions to check different demand variability levels: Uniform[0,20], Uniform[10,30], and Uniform[0,40]. Here, the first two alternatives are used to compare alternatives with the same variance but different means, whereas the last two are used to compare alternatives with the same mean but different variances.

The assignment of probabilities to scenarios is a major component of stochastic programming. Therefore, the final factor concerning the scenario tree is the 'probability factor'. Here we consider two ways of assigning probabilities to the nodes of the scenario. The first way is to assign equal probabilities to the immediate descendants of a node. The second way is to use a normal distribution with mean $\mu = \frac{b_{low}+b_{high}}{2}$ and standard deviation 0.5 μ.

We compute the capacity of a period as $C = \zeta \times p \cdot \frac{b_{low}+b_{high}}{2}$, where ζ is the capacity scaling factor. We use four alternatives for ζ: 0.2, 0.4, 0.6, and 0.8. The postponement function is defined as $b(t) = \beta \cdot h \cdot t^2$ for $t \geqslant 0$, where β is the postponement cost scaling factor and is taken as 0.15 and 0.3. When β is 0.3, we expect the postponement cost to dominate the compression cost.

In the existing literature, the capacity is taken as a fixed parameter. Therefore, in order to deal with demand fluctuations, typically, inventory is carried to future periods for a demand surge that may or may not happen. As one of the important advantages of having controllable processing times, the capacity is no longer fixed and can be adjusted with respect to the immediate demand information. As demonstrated in several just-in-time applications, depending on the cost structures, this option could be more beneficial than carrying inventories. Therefore, we do not consider carrying the inventory as an alternative in our model but instead use the capacity as a buffer, if necessary. However, if a single scenario is used to estimate the demand, inventory is inevitable when the estimated scenario is not realized. Thus, in order to be able to make a comparison, we assign the inventory holding cost a specific value. In practice, the inventory holding cost is generally lower than the postponement cost, thus, we take the inventory holding cost per unit per period as 80% and 20% of the postponement cost coefficient (we assume a linear inventory holding cost function).

3.2. Computation times

In our first experiment, we investigate the performances of our formulations *MMPS-L1* and *MMPS-L3* in terms of CPU times under different input parameters. In this case, we consider two levels for factor B (capacity scaling factor) since the performance of *MMPS-L1* limited us in the total number of runs that could be taken. We consider an additional level, 0.01, for the postponement cost ratio (factor C) in order to test the case where the postponement cost is negligible. We do not use factor G since it will be used to evaluate the single-scenario strategies below. Therefore, we have a total of 144 factor combinations and 720 randomly generated instances.

Formulation *MMPS-L3* solves all the problem instances within at most four seconds. Moreover, *MMPS-L3* always gives better results in terms of CPU time than *MMPS-L1*. Formulation *MMPS-L1* cannot prove optimality within one hour for 10 instances corresponding to two factor combinations. In these instances, the factor levels are: $A = 2$, $B = 0.8$, $D =$ Normal, $E = [0,40]$, and $F = [7,14]$. Factor D is 0.01 and 0.15. These correspond to cases where k_{max} is high due to high capacity, the demand has a larger mean and variability, the number of descendants has a high mean, and the tree is unbalanced in terms of the probabilities assigned to nodes. For the remaining instances, the maximum computation time is 2160 seconds with formulation *MMPS-L1*.

Our experimental results clearly indicate that *MMPS-L3* outperforms *MMPS-L1* and is very efficient in terms of computation time. The significant reduction in computation time is mainly due to the outcomes of Section 2 (such as the threshold value, τ, and concavity of the profit function) and the new formulation. Moreover, its

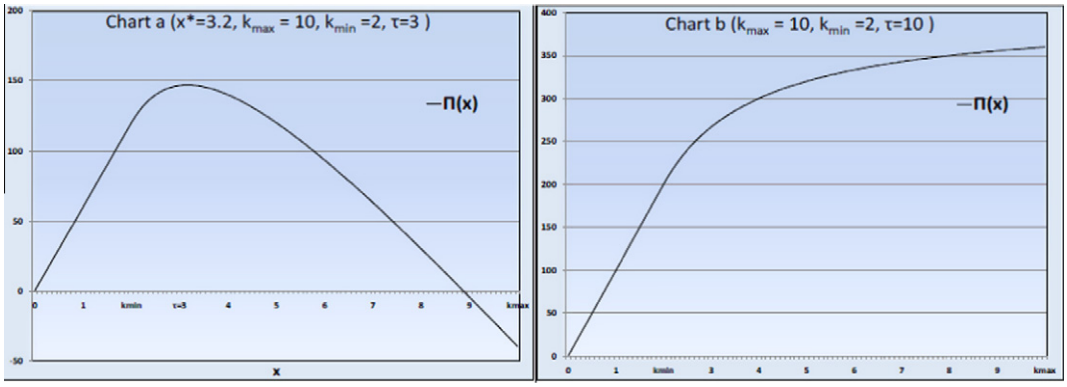

Fig. 3.1. Profit functions for $a/b = 3$ and $a/b = 2$ $\left(k_{min} = \left\lfloor \frac{c}{p} \right\rfloor\right)$.

performance is robust to changes in parameters. This enabled us to perform an extensive computational study to test the quality of multi-stage stochastic programming solutions.

3.3. Comparing multi-stage stochastic programming with single scenario strategies

In the second experiment, we compare the performance of the multi-stage stochastic programming solution (MSP) with the solutions of single-scenario strategies, namely, choosing the most likely scenario (ML), choosing the most optimistic scenario (OPT), choosing the most pessimistic scenario (PES), or using rounded expected demand values (EXP), as shown in Example 1 in the Introduction.

3.3.1. Adjustment policies

As the single-scenario strategies ignore uncertainty in demand, we use two different adjustment policies to improve their solutions.

- *T*-periods frozen policy (TPF): In this policy, demand values are first estimated according to the given single-scenario strategy. The production amounts are determined by solving *MMPS-L3*, where the scenario tree is a path and the demand values are taken as the estimated ones. These production amounts do not change, regardless of the realized demand scenario, i.e., they are frozen for *T* periods.

- One-period frozen myopic adjustments policy (OPF): In this policy, the initial production amounts are calculated just as in TPF. If the estimated scenario is not realized, these production amounts are adjusted myopically as follows: For any period *t*, if there is positive inventory left from previous periods, the production amount is decreased by the amount of inventory, and if there is shortage, then the production is increased by the amount of shortage. This policy corresponds to the chase policy in master production scheduling, since production amounts are more sensitive to immediate demand realizations.

3.3.2. Relative regret as a measure of quality

We compare the solution quality of the single-scenario strategies coupled with the two different production policies (resulting in eight different strategy – policy combinations, e.g., ML-TPF denotes choosing the most likely strategy with the *T*-periods frozen policy) to the quality of the multi-stage stochastic programming solution. We use relative regret as the metric of comparison for two reasons. First, it measures how our solutions perform compared to an optimal solution that we would have if we had perfect knowledge about the input parameters, and hence gives insight into the performance of our methods to hedge against uncertainty. Second, profit values may differ significantly among different settings and may thus cause settings with higher profit values to dominate the results. Relative regret provides a scaled measure to compare performance under different settings.

As discussed above, there are 1920 randomly generated scenario trees. Since the single-scenario strategies are deterministic, it is not possible to compare expected profits or propose a common ground for comparison. Therefore, we use ex-post profit gained by applying the schedules generated by MSP and the other single-scenario strategy – policy combinations. Namely, for each problem instance, we first generate 10 randomly selected scenarios (i.e., select 10 nodes from the leaf nodes of the scenario tree) which represent 10 ex-post demand realizations. Afterwards, for each strategy-policy combination, we compute the production amounts. The total profit is calculated for the given values of the production amounts and demand realizations of the randomly generated scenario. We calculate the relative regret *R* as follows:

$$R = 100 \times \frac{profit_{\text{optimal}} - profit_{\text{strategy}}}{profit_{\text{optimal}}}.$$

To compute the optimal profit for each scenario, we give the realized demand values as an input to *MMPS-L3* and solve a single-scenario model.

The following example explains the procedure in detail.

Example 3. Consider the numerical example given in the Introduction. Suppose that scenario 8 is realized (i.e., the randomly selected scenario is scenario 8). Corresponding (ex-post) demand realizations of periods 1, 2, and 3 are 5, 7, and 1, respectively. Suppose that we select the pessimistic strategy, where the estimated demands are 5, 4, and 2, respectively. Next, we explain how two adjustment policies are applied in this case.

- *T*-periods frozen policy: If the TPF policy is applied, *MMPS-L3* is solved for a single-path scenario where the demands are the estimated demands, which are 5, 4, and 2. The optimal production amounts are 4 in the first period, 4 in the second period, and 3 in the last period.
- One-period frozen myopic adjustments policy: In OPF, first, *MMPS-L3* is solved and the optimal production amounts of 4, 4, and 3 are obtained as explained above. In the first period, 4 units are produced. In the second period, the production of 4 units takes place but a demand of 7 is realized instead of 4. This is compensated for in period 3; the production amount in this period is changed to $3 + 7 - 4 = 6$. In summary, the production amounts are 4, 4, and 6 for periods 1, 2, and 3, respectively.

Now we explain how we compute the profits. Suppose that the TPF policy is chosen. The realized demands are 5, 7, and 1 and the production amounts are 4, 4, and 3. Therefore, there is a total postponement of 4 units and a shortage of 2 units. There is no excess and no inventory. We assume that postponement and shortage costs are zero. In total, 11 units are produced, and so there is a profit of $2 \times \Pi(4) + \Pi(3) = 628.4$.

The optimal policy in this case is to produce 5, 4, and 4, which yields a total profit of 728.4. Thus the relative regret is 11.3% for the PES strategy – TPF policy combination.

3.3.3. The effects of shortage and excess production costs

Excess production or shortage may occur within the planning horizon if the total demand of the realized scenario is less or more than the demand of the estimated scenario. Here we analyze the effects of shortage and excess production costs. In order to have comparable numbers, we define a 'shortage factor' δ_f and an 'excess factor' ξ_f. We assume a linear cost function for excess and shortage costs. The unit costs per period are obtained by multiplying the associated factor by per unit profit *h*, such that they take a value in the range of $h \times \text{Uniform}[0,2]$. We compare the mean of the regret values of the eight strategy – policy combinations described above with the regret value of MSP for different shortage and excess factor values.

If we consider ξ_f and δ_f as exogenous variables, we have a three-dimensional profit function. Fig. 3.2 displays a cross-section from this profit function, where the excess and shortage cost factors are equal. We observe here that the multi-stage solution is always better than the solutions of other strategies, regardless of the values of the excess and shortage factors.

In the figure, some strategy – policy combinations exceed 100% regret (their profit drops to negative) and becomes out of scale. When shortage and excess costs equally increase, the relative difference between MSP and other strategies tends to increase as well. As we checked for isolated effects of increases in shortage and excess costs, we encountered a similar result: MSP always out-

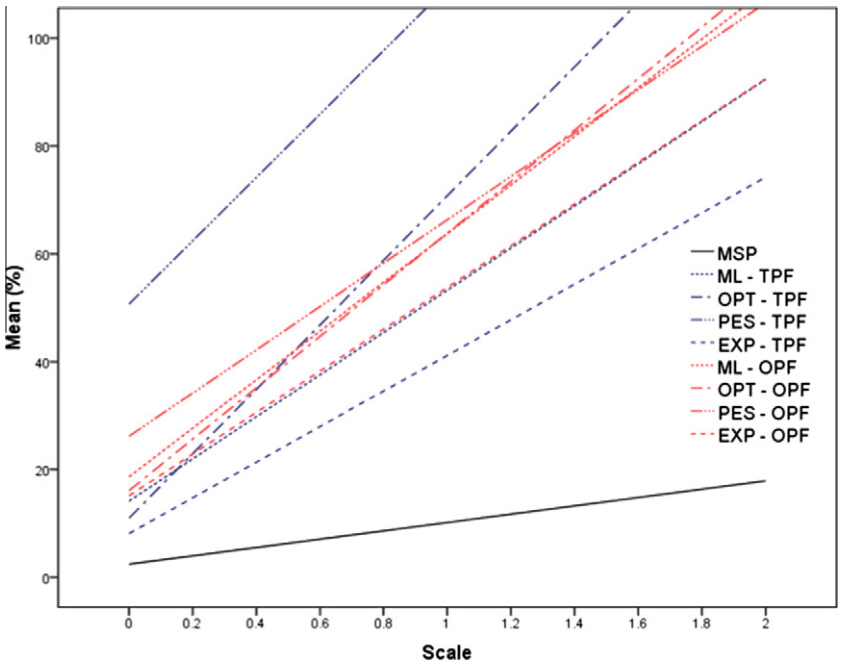

Fig. 3.2. Cross-section of the regret function where shortage and excess are equal.

performs other strategies and as the excess/shortage cost increases, the gap between MSP and the other strategies increases as well, since MSP considers recourse explicitly when computing the optimal policy. Therefore, adding a shortage or excess cost favors MSP against all other strategy – policy combinations. In order not to affect the outcome of the analysis in favor of MSP, we take shortage and excess costs as zero in the remainder of the analysis, noting that all the results we present below can be extended to the case with non-zero excess and shortage costs as well.

3.3.4. Analysis of regret values

In this section, we first present a general discussion on our results. Then we summarize our findings for the experimental factors other than the capacity factor. We defer the discussion on the effect of capacity to the next section.

Table 3.2 gives the average relative regret values of each strategy and policy combination and the number of times a strategy – policy combination gives the minimum regret value for different capacity levels.

As the table suggests, using multi-stage stochastic programming gives a smaller average relative regret value than all other strategies regardless of the adjustment policy. Moreover, MSP has a significant dominance in terms of the number of minimum regret values. The difference between MSP's regret and those of other strategies is significantly high (as much as 50%). Moreover, MSP dominates other strategy-policy combinations in terms of the number of times it achieves the minimum regret values (as high as 99.5%).

Table 3.3 gives the statistics and confidence intervals regarding the differences between the solution quality of single-scenario strategies and multi-stage stochastic programming. As the table shows, the solutions of multi-stage stochastic programming are statistically significantly better than the solutions of all other strat-

Table 3.3
Pairwise statistics of differences of regret levels.

	95 % CI for TPF policy		95 % CI for OPF policy	
	Lower	Upper	Lower	Upper
ML–MSP	11.4	11.9	16.0	16.4
OPT–MSP	8.4	8.7	13.5	13.8
PES–MSP	47.9	48.6	23.5	23.9
EXP–MSP	5.5	5.8	12.5	12.8

Table 3.2
Average relative regrets and the number of times a strategy - policy combination gives the minimum regret value.

Policies	ζ	Performance metric	ML	OP	PES	EXP	MSP[a]
TPF	0.2–0.4	Average regret	15.63	13.34	50.69	9.29	4.80
		# of times min	2099	2766	56	2904	7363
	0.6–0.8	Average regret	12.60	8.56	50.69	6.92	0.05
		# of times min	95	15	0	210	9416
	All	Average regret	14.12	10.95	50.69	8.11	2.42
		# of times min	2194	2781	56	3114	16779
OPF	0.2–0.4	Average regret	18.21	15.63	27.15	14.78	4.80
		# of times min	1583	1822	475	1913	8403
	0.6–0.8	Average regret	19.02	16.47	25.12	15.41	0.05
		# of times min	9	14	0	35	9560
	All	Average regret	18.61	16.05	26.13	15.09	2.42
		# of times min	1592	1836	475	1948	17963

[a] The multi-stage solution is not dependent on policy changes.

egies regardless of the adjustment policy. Moreover the 95%-confidence-interval lower bounds of each strategy – policy combination are considerably high, indicating the superiority of MSP over the single-scenario strategies.

As we do not have enough space to discuss the effects of all seven factors here, we summarize our findings on six factors and give our results concerning the capacity factor in more detail in the next section as it plays an important role in questioning one of the basic assumptions of the traditional MPS: the infinite capacity assumption. Figs. 3.3 and 3.4 illustrate the average relative regret values for each strategy – policy combination and the multi-stage stochastic programming for factors A, C, D ($2^3 = 8$ different settings) and E, F, and G ($3 \times 2^2 = 12$ different settings), respectively. We separated these factors to reduce the number of settings displayed at one time and hence make the figures easier to interpret. In Fig. 3.3, the pessimistic strategy combined with the TPF policy is not illustrated since it is grossly out of scale (its regret values are approximately 80%). As is clear in both figures, the average regret performance of multi-stage stochastic programming is significantly better in all settings than other strategy – policy combinations.

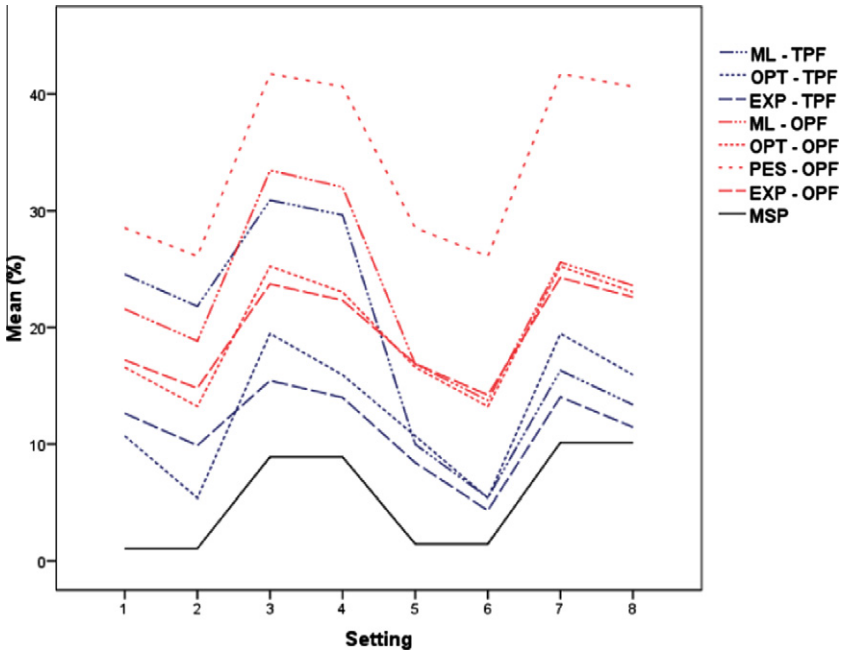

Fig. 3.3. Average regrets of strategy – policy combinations and multi-stage for combinations of factors *A*, *C*, and *D*.

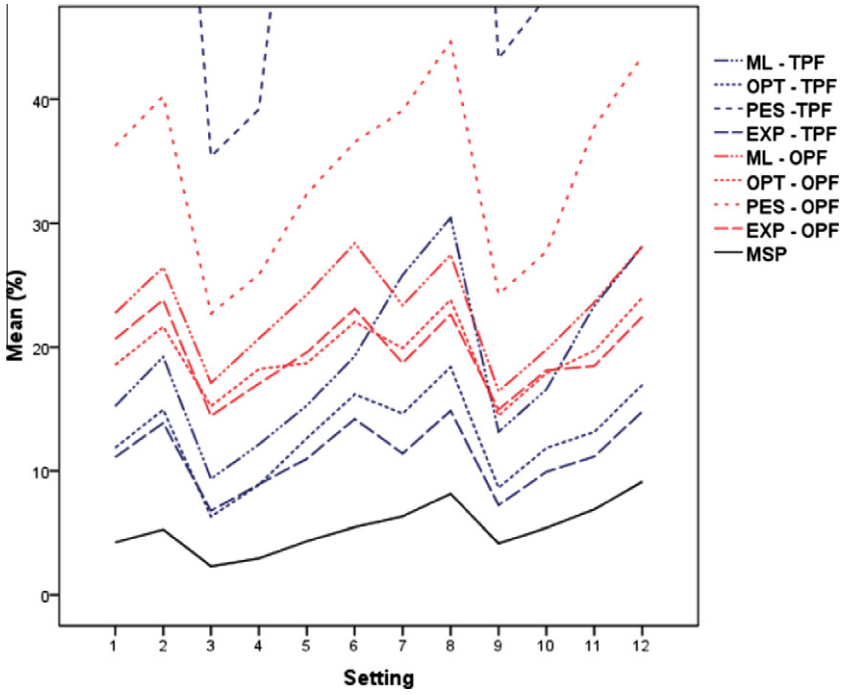

Fig. 3.4. Average regrets of strategy policy combinations and multi-stage for combinations of factors *E*, *F*, and *G*.

3.3.5. The effect of capacity

Here we investigate the effect of capacity on our results. First, in Table 3.2, where the average relative regret values of each strategy and policy combination are given, we can see that the available capacity significantly affects the overall results. The average regret for multi-stage stochastic programming decreases from 4.80% to 0.05% as we relax the capacity. This actually points out that assuming infinite capacity in master production scheduling is unreasonable because solutions are very sensitive to changes in capacity.

To further analyze the effects of the capacity factor on the differences between the solution performances of multi-stage stochastic programming and other strategies, we use paired sample *t*-tests. In Table 3.4, we present the statistics and significance values for the eight strategy – policy combinations and multi-stage stochastic programming.

In all cases, the MSP solution is statistically significantly better than the other solutions, however, the performance difference changes with respect to the capacity factor. For the first three strategies, the capacity effect has a triangular shape as the regret takes its maximum in the middle and its minimum at very low and very high capacity values. EXP has an interesting structure; it has a decreasing regret as capacity becomes looser but an increasing regret as the capacity becomes very loose. For MSP, as capacity decreases, regret increases. This result is quite expected because tighter capacity results in postponing jobs, thus MSP loses its flexibility to adjust to changes in demand. In the OPF policy, the significance of capacity for the first four strategies decreases although capacity still affects solution performance. In general, all strategies are quite sensitive to the available capacity, thus it is essential to revise the unrealistic infinite capacity assumption of the existing MPS algorithms.

3.4. The effect of controllability

In this stage of the experiment, we aim to test the role of controllability on the solution performance of multi-stage stochastic programming. To this end, we generate the same scenario tree with the same parameters and solve the multi-stage stochastic programming formulation with and without controllability. We denote the one with controllability as MSPC and the one without controllability as MSPF. The input parameters and factors are the same as those used in the previous experiment.

Table 3.4
Pairwise statistics of regret differences for different levels of the capacity factor.

	95 % CI for TPF Policy				95 % CI for OPF Policy			
	$\zeta = 0.2$		$\zeta = 0.4$		$\zeta = 0.2$		$\zeta = 0.4$	
	LB	UB	LB	UB	LB	UB	LB	UB
ML vs. MSP	5.9	6.9	14.7	15.7	8.3	9.1	17.7	18.5
OPT vs. MSP	2.2	2.9	14.1	15.0	5.6	6.2	15.4	16.0
PES vs. MSP	37.2	38.7	53.3	54.4	17.5	18.5	26.3	27.1
EXP vs. MSP	1.3	1.9	7.1	7.6	5.7	6.4	13.6	14.2
	$\zeta = 0.6$		$\zeta = 0.8$		$\zeta = 0.6$		$\zeta = 0.8$	
ML vs. MSP	12.6	13.5	11.7	12.5	18.3	19.0	18.9	19.6
OPT vs. MSP	10.9	11.5	5.7	6.0	16.0	16.6	16.2	16.9
PES vs. MSP	51.0	52.4	48.8	50.3	25.4	26.2	24.0	24.8
EXP vs. MSP	6.2	6.6	7.1	7.6	14.5	15.1	15.6	16.3

Table 3.5 summarizes the results of this study. We use three different performance metrics to compare MSPC and MSPF. The first performance metric compares the expected profit values of MSPC and MSPF. The calculation for the scenario-based metric is the same as that used in Section 3.3. We randomly generate 10 scenarios and compare the profits of MSPC and MSPF when these 10 scenarios are realized. In the first two metrics we take the shortage cost as zero. In the third metric, we compare the amount of shortage incurred when the randomly generated scenarios used for metric 2 are realized. Note that for the first two metrics, a higher value is better, but for the third one, a lower value is better. We first calculate the average performance of two alternatives using each of these metrics. Then, we make a pairwise comparison of MSPC and MSPF using each metric and calculate the number of times that one is at least as good as the other.

In Table 3.5, the first two columns give the average and number of times best values for the first metric of expected profit. The results show that in terms of average expected profit, controllability provides an improvement of approximately 23%. The third and fourth columns in the table show the average profit and number of times best values when the scenario-based metric is used. Similar to the first result, controllability improves the solutions approximately 27% on the average, and MSPC finds a solution that is at least as good as MSPF in 99.8% of 19,200 randomly generated runs. Finally, the last two columns give the average (in terms of shortage factor δ_f) and the number of times best values with respect to the third metric. Using controllability decreases the shortage cost by approximately 80% on the average and always gives a shortage value at least as good as the one without controllability. Thus, controllability has a very significant effect on the solution performance of multi-stage stochastic programming.

The significance of capacity is another observation that we make based on the results in Table 3.5; capacity drastically affects the improvement provided by controllability. If capacity is tight, controllability provides an improvement of up to 80%, but this improvement decreases to between 5% and 10% when the capacity is loose. This result is intuitive; controllability provides flexibility in capacity and as capacity gets tighter, flexibility becomes more and more critical.

4. Conclusion

The existing algorithms in the literature to solve the MPS problem are generally based on limiting assumptions of infinite capacity, fixed processing times, and fixed and known demand realizations, despite the fact that there may be few applications where these assumptions can be justified. In this paper, we questioned these assumptions and devised a model with finite capacity, controllable processing times, and uncertain demand values. We used multi-stage stochastic programming to handle this uncer-

Table 3.5
Comparing the multi-stage solution with and without controllability.

Processing times	Capacity	Expected profit		Scenario-based		Shortage cost	
		Average	Num. best	Average	Num. best	Average	Num. best
Controllable (MSPC)	0.2–0.4	56869	9600	51765	9578	$5847\delta_f$	9600
	0.6–0.8	70061	9600	63771	9586	$2\delta_f$	9600
	Total	63465	19200	57768	19164	$2924\delta_f$	19200
Fixed (MSPF)	0.2–0.4	37135	118	30002	118	$27878\delta_f$	0
	0.6–0.8	66105	4766	60766	4766	$2658\ \delta_f$	1440
	Total	51620	4884	45384	4884	$15268\delta_f$	1440

tainty and proposed a very effective formulation that solves large instances in a short computation time, i.e., in a maximum of four seconds. This ability enabled us to conduct an extensive computational study, and our results showed that using multi-stage stochastic programming instead of single-scenario strategies significantly improved the solution quality. Moreover, using controllability provided improvements up to 80% in the total profit. Finally, capacity had a large impact on the solution performances of the single-scenario strategies and the multi-stage stochastic programming approach. Therefore, our computational results suggest that changing these three unrealistic assumptions of MPS provides a huge improvement with little computational cost. The efficiency of our formulation enables further analysis with various different factors, strategies, and policies. Moreover, it provides the time flexibility to conduct sensitivity and what-if analysis.

Being aware of the severe limitations of the MPS algorithms in the current ERP software, firms are making significant investments in new advanced planning and scheduling (APS) software. Unfortunately, these systems rely on relatively simple heuristic methods (such as capable-to-promise, etc.) to solve the MPS problems. Our computational results indicate that the multi-stage stochastic programming approach can be used effectively to solve MPS problems.

As a future research, we could extend the proposed MPS approach to handle a multi-product and multi-resource case. In the proposed approach, node probabilities can be generated from any stochastic process and do not depend on any assumption on the structure of the underlying stochastic process. In the computational study, we employed two different techniques in generating node probabilities, but the demand at a node is independent of the location of the node in the tree. Since the stochastic process that generates node probabilities may be a significant factor, another possible future research direction is the inclusion of the demand correlation over time as one of the factors in the experimental design. Finally, the proposed approach could be implemented using a rolling horizon scheduling approach such that the MPS problem may be solved over the entire planning horizon, but only the imminent period's decision is implemented. Then, the new period's information is appended to maintain a constant length of planning horizon. This new implementation could create a nervousness since the optimum solution may not be the same between two consecutive implementations, which create a necessity to develop a new MPS algorithm to minimize the nervousness in addition to maximizing the overall profit function.

Appendix A. Easy special cases

In this section, we first give some trivial instances of the problem. Afterwards, we introduce two polynomially solvable special cases, namely, the case where there is no postponement cost and the case with the deterministic demand. Finally, we have a negative result: The technique that we use to establish the polynomial solvability of these special cases is not valid for the general case.

A.1. Sufficient conditions for optimality

Suppose that the demand at each node is no more than the threshold value and equal to each other. Then, by Proposition 4.1, the solution where each job is produced at its own node (the node in which the demand of the job is realized) is optimal.

Proposition 4.1. *Suppose that $d_i = d \leqslant \tau$ for all i in set N. Then, the solution where all the demand is satisfied and each job is produced at its own node is optimal.*

Another easy case arises when the demand is not less than the threshold value at all nodes. A sufficient condition for optimality in this case is given in Proposition 4.2.

Proposition 4.2. *Suppose that $d_i \geqslant \tau$ for all $i \in N$. Then, the solution where τ jobs are produced at all nodes is optimal.*

A.2. The stochastic problem with no postponement cost

We propose a different formulation for *MMPS* without any postponement costs. This formulation is based on a simple observation: As we do not have postponement costs, we do not need to keep track of the demands of which periods are satisfied from the production. Therefore, it is sufficient to impose the requirement that we do not produce more than the demand. This special case can be formulated as follows:

$$
\begin{aligned}
(\textit{MMPS-N1}) \quad \max \quad & \sum_{i\in N} \gamma_i \cdot \Pi(y_i) \\
\text{s.t.} \quad & \sum_{j\in P_{1i}} y_j \leqslant \sum_{j\in P_{1i}} d_j \quad \forall i \in N, \qquad (A.1)\\
& y_i \leqslant \tau \quad \forall i \in N, \\
& y_i \in \mathbb{Z}_+ \quad \forall\, i \in N.
\end{aligned}
$$

We call this formulation *MMPS-N1*. Here, constraints (A.1) ensure that for any node the total production amount along the path from node 1 to this node does not exceed the total demand on the same path. Formulation *MMPS-N1* is nonlinear. We use it to establish the complexity status of our problem without postponement costs.

An integral matrix A is totally unimodular if each square matrix of A has a determinant equal to 0, 1, or -1. Hochbaum and Shanthikumar (1990) show that minimizing a convex separable objective function over a set of constraints with a totally unimodular constraint matrix and integer variables is polynomially solvable.

Lemma 4.3. *The constraint matrix of MMPS-N1 is totally unimodular.*

Proof. The constraint matrix consists of three submatrices; the first submatrix consists of the coefficients of constraints (A.1), the second submatrix consists of the coefficients of the upper bound constraints, and the third submatrix consists of the coefficients of the nonnegativity constraints. The last two submatrices are identity matrices. Therefore, it is sufficient to prove that submatrix one is totally unimodular. Each row of submatrix one corresponds to a node in the scenario tree. If we sort the rows of this matrix using the order of a depth-first search on the scenario tree, then this sorted submatrix satisfies the consecutive one's property and thus is totally unimodular (Fulkerson and Gross, 1965). □

Theorem 4.4. *Problem MMPS with no postponement costs is polynomially solvable.*

Proof. For i in N,$\gamma_i \cdot \Pi(y_i)$ is a concave function. The summation of these terms over all $i \in N$ generates a concave separable objective function. Moreover, by Lemma 4.3, the constraint matrix is totally unimodular. Now, using Hochbaum and Shanthikumar (1990)'s result, we can conclude that this special case is polynomially solvable. □

A.3. The deterministic problem

In this section, we consider the problem with deterministic demand. In the deterministic case, the scenario tree is a path. As

a result, our first nonlinear formulation *MMPS-N* simplifies as follows. We redefine the decision variables; y_j is the amount of production in period $j \in \{1,\dots,T\}$ and x_{ij} is the amount of demand of period j that is satisfied from the production in period i for $i \in \{1,\dots,T\}$ and $j \in \{1,\dots,i\}$.

The deterministic problem can be formulated as follows.

$$(MMPS\text{-}N2) \quad \max \quad \sum_{i=1}^{T}\left(\Pi(y_i) - \sum_{j=1}^{i} b(s_i - s_j)\cdot x_{ij}\right)$$

$$\text{s.t.} \quad \sum_{j=1}^{i} x_{ij} = y_i \quad \forall i \in \{1,\dots,T\}, \tag{A.2}$$

$$\sum_{i=j}^{T} x_{ij} \leqslant d_j \quad \forall j \in \{1,\dots,T\}, \tag{A.3}$$

$$y_j \leqslant \tau \quad \forall j \in \{1,\dots,T\}, \tag{A.4}$$

$$x_{ij} \in \mathbb{Z}_+ \quad \forall\, i \in \{1,\dots,T\},\ j \in \{1,\dots,i\},$$

$$y_i \in \mathbb{Z}_+ \quad \forall i \in \{1,\dots,T\}.$$

We note here that our model has similarities with the transportation model of Bowman (1956) for multiperiod production planning; but in our model we do not impose to satisfy all the demand and we maximize profit rather than minimize cost.

Lemma 4.5. *The constraint matrix of formulation MMPS-N2 is totally unimodular.*

Proof. We show that given any subset of rows of the constraint matrix of formulation *MMPS-N2*, it is possible to find a partition of these rows into two sets such that the difference of row sums over these two sets is a vector with entries equal to 1, 0, or −1. Here, we ignore the nonnegativity constraints. Given a set of rows, we put the rows corresponding to constraints (A.2) into set one and the rows corresponding to constraints (A.3) and (A.4) into set two. Then the vector of differences of the sums of rows of sets one and two has entries that are equal to either 1, 0, or −1. Therefore, the matrix is totally unimodular (Schrijver, 1998). □

Theorem 4.6. *The deterministic problem is polynomially solvable.*

Proof. The objective function is a separable concave function since for i in $\{1,\dots,T\}$, $\Pi(y_i)$ is concave and $b(x)$ is convex, which leads to $-b(x)$ being concave. Moreover, by Lemma 4.5, the constraint matrix is totally unimodular. By Hochbaum and Shanthikumar (1990)'s result, the deterministic version of the problem is polynomially solvable. □

A.4. The constraint matrix of MMPS-N

In the previous sections, we proved that some special cases of the problem are polynomially solvable. We achieved these results by suggesting formulations with totally unimodular constraint matrices and concave separable objective functions. In this section,

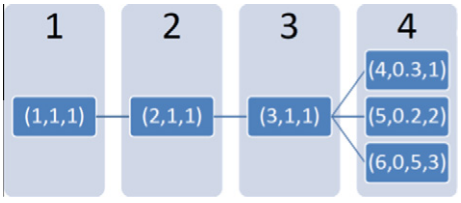

Fig. A.1. A counter-example for total unimodularity of the constraint matrix of *MMPS-N*.

Table A.1
The constraint coefficient matrix of formulation *MMPS-N* for the scenario tree in Fig. A.1.

Ctr	x_{11}	x_{21}	x_{22}	x_{31}	x_{32}	x_{33}	x_{41}	x_{43}	x_{52}	x_{53}	x_{61}	x_{62}	x_{42}	x_{44}	x_{51}	x_{55}	x_{63}	x_{66}	y_1	y_2	y_3	y_4	y_5	y_6	z_1	z_2	z_3	z_4	z_5	z_6
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	−1	0	0	0	0	0	0	0	0	0	0	0
2	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	−1	0	0	0	0	0	0	0	0	0	0
3	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	−1	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	1	1	0	0	0	0	1	1	0	0	0	0	0	0	0	−1	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	1	0	0	0	0	0	0	−1	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	1	0	0	0	0	0	−1	0	0	0	0	0	0
7	1	1	0	1	0	0	1	0	0	0	1	0	0	0	0	0	0	0	−1	0	0	0	0	0	−1	0	−1	0	0	0
8	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	−1	0	0	0	0	0	−1	0	−1	0	0	0
9	0	0	1	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	−1	0	0	0	0
10	0	0	1	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	−1	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	−1	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	−1	0	0
13	0	0	1	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	−1	0
14	1	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	−1	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	−1
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

we show that the constraint matrix of *MMPS-N* may not be totally unimodular.

Example 4. Consider the scenario tree given in Fig. A.1. The constraint coefficient matrix of formulation *MMPS-N* for this scenario tree is given in Table A.1.

This matrix has at least one submatrix with a determinant different than 1, 0 or −1. For instance, the submatrix consisting of the intersection of rows four to 12 and columns four to 12 in Table A.1 has a determinant of 2. Therefore, the constraint coefficient matrix is not totally unimodular. Consequently, the computational complexity of the general stochastic problem with postponement costs is an open question.

References

Ahmed, S., King, A.J., Gyana, P., 2003. A multi-stage stochastic integer programming approach for capacity expansion under uncertainty. Journal of Global Optimization 26, 3–24.

Atamturk, A., Zhang, M., 2007. Two-stage robust network flow and design under demand uncertainty. Operations Research 55, 662–673.

Balibek, E., Koksalan, M., 2010. A multi-objective multi-period stochastic programming model for public debt management. European Journal of Operational Research 205, 205–217.

Birge, J.R., Louveaux, J., 1997. Introduction to Stochastic Programming. Springer.

Bookbinder, J.H., Tan, J.Y., 1988. Strategies for the probabilistic lot sizing problem with service level constraints. Management Science 34, 1096–1108.

Bowman, E.H., 1956. Production scheduling by the transportation method of linear programming. Operations Research 3, 100–103.

Charnes, A., Cooper, W.W., Symonds, G.H., 1958. Cost horizons and certainty equivalents: An approach to stochastic programming of heating oil. Management Science 4, 235–263.

Cheng, T.C.E., Kovalyov, M.Y., Shackhlevich, N.V., 2006. Scheduling with controllable release dates and processing times: Total completion time minimization. European Journal of Operational Research 175, 769–781.

Dyer, M., Leen, S., 2006. Computational complexity of stochastic programming problems. Mathematical Programming 106, 423–432.

Escudero, L.F., Kamesan, P.V., King, A.J., Wets, J.B., 1993. Production planning via scenario modelling. Annals of Operations Research 43, 311–335.

Fulkerson, D.R., Gross, O.A., 1965. Incidence matrices and interval graphs. Pacific Journal of Mathematics 15, 835–865.

Guan, Y., Ahmed, S., Nemhauser, G.L., Miller, A.J., 2006. A branch-and-cut algorithm for the stochastic uncapacitated lot-sizing problem. Mathematical Programming 105, 55–84.

Gurel, S., Akturk, M.S., 2007. Optimal allocation and processing time decisions on non-identical parallel CNC machines: ϵ-constraint approach. European Journal of Operational Research 183, 591–607.

Higle, J.L., Kempf, K.G., 2011. Production planning under supply and demand uncertainty: A stochastic programming approach. In: Infanger, G. (Ed.), Stochastic Programming: The State of the Art. Springer, Berlin, pp. 297–315.

Hochbaum, D.S., Shanthikumar, J.G., 1990. Convex optimization is not much harder than linear optimization. Journal of the Association for Computing Machinery 37, 843–862.

Holt, C.C., Modigliani, F., Muth, J.F., 1956. Derivation of a linear decision rule for production and employment. Management Science 2, 159–177.

Huang, K., Ahmed, S., 2009. The value of multistage stochastic programming in capacity planning under uncertainty. Operations Research 57, 893–904.

Karabuk, S., 2008. Production planning under uncertainty in textile manufacturing. Journal of the Operations Research Society 59, 510–520.

Kayan, R.K., Akturk, M.S., 2005. A new bounding mechanism for the CNC machine scheduling problems with controllable processing times. European Journal of Operational Research 167, 624–643.

Leyvand, Y., Shabtay, D., Steiner, G., 2010. A unified approach for scheduling with convex resource consumption functions using positional penalties. European Journal of Operational Research 206, 301–312.

Orcun, S., Uzsoy, R., Kempf, K.G., 2009. An integrated production planning model with load-dependent lead times and safety stocks. Computers and Chemical Engineering 33, 2159–2163.

Peters, R.J., Boskma, K., Kupper, H.A.E., 1977. Stochastic programming in production planning: A case with non-simple recourse. Statistica Neerlandica 31, 113–126.

Schrijver, A., 1998. Theory of Linear and Integer Programming. Wiley.

Schultz, R., Stougie, L., Vlerk, M.H., 1996. Two-stage stochastic integer programming: A survey. Statistica Neerlandica 50, 404–416.

Shabtay, D., Steiner, G., 2007. A survey of scheduling with controllable processing times. Discrete Applied Mathematics 155, 1643–1666.

Sridharan, V., Barry, W.L., Udayabhanu, V., 1987. Freezing the master production schedule under rolling planning horizons. Management Science 33, 1137–1149.

Tang, O., Grubbström, R.W., 2002. Planning and replanning the master production schedule under demand uncertainty. International Journal of Production Economics 78, 323–334.

Vieira, G.E., 2006. A practical view of the complexity in developing master production schedules: Fundamentals, examples and implementation. In: Herrman, J.W. (Ed.), Handbook of Production Scheduling, vol. 89. Springer, Berlin, Hiedelberg, pp. 149–176.

Voss, S., Woodruff, D.L., 2006. Introduction to Computational Optimization Models for Production Planning in a Supply Chain, second ed. Springer, Berlin, New York.