A MULTI-STAGE STOCHASTIC PROGRAMMING APPROACH IN MASTER PRODUCTION SCHEDULING

ERSİN KÖRPEOĞLU, HANDE YAMAN, AND M. SELİM AKTÜRK

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 questioned these assumptions and considered a problem with finite capacity, controllable processing times, and several demand scenarios instead of just one. We used a multi-stage stochastic programming approach in order to come up with maximum expected profit given the demand scenarios. Controllable processing times enlarges the solution space so that the limited capacity of production resources are utilized more effectively. We proposed an effective formulation which enabled an extensive computational study. Our computational results clearly indicate that MPS problems could be solved to optimality using multi-stage stochastic programming approach instead of relying on relatively simple heuristic methods, and controllability increases the performance of multi-stage solutions.

1. INTRODUCTION

Master Production Schedules (MPS) are widely used by manufacturing facilities to handle the production and scheduling decisions. In the current industry practice, MPS produces the production schedules in a finite planning horizon assuming infinite capacity, fixed processing times, and deterministic demand. As an example, the largest auto manufacturer in Turkey recently introduced a new multi-purpose vehicle to the market. The company installed a new single production line with a limited production capacity and dedicated it to this particular

Key words and phrases. Stochastic programming; Master production scheduling; Flexible manufacturing; Controllable processing times.

E. Körpeoğlu: Tepper School of Business, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, USA (ekorpeog@andrew.cmu.edu)

M. S. Aktürk, H. Yaman: Department of Industrial Engineering, Bilkent University, 06800 Bilkent, Ankara, Turkey (akturk@bilkent.edu.tr, hyaman@bilkent.edu.tr) .

model. Since the production facilities are flexible, the processing times can be altered or controlled (albeit at higher manufacturing cost) by changing the machining conditions in response to the demand changes. As this model is new, the company generated different demand scenarios for each time period. One of the important planning problems is to develop a master production schedule to decide on how many units of this new model will 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 ERP system as an important input to the materials management module to explode the component requirements and to 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 which has 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 on infinite capacity, fixed processing times, and deterministic demand, and to propose a new approach which 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.

1.1. **Master Production Scheduling.**

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 the production scheduling decisions are not altered even when the demand realizations turn out to be different than the estimates. The longer frozen time period will be less responsive to the demand changes, but creates less nervousness, while the shorter one will act oppositely. The studies by Sridharan et al. (1987) and Tang and Grubbström (2002) are good examples that consider the effect of the length of the frozen zone on production and inventory costs. Based upon his industry practice, Vieira (2006) pointed 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.

1.2. **Controllable Processing Times.**

There are several instruments that can be used to control the 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 you increase the cutting speed and/or the feed rate, the processing time of the operation is compressed at an additional cost that arises due to increased tooling costs as discussed in Gurel and Akturk (2007). This 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. (????) 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 this study, we define the compression cost function $f(y) = \kappa \cdot y^{a/b}$ 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.

A review of scheduling with controllable processing times can be found in Shabtay and Steiner (2007). 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 amount of cost. Thus, it brings up the trade-off between the revenue gained by satisfying an additional demand and the amount of compression cost. The value of the controllable processing times becomes even more evident during the current economic crisis, since it allows companies to adjust their production quantities more effectively to meet the immediate demand that varies significantly during the planning horizon.

1.3. **Multi-stage Stochastic Programming.** 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 the 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 affecting the outcome of the first-stage decision. A recourse decision can then be made in the second stage to compensate for any bad 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). Shmoys and Sozio (2007) apply two-stage stochastic programming to single machine scheduling and propose approximation algorithms. Using more than one stage in decision making is also utilized in robust optimization. Atamtürk and Zhang (2007) apply two-stage robust optimization to network flow and design problems. They give a numerical example which 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 more future data as they become available, hence enables decision making in a less uncertain environment. On the other hand, they are generally more difficult to solve than their two-stage counterparts. Therefore, their applications are rare compared to two-stage stochastic programming. Karabuk (2008) applied multi-stage stochastic programming to production planning in textile manufacturing, whereas Balibek and Koksalan (2010) applied a multi-objective multi-stage stochastic programming approach for the public debt management problem. Guan et al. (2006) studied the uncapacitated lot-sizing problem and Ahmed et al. (2003) studied the capacity expansion problem with uncertain demand and cost parameters. Huang and Ahmed (2009) provided analytical bounds for the value of multistage 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 which applies multi-stage stochastic programming to master production scheduling.

Stochastic programming problems are generally considered to be difficult problems (Dyer and Leen, 2006). However, in this paper, we provide a formulation for our problem which is solved efficiently for large size problems.

When the uncertain parameters evolve as a discrete time stochastic process with finite probability space, the uncertainty can be represented with a scenario tree. Figure 1 depicts an example of a scenario tree. The nodes of the tree represent demand scenarios for periods. For each node, we give in parenthesis, 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

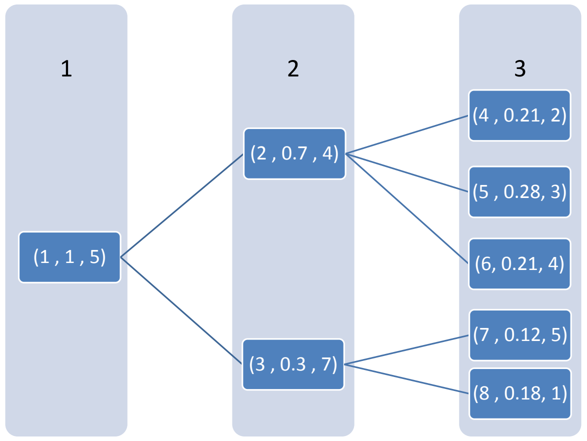

FIGURE 1. A scenario tree for three periods

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 which 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 the information on the demand of each period becomes available at the beginning of the period, our decision stages correspond to periods.

The benefits of using a multi-stage stochastic programming approach instead of using fixed demand estimates as in the case of the classical MPS is illustrated in the following example.

Example 1. Consider the scenario tree in Figure 1. In 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, maximum compression amount is 4 time units, and the capacity is 36 time units. For simplicity, we assume that the postponement and the shortage costs are zero. The cost that is 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 effect the overall results.

In Table 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 the second best performance in all scenarios and has the maximum expected profit.

Table 1. Maximum profits of different strategies.

		Possible strategies				
Realized scenario	Prob	Pessimistic	Most likely	Optimistic	Expected demand	Multi-stage
1-2-4	0.21	628.4	568.4 - ξ	485.8 - 6ξ	588.4 - 2ξ	604.2
1-2-5	0.28	628.4	672.6	545.8 - 5ξ	648.4 - ξ	664.2
1-2-6	0.21	628.4	672.6	605.8 - 4ξ	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 - 4ξ	708.4	672.9
Expected	profit	628.4	650.7 - 0.2ξ	592.6 - 4.2ξ	666.4 - 0.7ξ	684.2

Another possible measure that can be used to evaluate the performance of strategies is the 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 relatively small $\xi = 10$. The results are given in Table 2.

Here we see that the relative regret of the multi-stage stochastic programming solution is very small compared to the ones 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%.

Table 2. Relative regrets of different strategies.

		Possible strategies				
Realized scenario	Prob	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

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 the performance of the schedules. 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. This 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 give the notation and a nonlinear and a linear integer programming formulation. Then we study two subproblems and use the outcomes of this study to derive an alternative linear integer programming formulation, which turns out to be quite efficient. In section 3, we introduce two trivial cases and analyze two special cases of the problem that are polynomially solvable. In section 4, 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 5.

2. THE MULTI-STAGE STOCHASTIC PROGRAMMING MODELS

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

In this section, we first give a nonlinear formulation that decides on 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.

Afterwards, 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 on 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 $t = 1, 2, ..., T$. 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 realization of 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 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, 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 multistage master production scheduling and abbreviate it with *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.

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

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

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

$$z_j \le d_j \qquad \forall j \in N \tag{2.4}$$

$$y_j \le k_{max} \qquad \forall j \in N \tag{2.5}$$

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

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

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

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 which 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 $k \in \{0, 1, ..., k_{max}\}$. 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 these in the above formulation, and adding the constraints that ensure that for each node $i \in N$, exactly one k value in $\{0, 1, ..., k_{max}\}$ is picked as the production amount, we obtain the following linear integer programming formulation, that is referred to as *MMPS-L1*.

$$
\begin{aligned}
(\textit{MMPS-L1}) \qquad & \max \quad \sum_{i \in N} \gamma_i \cdot \left(\sum_{k=0}^{k_{max}} \Pi(k) \cdot w_{ik} - \sum_{j \in B_i} b(s_i - s_j) \cdot x_{ij} \right) \\
& s.t. \quad (2.3), (2.4), (2.6), (2.7) \\
& \qquad \sum_{k=0}^{k_{max}} w_{ik} = 1 \qquad \forall i \in N \\
& \qquad \sum_{j \in B_i} x_{ij} = \sum_{k=0}^{k_{max}} k \cdot w_{ik} \qquad \forall i \in N \\
& \qquad w_{ik} \in \{0, 1\} \qquad \forall i \in N,\ k \in \{0, 1, ..., k_{max}\}.
\end{aligned}
$$

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 $\{0, 1, ..., k_{max}\}$ 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 sub-problems which are used to reduce the possible number of jobs that are 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 Cost Minimization Objective.

In this section, we introduce and study our first sub-problem, which is the single period capacitated deterministic scheduling problem with 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 that has the minimum compression cost is p and the maximum compression amount is u. There are $n \leq k_{max}$ jobs in the work center. The compression cost function is $f : \mathbb{R}_+ \to \mathbb{R}_+$ and is a strictly convex function. 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 $\{1, ..., n\}$. Now, this problem can be formulated as follows.

$$\begin{aligned} \min \quad & \sum_{j=1}^{n} f(c_j) && (2.9) \\ s.t. \quad & c_j \leq u \qquad \forall j \in \{1, ..., n\} && (2.10) \\ & \sum_{j=1}^{n} (p - c_j) \leq C && (2.11) \\ & c_j \in \mathbb{R}_+ \quad \forall j \in \{1, ..., n\}. && (2.12) \end{aligned}$$

The objective function (2.9) is equal to the sum of 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 $n \cdot (p - u) \leq C$. 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 $j = 1, ..., n$ 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 $\bar{c_i} = \bar{c_j} = \frac{c_i+c_j}{2}$. The solution $\bar{c}$ is feasible and by strict convexity of the cost function, $f(\bar{c_i})+f(\bar{c_j}) < f(c_i)+f(c_j)$. This contradicts the optimality of the initial solution c. Now, it follows immediately that $c_j = \max\{p - \frac{C}{n}, 0\}$ for all $j = 1, ..., n$ is an optimal solution. □

Proposition 2.1 is intuitive. As the compression cost function is a strictly convex function, the more the compression amount is, the more the marginal compression cost that 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 $\Pi(x) = \begin{cases} x \cdot h - x \cdot \kappa \cdot (p - \frac{C}{x})^{\frac{a}{b}} & \text{if } x > \frac{C}{p}, \\ x \cdot h & \text{otherwise.} \end{cases}$

Corollary 2.2. *Let n be a positive integer with $n \cdot (p - u) \leq C$. 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 Profit Maximization Objective.** In this section, we study the second sub-problem and use the results to reduce the size of formulation *MMPS-L1.* For this sub-problem, 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:

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

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* $(\frac{C}{p}, +\infty)$ *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:

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

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:

$$\frac{d^2\Pi^c}{dx^2}(x) = -\frac{a}{b} \cdot \kappa \cdot (p - \frac{C}{x})^{\frac{a}{b}-1} \cdot \frac{C}{x^2} - \frac{a}{b} \cdot (\frac{a}{b} - 1) \cdot \kappa \cdot (p - \frac{C}{x})^{\frac{a}{b}-2} \cdot \frac{C^2}{x^3} + \frac{a}{b} \cdot \kappa \cdot (p - \frac{C}{x})^{\frac{a}{b}-1} \cdot \frac{C}{x^2}$$

$$= -\frac{a}{b} \cdot (\frac{a}{b} - 1) \cdot \kappa \cdot (p - \frac{C}{x})^{\frac{a}{b}-2} \cdot \frac{C^2}{x^3} \leq 0$$

since $a > b$ and $x \geq \frac{C}{p}$, $\frac{d\Pi^c}{dx}(x)$ is monotonically decreasing. Moreover, h is monotonically non-increasing. 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 $(\frac{C}{p}, +\infty)$ such that $\frac{d\Pi^c}{dx}(x^*) = 0$. Since $\Pi(x)$ has the same values as $\Pi^c(x)$ on the domain $(\frac{C}{p}, +\infty)$, 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,*

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

On the other hand, if $h \ge \kappa \cdot p^{\frac{a}{b}}$, 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 $v_{ik} \geq v_{i(k+1)}$ for all $k \in \{1, \ldots, \tau - 1\}$. This formulation, named as *MMPS-L2*, is as follows.

$$
\begin{aligned}
(\textit{MMPS-L2}) \qquad & \max \quad \sum_{i \in N} \gamma_i \cdot \left(\sum_{k=1}^{\tau} (\Pi(k) - \Pi(k-1)) \cdot v_{ik} - \sum_{j \in B_i} b(s_i - s_j) \cdot x_{ij} \right) \\
& s.t. \quad (2.3),\ (2.4),\ (2.6),\ (2.7) \\
& \qquad v_{ik} \geq v_{i(k+1)} \qquad \forall i \in N,\ k \in \{1, \ldots, \tau - 1\} \qquad (2.13) \\
& \qquad \sum_{j \in B_i} x_{ij} = \sum_{k=1}^{\tau} v_{ik} \qquad \forall i \in N \\
& \qquad v_{ik} \in \{0, 1\} \qquad \forall i \in N,\ k \in \{1, \ldots, \tau\}.
\end{aligned}
$$

Constraints (2.13) 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.

In our computational study, we compare the solution times for formulations *MMPS-L1* and *MMPS-L3* and conclude that *MMPS-L3* outperforms *MMPS-L1*.

3. 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.

3.1. **Sufficient Conditions for Optimality.** In this section, we introduce some trivial cases.

Suppose that the demand at each node is no more than the threshold value and equal to each other. Then by Proposition 3.1, 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 3.1. *Suppose that, $d_i = d \leq \tau$ 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 of the nodes. A sufficient condition for optimality in this case is given in Proposition 3.2.

Proposition 3.2. *Suppose that $d_i \geq \tau$ for all $i \in N$. Then the solution where τ jobs are produced at all nodes is optimal.*

3.2. **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.

$$
\begin{aligned}
(\textit{MMPS-N1}) \qquad\qquad & \max \sum_{i \in N} \gamma_i \cdot \Pi(y_i) \\
& s.t. \sum_{j \in P_{1i}} y_j \leq \sum_{j \in P_{1i}} d_j \qquad \forall i \in N \qquad\qquad (3.1) \\
& y_i \leq \tau \qquad \forall i \in N \\
& y_i \in \mathbb{Z}_+ \qquad \forall\, i \in N.
\end{aligned}
$$

We call this formulation *MMPS-N1*. Here constraints (3.1) 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 determinant equal to 0, 1, or -1. Hochbaum and Shantikumar (1990) have shown 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 3.3. *The constraint matrix of MMPS-N1 is totally unimodular.*

Proof. The constraint matrix consists of three sub-matrices; the first sub-matrix consists of the coefficients of constraints (3.1), the second sub-matrix consists of the coefficients of the upper bound constraints, and the third sub-matrix consists of the coefficients of the nonnegativity constraints. The last two sub-matrices are identity matrices. Therefore, it is sufficient to prove that sub-matrix one is totally unimodular. Each row of sub-matrix 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 sub-matrix satisfies the consecutive 1's property and thus is totally unimodular (Fulkerson and Gross, 1965). □

Theorem 3.4. *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 3.3, the constraint matrix is totally unimodular. Now using the result of Hochbaum and Shantikumar (1990), we can conclude that this special case is polynomially solvable. □

3.3. **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.

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

$$s.t. \quad \sum_{j=1}^{i} x_{ij} = y_i \qquad \forall i \in \{1, ..., T\} \tag{3.2}$$

$$\sum_{i=j}^{T} x_{ij} \le d_j \qquad \forall j \in \{1, ..., T\} \tag{3.3}$$

$$y_j \le \tau \qquad \forall j \in \{1, ..., T\} \tag{3.4}$$

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

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

Lemma 3.5. *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 (3.2) into set one and the rows corresponding to constraints (3.3) and (3.4) 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 3.6. *The deterministic problem is polynomially solvable.*

Proof. The objective function is a separable concave function since for i in $\{1, \ldots, T\}$, $\Pi(y_i)$ is concave and $b(x)$ is convex which leads to -$b(x)$ being concave. Moreover, by Lemma 3.5, the constraint matrix is totally unimodular. By the result of Hochbaum and Shantikumar (1990), the deterministic version of the problem is polynomially solvable. □

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

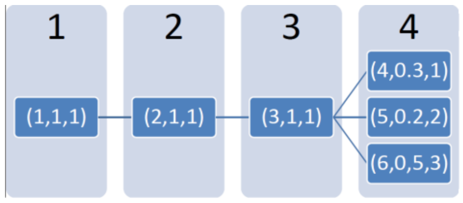

FIGURE 2. A counter example for totally unimodularity of constraint matrix of *MMPS-N*

Example 2. Consider the scenario tree given in Figure 2. The constraint coefficient matrix of formulation *MMPS-N* for this scenario tree is given in Table 3.

Table 3. The constraint coefficient matrix of formulation *MMPS-N* for the scenario tree in Figure 2

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**	**0**	**0**	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0
8	1	1	0	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0
9	0	0	1	**0**	**1**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	0	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**	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0
11	0	0	0	**0**	**0**	**1**	**0**	**1**	**0**	**0**	**0**	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0
12	0	0	0	**0**	**0**	**1**	**0**	**0**	**0**	**1**	**0**	**0**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	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	-1	0	0	0	0
14	1	1	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0
15	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	-1	0	0	0
16	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	0	-1	0	0
17	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
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	-1

This matrix has at least one sub-matrix with determinant different that 1,0 or -1. For instance, the sub-matrix consisting of the intersection of rows 4 to 12 and columns 4 to 12 in Table 3 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.

4. 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 of 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.

4.1. **The Design of Experiments.** 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 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 tightness of capacities, 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 with single scenario strategies. Table 4 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 Ghz Intel Xeon CPU and 8GB 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 Figure 3.a, we depict the profit function for $a/b = 3$ as an example. In this case, $\tau = 3 < k_{max} = 10$. The profit function for $a/b = 2$ is given in Figure 3.b 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 $\tau = k_{max}$.

TABLE 4. 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	-	-

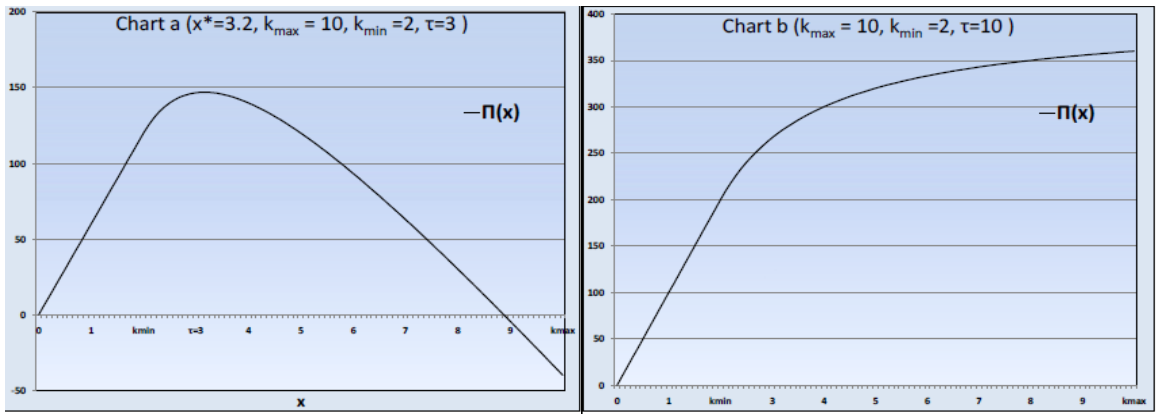

FIGURE 3. Profit functions for $a/b = 3$ and $a/b = 2$ ($k_{min} = \lfloor \frac{C}{p} \rfloor$).

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 alternatives with the same mean but different variances.

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 0.5μ.

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 $t \geq 0$, 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 the future periods for a demand surge that may or may not happen in the future. One of the important advantages of having controllable processing times, the capacity is no longer fixed and it can be adjusted with respect to the immediate demand information, which is more beneficial than carrying inventories. Therefore, we do not consider carrying the inventory as an alternative in our model, instead we 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 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 linear inventory holding cost function).

4.2. **Computation times.** In our first experiment, we investigate the performances of our formulations *MMPS-L1* and *MMPS-L3* in terms of CPU times on 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 postponement cost ratio (factor C) in order to test the case where 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 in total.

Formulation *MMPS-L3* solves all the problem instances within at most 4 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 ten 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 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.

4.3. **Comparison of 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 solution 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. As the single scenario strategies ignore the uncertainty in the demand, we use two different adjustment policies to improve their solutions.

- T periods frozen policy (TPF): In this policy, the 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 whatever the realized demand scenario is, 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. These production amounts are adjusted

myopically if the estimated scenario is not realized. Myopic adjustment works 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 the production amounts are more sensitive to the immediate demand realizations.

We compare the solution quality of the single scenario strategies coupled with these two different production policies (resulting in a total of 8 different strategy - policy combinations, e.g., ML-TPF denotes choosing the most likely (ML) strategy with the T periods frozen (TPF) policy) with the multi-stage stochastic programming (MSP) solution. We use relative regret as the metric of comparison for two reasons. First, it measures how bad our solutions perform compared to an optimal solution that we would have if we had perfect knowledge about the input parameters and hence it gives an insight about the performance of our methods to hedge against uncertainty. Second, the profit values may differ significantly among different settings which may cause settings with higher profit values to dominate the results. Therefore, relative regret provides a scaled measure to compare the performance under different settings.

As discussed above, there are 1920 randomly generated scenario trees. Since the single scenario strategies are deterministic strategies, 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 production amounts and the 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 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 (PES), 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 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, production of 4 units takes place but a demand of 7 is realized instead of 4. This is compensated 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 policy TPF 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.

Similar to inventory, some excess production or shortage may occur within the planning horizon if the total demand of the realized scenario is strictly less or more than the demand of the estimated scenario. We first 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 8 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. Figure 4 displays a cross-section from this three dimensional 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 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 increase in shortage and excess costs, we encountered a similar result: MSP always outperforms other strategies and as the excess/shortage cost increases, the gap between MSP and the other strategies increases as well. 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 remaining of the analysis, noting that all of the results which we will present below can be extended to the case with non-zero excess and shortage costs case as well.

Before going into the detailed analysis of the significance of capacity, let us begin with a general analysis of our results. Table 5 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 whatever the adjustment policy is. Moreover, MSP has a significant dominance in terms of number of minimum regret values. The difference between the regret of MSP and other strategies is significantly high (going up to

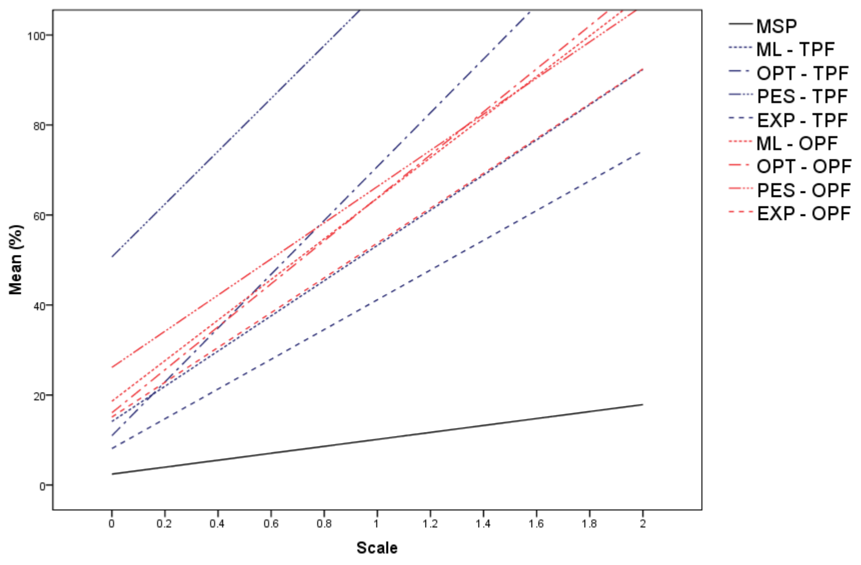

FIGURE 4. Cross section of the regret function where shortage and excess are equal

TABLE 5. Average relative regret and number of times minimum values of each scenario selection strategy.

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

* The multi-stage solution is not dependant on policy changes.

50%). Moreover, MSP dominates other strategy-policy combinations in terms of the number of times it achieves the minimum regret values (going up to 99.5%). Another interesting conclusion is that the available capacity significantly affects the overall results. This actually points out that assuming infinite capacity in master production scheduling is quite unreasonable since solutions are very sensitive to changes in capacity.

Table 6 gives the statistics and the 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 strategies whatever adjustment policy is used. 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.

Table 6. 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

We do not have enough space to discuss the effects of all seven factors here. Therefore we summarize our findings on 6 factors and we give our results concerning the capacity factor in more detail as it plays an important role in questioning one of the basic assumptions of the traditional MPS: the infinite capacity assumption. Figures 5 and 6 illustrate the average relative regret values for each strategy - policy combination and 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 once and hence make the figures easier to interpret. In Figure 5, the pessimistic strategy combined with TPF policy is not illustrated since it is way out of scale (its regret values are around 80%). As it is clear in both figures, the average regret performance of multi-stage stochastic programming is significantly better than other strategy - policy combinations in all of the settings.

In order to find out whether a factor significantly affects the solution performance of strategies, we first conducted one way analysis of variance (ANOVA) test. The ANOVA results showed that all the factors that we parameterized are significant for all strategy-policy combinations and multi-stage stochastic programming solution. Although we have

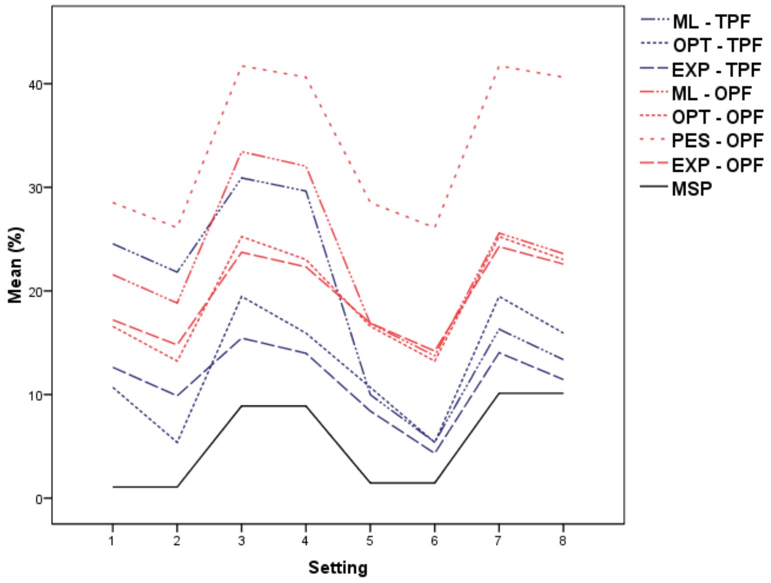

FIGURE 5. Average regrets of strategy policy combinations and multi-stage for combinations of factors A, C, and D.

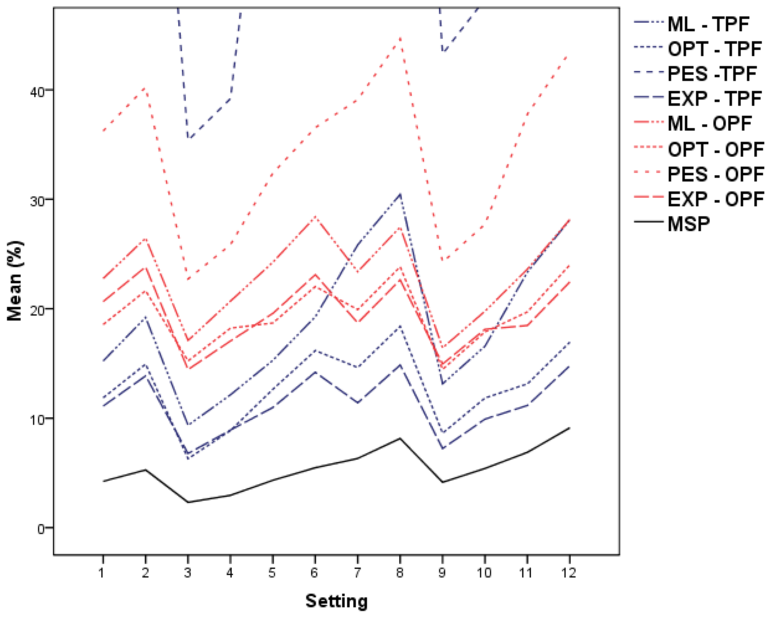

FIGURE 6. Average regrets of strategy policy combinations and multi-stage for combinations of factors E, F, and G.

made a thorough analysis of the effects all factors, due to limited space and to focus on our main purpose, we will not go into detail in those and we will focus on the significance of capacity in the remaining part of this section. The tables of the other factors can be obtained from the first author. The ANOVA results for the capacity factor are given in Table 7. As the table suggests, the capacity factor is significant not only for MSP, but also for other strategy policy combinations. Besides, confidence intervals also support this claim since they do not generally intersect for different capacity levels. The effect of capacity on the solution performance of MSP is evident and MSP gives relatively better solutions if the capacity is looser compared to the case where it is very tight. This is quite expected because the tighter capacity results in postponement of more jobs thus MSP looses its flexibility to adjust to changes in demand.

TABLE 7. A summary of ANOVA results for capacity factor ζ.

			CI for $\zeta = 0.2$		CI for $\zeta = 0.4$		CI for $\zeta = 0.6$		CI for $\zeta = 0.8$	
	F	Significance	LB	UB	LB	UB	LB	UB	LB	UB
ML - TPF	63.0	0.000	14.6	15.5	15.7	16.7	12.7	13.6	11.7	12.5
OPT - TPF	627.8	0.000	10.8	11.4	15.1	16.0	11.0	11.6	5.7	6.0
PES - TPF	106.3	0.000	45.9	47.2	54.3	55.5	51.1	52.5	48.8	50.3
EXP - TPF	142.2	0.000	9.9	10.5	8.1	8.7	6.3	6.7	7.1	7.6
ML - OPF	23.5	0.000	16.9	17.7	18.8	19.5	18.4	19.1	18.9	19.6
OPT - OPF	45.0	0.000	14.2	14.8	16.5	17.1	16.1	16.7	16.2	16.9
PES - OPF	41.6	0.000	26.1	27.0	27.3	28.2	25.5	26.3	24.0	24.8
EXP - OPF	14.0	0.000	14.4	14.9	14.6	15.2	14.6	15.2	15.6	16.3
MSP - OPF	2386.0	0.000	8.3	8.9	0.9	1.1	0.1	0.1	0.0	0.0

To analyze the effects of the factors on the difference between the solution performances of multi-stage stochastic programming and other strategies, we use paired sample t-tests. In Table 8, 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 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 is minimum at very low and very high capacity values. EXP has another interesting structure because it has a decreasing regret as capacity becomes looser but the

TABLE 8. Pairwise statistics of regret differences for different levels of 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

regret starts to increase as the capacity is very loose. For MSP, as capacity decreases, the regret increases. This is mainly due to the fact that as capacity becomes smaller, the number of jobs that is produced at their own nodes decreases leading to an increase in error margin. In OPF policy, the significance of capacity for the first four strategies decreases although capacity still affects the performance of solutions. In general, all of the strategies are quite sensitive to the available capacity, thus it is essential to revise the unrealistic infinite capacity assumption of the existing MPS algorithms.

4.4. **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 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 the ones that we used in the previous experiment.

Table 9 summarizes the results of this study. We use three different performance metrics to compare MSPC and MSPF. The first performance metric compares expected profit values of MSPC and MSPF. Calculation of the scenario based metric is the same as the one that is used in Section 4.3. We randomly generate 10 scenarios and compare the profit of MSPC and MSPF when these 10 scenarios are realized. In the first two metrics we take shortage cost as zero. In the third metric, we compare the amount of shortage incurred when the

Table 9. Comparison of multi stage solution with and without controllability.

		Expected profit		Scenario based		Shortage cost	
Processing times	Capacity	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

randomly generated scenarios that are 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 9, 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 around 23%. The third and fourth columns in the table show the average profit and number of times best values when scenario based metric is used. Again, controllability improves the solutions around 27% on the average and MSPC finds a solution that is at least as good as MSPF in 99.8% of 19200 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 around 80% on the average and always gives a shortage value at least as good as the other one. Thus, controllability has a very significant effect on the solution performance of multi-stage stochastic programming.

Another observation that we make based on the results in Table 9 is on the significance of capacity; capacity drastically affects the improvement provided by controllability. If the capacity is tight, controllability provides an improvement up to 80%, but this improvement decreases to around 5-10% when the capacity is loose. This result is intuitive since controllability provides flexibility in capacity and as capacity gets tighter, this flexibility becomes more and more critical.

4.5. **Concluding Remarks on Computational Results.** In Section 4.2, we tested the performance of our formulations and formulation *MMPS-L3* proved to be very efficient in terms of CPU performance. We exploited this fact during our experiments in Sections 4.3 and 4.4 taking approximately 250,000 runs in the experiment of Section 4.3 and 40,000 runs in the experiment of Section 4.4.

In Section 4.3, we compared the multi-stage stochastic programming approach with four other strategies coupled with two adjustment policies. We have shown that capacity significantly affected solution quality of strategies as well as pairwise differences between the single scenario strategies and the multi-stage stochastic programming approach. We also showed that MSP gave statistically significantly better results than other strategies in all factor combinations and policies. We also pointed out that MSP generally gave one of the best solutions in the number of times minimum analysis. Therefore, our computational results suggest that using a multi-stage stochastic programming approach improves the solution performance significantly compared to single scenario strategies which are used in current industry practice.

In Section 4.4, we tested the effect of controllability on the profit performance of multi-stage. The computational results showed that controllability provided a huge improvement which was evident due to three different criteria: Expected profit, the performance when the comparison technique of Section 4.3 is utilized, and finally, in terms of shortage values. Also, capacity was found to be a significant factor which critically affects the improvement that controllability provided. Therefore, our computational results suggest that utilizing finite capacity and controllability significantly improve solution performances especially in environments with tight capacity.

Based on the results obtained in Sections 4.3 and 4.4, we can conclude that using a capacitated version of MPS with controllability and considering several scenarios for demand realizations provide drastic improvements in the solution performance of MPS. Moreover, the fact that *MMPS-L3* solves even large instances in a few seconds opens up new application possibilities of our approach such as establishing it in ERP software or conducting what if and sensitivity analysis. It is also possible to extend the analysis that we made in the computational study by adding new factors or considering new strategies and policies, even firm based strategies and policies.

5. CONCLUSION

The existing algorithms in the literature to solve the MPS problem are generally based on the limiting assumptions of infinite capacity, fixed processing times, and fixed and known demand realizations, although there might be very few applications where these assumptions can be justified. In this paper, we questioned these assumptions and came up with a model with finite capacity, controllable processing times, and uncertain demand values. We used multi-stage stochastic programming to handle this uncertainty and proposed a very effective formulation which solves large instances in a very short computation time. The fact that the formulation solves considerably large instances at a maximum of four seconds, enabled us to conduct an extensive computational study. Our computational 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 both 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 huge improvement with a very 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 new APS systems rely on relatively simple heuristic methods (such as capable-to-promise, etc.) to solve the MPS problems. Our computational results clearly indicate that MPS problems could be solved to optimality using multi-stage stochastic programming approach in short computation times.

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 (1), 3–24.

Atamtürk, 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 (1), 215–217.

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

Cheng, T., Kovalyov, M., Shakhlevich, N., 2006. Scheduling with controllable release dates and processing times: Total completion time minimization. European Journal of Operational Research 175 (2), 769–781.

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

Fulkerson, D., Gross, O., 1965. Incidence matrices and interval graphs. Pasific Journal of Mathematics 15 (3), 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 (1), 55–84.

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

Hochbaum, D. S., Shantikumar, J. G., 1990. Convex optimization is not much harder than linear optimization. Journal of the Association for Computing Machinery 37 (4), 843–862.

Huang, K., Ahmed, S., 2009. The value of multistage stochastic programming in capacity planning under uncertainty. Operations Research 57 (4), 893–904.

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

Leyvand, Y., Shabtay, D., Steiner, G., ???? A unified approach for scheduling with convex resource consumption functions using positional penalties. European Journal of Operational ResearchTo appear, doi:10.1016/j.ejor.2010.02.026.

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 (13), 1643–1666.

Shmoys, D. B., Sozio, M., 2007. Approximation algorithms for 2-stage stochastic scheduling problems. In: Integer Programming and Combinatorial Optimization. Springer Berlin / Heidelberg.

Sridharan, V., Barry, W. L., Udayabhanu, V., 1987. Freezing the master production schedule under rolling planning horizons. Management Science 33 (9), 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: Handbook of Production Scheduling. Springer Berlin / Heidelberg.