International Journal of Production Research

ISSN: 0020-7543 (Print) 1366-588X (Online) Journal homepage: https://www.tandfonline.com/loi/tprs20

Selection of an optimal subset of sizes

Soheila Jorjani , Carlton H. Scott & David L. Woodruff

To cite this article: Soheila Jorjani , Carlton H. Scott & David L. Woodruff (1999) Selection of an optimal subset of sizes, International Journal of Production Research, 37:16, 3697-3710, DOI: 10.1080/002075499189998

To link to this article: https://doi.org/10.1080/002075499189998

INT. J. PROD. RES., 1999, VOL. 37, NO. 16, 3697–3710

Selection of an optimal subset of sizes

SOHEILA JORJANI†, CARLTON H. SCOTT‡ and DAVID L. WOODRUFF§*

For a company that produces a product in a range of sizes, it is sometimes possible to meet demand for a smaller size by substituting a larger size. In this paper, we give an integer linear programming model that addresses this issue. Various characteristics of the optimal policy are given. These properties are exploited when the formulation is extended to the multi-period stochastic demand case. This application was motivated by our experience with a company that manufactures a range of multiple-piece blind fasteners where savings in setup cost can be made using long-grip fasteners to meet the demand for short-grip fasteners.

1. Introduction

Many manufactured products come in a variety of sizes to meet the specific demands of an application. In some circumstances, it is possible to substitute one size of a product for another size that will meet or exceed requirements. For example, a motor of a higher horsepower may be substituted for one of a lower horsepower and be acceptable to the customer. Substitution, however, comes at a cost. In the literature this problem has typically been referred to as the assortment problem or the substitutable demand problem. The focus has generally been on inventory related issues. Early work (Sadowski 1959 and Wolfson 1965) assumed linear costs throughout and developed implementable algorithms. Subsequent work (Pentico 1976) extended this work to general concave production functions and a variety of substitution cost functions involving proportional, additive and fixed costs. The methodology used in this work is dynamic programming and an optimal stocking policy and a planning horizon are established. Stochastic demands have been incorporated (McGillivay *et al.* 1978, Pentico 1974, Tryfos 1985). More recently, motivated by problems in semiconductor manufacturing, random yields have been incorporated (Bitran and Dasu 1992).

In this paper, we are motivated by a specific problem that arises in the manufacture of multiple-piece blind fasteners that are available in a variety of diameters and grip lengths and for which substitution is possible. Production is in batches to

Revision received January 1999.

† College of Business, Center for High Technology Management, California State University, San Marcos, CA 92096-0001, USA.

‡ Graduate School of Management, University of California at Irvine, Irvine, CA 92717-3125, USA.

§ Graduate School of Management, University of California at Davis, Davis, CA 95616, USA.

* To whom correspondence should be addressed: e-mail: dlwoodruff@ucdavis.edu

International Journal of Production Research ISSN 0020–7543 print/ISSN 1366–588X online
http://www.tandf.co.uk/JNLS/prs.htm
http://www.taylorandfrancis.com/JNLS/prs.htm

meet a given demand scenario. This product is targeted at the construction market in an Asian country which is segmented into commercial and home construction. The commercial production market is highly unpredictable and is generally served by distributors who place large one-time orders with the manufacturer. Inventory is held by these distributors. The home market, on the other hand, is more predictable and generates a steady stream of orders over a multi-period time horizon. Each size produced incurs a considerable setup cost which militates against producing all sizes and encourages the selection of an optimal subset of sizes. In § 2, we give an integer programming model that determines the optimal subset of sizes to produce from a range of sizes. This is a particular case of the concave models discussed by Pentico (1976). Costs are associated with setup for a size, production and substitution. General observations about the form of the optimal solution are given as well as sensitivity with respect to the setup cost.

The advantage of an integer programming approach over dynamic programming approaches for our situation is the ability to handle high dimension even in the stochastic case and the ease with which additional constraints such as limited resource availability can be added. Specifically we construct the deterministic model using the algebraic modelling language AMPL (Fourer *et al.* 1993) and solve it using the linear integer programming code CPLEX (1994). We then extend the model to include multiple periods and stochastic demand in § 3. This program is addressed using a heuristic that exploits the characteristics of optimal single-period solutions. Computational results suggest that the heuristic is effective.

2. Single-period model

Suppose a product is available in a finite number N of sizes where 1 is the smallest size and N is the largest size. Further, suppose size i is substitutable for size j if $i > j$, i.e. larger sizes may fulfill demand for a smaller size. Let d_i, $i = 1, \ldots, N$, denote the demand for size i.

We introduce a cost structure for the production of the product. Let p_i, $i = 1, \ldots, N$, be the unit production cost for size i. Generally $p_i > p_j$ for $i > j$. Let s be the set up cost for producing units of any size and r be the unit penalty cost of meeting demand for size j with a larger size i.

We need the following decision variables:

$$z_i = \begin{cases} 1, & \text{if we produce size } i \\ 0, & \text{otherwise} \end{cases}$$

$$y_i = \text{number of units of size } i \text{ produced},$$

$$x_{ij} = \text{number of units of size } i \text{ cut to meet demand for size } j,\ j < i.$$

Hence, in order to find the optimal subset of the N sizes to produce so as to satisfy demand, we solve the following integer linear program:

$$\min \left(\sum_{i=1}^{N} \left[s z_i + p_i y_i \right] + r \sum_{j<i} x_{ij} \right) \quad \text{(ILP)}$$

subject to

$$y_i = d_i - \sum_{k>i} x_{ki} + \sum_{l<i} x_{il}, \quad i = 1, \ldots, N, \tag{1}$$

$$y_i - Mz_i \leqslant 0, \quad i = 1, \ldots, N, \tag{2}$$

$$z_i \in \{0, 1\}, \quad i = 1, \ldots, N, \tag{3}$$

$$x_i, y_i \in \{\text{non-negative integers}\}, \quad i = 1, \ldots, N. \tag{4}$$

The objective function denotes the sum of the setup cost, production cost, and substitution cost. Equation (1) is a balance equation to meet demand for size i, $i = 1, \ldots, N$. Inequality (2) ensures that if size i is not produced, then the production quantity is zero; otherwise the production quantity is unconstrained.

Program ILP is then a linear integer program with N zero–one variables and $2N$ integer variables. It is observed however that program ILP has a network structure. If z_i is fixed at 0 or 1, program ILP reduces to a pure network optimization model. A node is associated with each size and equation (1) represents conservation at each node.

It follows then that the optimal solution for $\boldsymbol{x}$ and $\boldsymbol{y}$ is integer. Hence constraints (4) may be relaxed to the form

$$\boldsymbol{x} \geqslant 0, \quad \boldsymbol{y} \geqslant 0,$$

which simplifies solution considerably since there are now only N zero–one variables and no general integer variables. In addition there are several characteristics of the optimal solution that are interesting to observe.

Remark 1 (Pentico 1976)**:** If it is optimal to produce size i, then no larger size is used to meet demand for size i, i.e. if $y_i > 0$ then $x_{ki} = 0$ for all $k > i$. Conversely, if it is optimal to meet demand for size i by substituting a larger size, it is non-optimal to produce any of size i, i.e. if $x_{ki} > 0$ for any $k > i$, then $y_i = 0$.

Remark 2 (Pentico 1976)**:** Similarly it may be shown that it is nonoptimal to substitute a larger size for i and then use i to substitute for a smaller size, i.e. $x_{ki}x_{il} = 0$.

These remarks, and the integer property, valid in the absence of capacity constraints are also true for the following situation. Suppose that a resource is in limited supply R and that size i consumes an amount r_i of this resource. Clearly $r_i < r_j$ for $i < j$. Consequently, we have the additional constraint

$$\sum_i r_i y_i \leq R. \tag{5}$$

A simple proof may be developed by contradiction. Suppose it is optimal to produce both a particular size i and cut larger size j to service demand for size i. Then cost could be reduced by not cutting down the larger size j and entirely meeting the demand for size i by producing for size i. The resource constraint is then easier to satisfy and there is a contradiction.

Similarly suppose there is a non-integer solution. It follows that the demand for at least one size is met by two or more different sizes. Then it is cheaper and feasible to produce one item of the smallest size and no fractions of the larger sizes. Once again we have a contradiction.

Of course, in the case where $r_i = 1$ for all i, we have a network and the solution is automatically integral. A version of this constraint will be used in the multi-period formulation which follows.

Remark 3: In the motivating example, the company was concerned about the sensitivity of the optimal solution to the setup cost. In order to study this it is convenient to write program ILP in the following form:

$$OV(s) = \min_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}} \left\{ \sum_{i=1}^{N} sz_i + F(\boldsymbol{x},\boldsymbol{y}) | (\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}) \in \mathcal{C} \right\}, \tag{6}$$

where

$$F(\boldsymbol{x},\boldsymbol{y}) = \sum_{i=1}^{N} \quad p_i y_i + r \sum_{j<i} x_{ij} \Bigg)$$

and $\mathcal{C}$ summarizes the constraints. It is clear that increasing the setup cost s will increase the objective value $OV(s)$. Hence $OV(\cdot)$ is increasing in s. Further $OV(\cdot)$ is piecewise linear, continuous and concave (Noltemeier 1970). This parallels the case of a parametric linear programming and essentially follows since the feasible region of ILP can be replaced by its convex hull without loss of optimality. Consequently, there is a range of s for which the OV function has a constant slope. Finally, suppose (x^0, y^0, z^0) is optimal for $s = s_1$, and let $\sum_{i=1}^{N} z_i^0 = N(s_1)$ be the optimal number of sizes. From equation (6), we have

$$OV(s_1) = s_1 N(s_1) + F(x^0, y^0).$$

Since $N(s_1)$ is a constant for a range of sizes, it follows from the above that $OV(s)$ is a piecewise linear increasing concave function where the slope of the linear segments is the number of sizes to produce at optimality for a given setup cost. This reflects our intuition that as the setup cost increases we make fewer different sizes in order to incur fewer setup costs.

3. Multi-period, stochastic formulation

Formulation ILP is a useful planning tool, but it ignores the fact that there are likely to be additional demands in the next period (e.g., next month). We may not know exactly what those demands will be, but we would be better off if our model could incorporate whatever we know about them. If demand constraints for the current period are based on firm orders but future demands are based on forecasts or conjecture, then they should not be treated in the same fashion. We must recognize that future demand constraints are stochastic. That is to say that they should be modelled as random variables.

It may not be reasonable or useful to consider the entire demand probability distribution functions. It may not be reasonable because there may not be sufficient data to estimate an entire distribution. It may not be useful because the essence of the stochastics may be captured by specifying a small number of representative *scenarios*. We assume that scenarios are specified by giving a full set of random variable realizations and a corresponding probability. We index the scenario set, $\mathcal{L}$, by ℓ and refer to the probability of occurrence of ℓ (or, more accurately, a realization 'near' scenario ℓ) as $Pr(\ell)$. We refer to solution systems that satisfy constraints with probability 1 as *admissible*.

In addition to modelling stochastics, we would like to model *recourse* as well. That is, we would like to model the ability of decision makers to make use of new information (e.g., orders) at the start of each planning period. We allow our solution vectors to depend on the scenario that is realized, but we do not want to assume prescience. We refer to a system of solution vectors as *implementable* if for all decision times t, the solution vector elements corresponding to period $1, \ldots, t$ are constant with respect to information that becomes available only after stage t. We refer to the set of implementable solutions as $\mathcal{N}_{\mathcal{L}}$. It is possible to require implementable solutions by adding *non-anticipatitivity constraints*, but we will instead make use of solution procedures that implicitly guarantee implementable solutions.

In the stochastic, multi-period formulation that follows the objective is to minimize expected costs. We invoke the network equivalence given earlier to drop the explicit requirement that $\boldsymbol{x}$ and $\boldsymbol{y}$ be integers. Variables and data are subscripted with a period index t that takes on values up to T. To model the idea that sleeves produced in one period can be used as-is or cut in subsequent periods, we use x_{ijt} to indicate that sleeves of length index i are to be used without cutting in period t if $i = j$ and with cutting otherwise. The $\boldsymbol{y}$ vector gives production quantities for each length in each period without regard to the period in which they will be used (and perhaps cut). The formulation is essentially an extension of ILP except that a capacity constraint must be added in the multiple-period formulation. Holding costs could be added, but an additional subscript becomes necessary without the benefit of any additional insight. As an aside, note that the addition of holding costs would add a large number of continuous variables, but no new integers so the impact on computational performance would not be catastrophic.

$$\min \sum_{\ell \in \mathcal{L}} \Pr(\ell) \sum_{t}^{T} \left[\sum_{i=1}^{N} \left(sz_{it\ell} + p_i y_{it\ell} \right) + r \sum_{j<i} x_{ijt\ell} \right] \qquad \text{(SMIP)}$$

subject to

$$\sum_{j \geqslant i} x_{ijt\ell} \geqslant d_{it\ell}, \quad \ell \in \mathcal{L},\ i = 1, \ldots, N,\ t = 1, \ldots, T \tag{7}$$

$$\sum_{t' \leqslant t} \left[\sum_{j \leqslant i} x_{ijt'\ell} - y_{it'\ell} \right] \leqslant 0, \quad \ell \in \mathcal{L},\ i = 1, \ldots, N,\ t = 1, \ldots, T \tag{8}$$

$$y_{it\ell} - M z_{it\ell} \leqslant 0, \quad \ell \in \mathcal{L},\ i = 1, \ldots, N,\ t = 1, \ldots, T \tag{9}$$

$$\sum_{i=1}^{N} y_{it\ell} \leqslant c_{t\ell}, \quad \ell \in \mathcal{L},\ t = 1, \ldots, T \tag{10}$$

$$z_{it\ell} \in \{0, 1\}, \quad \ell \in \mathcal{L},\ i = 1, \ldots, N,\ t = 1, \ldots, T \tag{11}$$

$$\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \in \mathcal{N}_{\mathcal{L}}. \tag{12}$$

Bear in mind that solution vector elements corresponding to periods two through T are not actually intended for use, they are computed just to see the effect that period 1 (the current period) decision would have on future optimal behaviour. At the start of period 2—or at any other time—the decision maker would run the model again with updated demands for the current period and new scenario estimates.

3.1. *Insights from small instances*

A two-period model is large enough to offer insights into the effects of random demand and the availability of recourse, but is still small enough to allow exact solution if there is a small number of scenarios. A simple way to solve a stochastic program is to produce what is known as the *deterministic equivalent* or DE.

For a two-period model this means eliminating the dependence on the scenario for first-period variables. For example, rather than providing non-anticipatitivity constraints such as $y_{i1\ell} - y_{i1\ell'} = 0$ for all i, and $\ell \neq \ell'$ we simply convert all occurrences of $y_{i1\ell}$ to y_{i1}. We can also factor the first-period variables out of the probability summation in the objective function if we wish to. Once we have created the deterministic equivalent we can solve it using a standard mixed integer programming solver such as CPLEX if the instance is not too big. We will refer to CPLEX applied to the DE as algorithm CDE. Notice that the method is unchanged if demands are not the only data that are random. Also notice that the number of integer variables grows with the number of scenarios.

Consider an instance in which the period one data is given in table 1 and the capacity for each period is 200 000 sleeves. Suppose that there are two scenarios with equal probability for second-period demand: 70%, and 130% of the demand shown in table 1. There are a number of possibilities for the use of optimization models:

(1) *Single-period model.* If we implement the ILP solution for period one and then re-optimize for period-two operation, the expected cost of operation for the two-period horizon is \$225 577.

(2) *Multi-stage deterministic model.* If we solve SMIP using a single scenario which uses the expected value of the data to get the first-period solution, then re-optimize at the start of the second period, the expected cost for the two-period horizon is \$25 472. A modest monthly savings over the single-period model for this particular instance. Compared with the one-stage solution, the first-period elements of this solution include production of two additional sleeve sizes and larger production quantities.

(3) *Multi-stage stochastic model.* If we solve the problem using formulation SMIP we get an expected cost of \$224 338 which represents a much larger

Sleeve length	Unit production cost (\$)	Demand
25	0.748	2 500
30	0.7584	7 500
35	0.7688	12 500
40	0.7792	10 000
45	0.7896	35 000
50	0.8	25 000
55	0.8104	15 000
60	0.8208	12 500
65	0.8312	12 500
70	0.8416	5 000
Unit cutting cost: \$0.008 Setup cost: \$453		

Table 1. The one-period data set.

savings. Compared with the one-stage solution, the first-period elements of this solution also call for larger production quantities but only one additional sleeve size.

Of course the exact nature of the changes in first-period solution values and the resulting savings depend on the data. This demonstration supports our intuition that since a multi-stage stochastic model is a better representation of the situation faced by managers, it would be preferable to use such a model if it is computationally practical. The next section presents a tractable algorithm.

3.2. *Larger instances*

There are no general purpose, integer multi-stage stochastic programming solvers commercially available or even reported in the literature. If one drops the requirement that integers be used, there are a number of approaches that are possible (see, e.g., Birge and Wets 1991, Rockafeller and Wets 1992, Infanger 1994). Approaches for some cases with integer variables in the first stage have been discussed (see, e.g., Averbakh 1990). However, for our application, there are a large number of integer variables and they are a central feature of the formulation in every period.

The two-period, two-scenario instance of SMIP used as an example was solved using its deterministic equivalent. The solution time for CDE on a DEC alpha 3000/700 (the `specfp92` value is about 225) was 72 sec. If a third scenario is added, CPLEX had not converged after over 1800 sec and generated over 200 000 nodes in its branch and bound tree because each scenario adds $N = 10$ integer variables. Given this information and the fact that integer programs are NP-hard, we conclude that for general instances with multiple stages and scenarios, a heuristic approach is required.

We begin by noting that remarks 1 and 2 imply that the solution to a single-period ILP instance is completely specified by the vector $\boldsymbol{z}$. This, in turn, suggests that a greedy algorithm for setting $\boldsymbol{z}$ would be useful. Such an algorithm is given in figure 1.

The key step in the algorithm is 4(*c*). If the cost of cutting and differential production costs are not greater than the cost of a setup, then demand for sleeve length j is met by production of length i. This simple greedy algorithm carries no performance guarantees but it does work pretty well on the table 1 data set. The

(1) $m \quad \text{argmax}_i \{d_i > 0\}$
(2) $z_i \quad 0$ for $m < i \leqslant N$
(3) $i \quad m$
(4) WHILE $i \geqslant 1$
  (*a*) $z_i \quad 1$
  (*b*) $j \quad i - 1$
  (*c*) WHILE $j \geqslant 1$ AND $(r + p_i - p_j)d_j < s$:
    (i) $z_j = 0$
    (ii) $j \quad j - 1$
  (*d*) $i \quad j$

Figure 1. Algorithm G1: greedy algorithm for setting $\boldsymbol{z}$ in ILP.

solution it gets is z= (0,0,0,0,1,1,1,0,0,1) which implies a cost of \$112 974 for the month versus the optimal of \$112 775 and current practices of \$114 140.

By randomizing the algorithm, we can guarantee that it will find the optimal solution eventually. The generalized random adaptive search procedure (GRASP) concept (Feo and Resende 1988) serves well for this purpose. Using the notation in algorithm G1, it is clear that z_m must be one, so the appropriate place for randomization is step 4(c). To produce algorithm R1, we modify this step to be

$$\text{WHILE}[\, j \geqslant 1] \text{ AND}[\{(r + p_i - p_j)d_j < s \text{ AND } R > \alpha\} \text{ OR } R > (1 - \alpha)]$$

where $0 < \alpha < 1$ is a parameter of the algorithm and R is a random variable between zero and one. Note: R is not a random *number* but a random variable; hence, each invocation of step 3 results in independent random instantiations. The new AND clause will occasionally result in early loop termination and the new OR clause will occasionally cause it to continue longer than it otherwise would have. To produce algorithm GRASP1 with parameters α and γ we execute algorithm R1 γ times, keeping track of the solution that implies the lowest objective function value seen so as to report this solution vector on termination. The following result has a very short proof.

Remark 4: For an instance of ILP, as γ goes to infinity, with probability 1 algorithm GRASP1 will report a z vector that implies an optimal solution.

In preparation for a two-stage algorithm, we create algorithm R2 which is the same as algorithm R1 except that we modify steps 2 and 4(a) to do the assignment if and only if $(R > \beta)$ where R is a random number between zero and one and $0 < \beta < 1$ is a parameter of the algorithm. In other words, some of these variables may be left unassigned. This algorithm, obviously, no longer can be counted on to produce a valid solution for ILP. It is intended to be imbedded in algorithm GRASP2, which is given in figure 2.

Remark 5: For a two-period instance of SMIP, as γ goes to infinity, with probability 1 algorithm GRASP2 will report an optimal solution.

This remark reassures us that increasing effort will produce improving results, but does not tell us enough about the practical performance of the algorithm. We are interested in its response to changes in its parameters. Both α and β tend to move the algorithm away from simple greedy performance. Intuitively, as α is increased the

(1) Create the deterministic equivalent (DE) for the instance of SMIP
(2) Repeat γ times:
    (a) Execute algorithm R2 for the first period and assign the result to the corresponding variables in (DE) by fixing and freeing them as dictated by the solution to R2.
    (b) For each scenario, execute algorithm R2 for the second period as a single period and assign the result to the corresponding variables in (DE) by fixing and freeing them as dictated by the solution to R2.
    (c) Solve the resulting instance of (DE).
    (d) If the objective function value is the best seen, record the solution.
(3) Report the best solution seen.

Figure 2. Algorithm GRASP2(α, β, γ): two-period randomized algorithm for SMIP.

algorithm becomes more random and as β is increased each iteration will take longer but produce better results. The parameter γ also controls the amount of time and quality of solutions. In addition to these parameters, we can also modify the parameters that control CPLEX as it solves the modified DE problems that are created on every iteration. The only CPLEX parameter that we vary as we study computational performance in the next section is the branch and bound tree node limit.

3.3. *Computational experiments*

We conducted in excess of 5000 computational experiments, some of which are described in the Appendix. We had two goals: study of changes with changing setup costs and study of the performance of the algorithm. The results can be summarized as follows:

- Changes in the solution as s is changed are summarized in table 2. These tests were done on the ten scenario instance named SZ10 described in the Appendix. The GRASP2 parameters were $\alpha = \beta = 0.2$ and $\gamma = 100$. As with the single-period deterministic model, the number of setups and the expected savings over the status quo (no cutting) are increasing in the setup cost. The model could be formulated with setup cost as a random variable as well as demand. No changes to the algorithm are required. However, after production begins the setup cost will be 'known' at least as well as any cost is, so that ongoing use of a model with stochastic setup costs does not seem useful.
- GRASP2 clearly dominates CDE and the degree of domination increases with the problem size. It can find better solution much faster and over a broad range of parameters. This is not terribly surprising since GRASP2 uses CPLEX to solve sub-problems. On the other hand, it is important because the only method heretofore available for multi-stage, stochastic integer programs was the use of a solver such as CPLEX on the DE (i.e. CDE).
- Remark 5 appears to be relevant for short periods of time. That is, additional time is well spent. The algorithm finds better solutions with larger values of γ. A few minutes on a workstation is sufficient to get good solutions to 20-scenario, two-period problems.
- Solution quality is also improved if CPLEX is allowed to spend more time on sub-problems. In other words, if the CPLEX node limit is increased. This is

Setup cost ($)	Optimal number of sizes	Expected two-period savings
200	7	719.67
300	5	1748.10
400	5	2907.17
453	5	3469.26
500	5	4042.81
600	5	5145.28
700	3	6601.90

Table 2. Sensitivity of the solution to the setup cost.

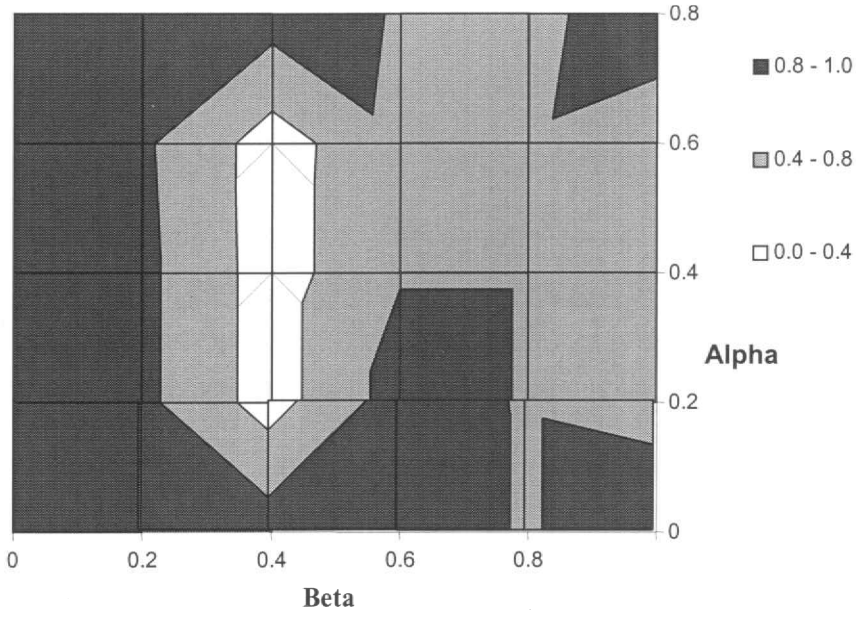

Figure 3. Contours of undominance of CDE as described in the Appendix.

relevant only for larger instances and only when β is significantly greater than zero.

- GRASP2 is fairly robust with respect to parameters. This is demonstrated graphically in figure 3. This figure is a countour plot of fractions of runs where GRASP2 is undominated by CDE. The details are described in the Appendix, but in this simple model we are using CDE just to provide calibration for our parameter sensitivity experiments. The parameter γ is varied over the range from one to one hundred over thousands of runs on five problem instances. The thing to notice is that the performance is robust with respect to α and β. There is a narrow band of sensitivity near $\beta = 0.4$ and otherwise the performance based on the simple, but important, measure of dominance is very robust.

4. Conclusions

Substitution of larger sizes for smaller affords manufacturers a means of reducing setup costs and hedging production against unknown future demand. We have presented formulations, insights and algorithms to help exploit this opportunity. Computational experiments on real-world data suggest that our contribution can be very meaningful.

We address the stochastic, multi-period formulation using an algorithm that takes advantage of structural results from the deterministic single-period model. If additional detail is needed, the formulation and algorithms can easily be extended to include holding costs. Extension to more than two periods, any or all data stochastic, and the use of discounting are all trivial. It remains as future research to determine which if any of these extensions are of practical importance.

An additional topic of future research concerns post optimality analysis. A manufacturing manager might reasonably wonder which scenarios have the biggest impact on the solution for the current period with an eye toward refining probability estimates and scenario descriptions for the more important scenarios. A related area concerns graphical interface for input/display of scenarios, solutions and importance measures. These issues will become increasingly important as the models used in production planning gain realism by incorporating integer variables, multi-period stochastics and recourse.

Acknowledgements

This work was supported in part by the National Science Foundation.

Appendix

In order to test parameter sensitivities for algorithm GRASP2 and its performance relative to CDE (CPLEX applied to the DE), we create a series of simple related instances SMIP. Each has costs and first-period demands given by table 13 and stochastic second period demands with a mean given by table 13 and a symmetric set of scenarios. For each instance, each scenario has equal probability. The instances are specified by a vector of multipliers—one multiplier for each scenario that is multiplied times the demand vector from table 13 to give the demand vector for the scenario. Each instance is given a short ID that indicates the number of scenarios. Instance SZ3 is defined by $(0.7, 1, 1.3)$; SZ5 by $(0.6, 0.8, 1, 1.2, 1.4)$; SZ10 by $(0.5, 0.6, 0.7, 0.9, 1.1, 1.2, 1.3, 1.4, 1.5)$; and SZ20 by $(0.5, 0.55, \ldots, 1.5)$. Note that SZ20 may be considered a higher-resolution sample from the same distribution as SZ10. Even though the algorithms do not depend on the distributions of the scenarios, we also generated instances that are 'more random' as follows: For each instance there is a random number B that is uniform on $(0.5, 1.5)$ and constant for the instance. Each demand is then obtained by multiplying the demand given in table 13 by B and by a uniform $(0.5, 1.5)$ random number. These instances are given IDs SA10, SB10, SC10, SD10 and SE10.

Table A 1 shows results for CDE for each instance. Different CPU times are obtained by varying the node limit parameter of CPLEX between 20 000 and 250 000.

Space limitations preclude display of the large number of runs done to test parameter sensitivity for GRASP2. Table A 2 shows a portion of the low γ runs done for this purpose. The column labelled 'seconds' gives the total CPLEX solve time for the sub-problems generated by GRASP2 for each instance (remember that GRASP2 uses CPLEX to solve induced problems). Time spent looping and tracking the

ID	Seconds	Obj. Val
SZ3	1859.8	224 747
SZ5	4195.2	225 624.7
SZ10	4.9	228 066.4
SZ10	56.0	226 936.9
SZ10	7715.5	226 936.9
SZ20	204.5	230 667.6

Table A 1. CPLEX applied to the DE (algorithm CDE).

			$\gamma = 2$		$\gamma = 10$	
ID	α	β	Seconds	Obj. Val.	Seconds	Obj. Val
SZ3	0.2	0.2	0.3	225 095.1	1.2	225 095.1
	0.2	0.4	1.4	224 846.8	25.1	224 846.8
	0.2	0.6	6.0	224 810.6	115.5	224 734.1
	0.4	0.2	0.6	226 031.7	1.6	225 190.4
	0.4	0.4	7.5	225 481.7	19.1	224 877.0
	0.4	0.8	1 745.8	224 561.8	5 498.5	224 561.8
	0.6	0.2	0.4	225 803.9	1.2	225 726.1
	0.6	0.4	3.6	225 343.5	15.4	225 179.3
	0.6	0.6	19.6	225 160.4	243.2	224 652.7
SZ5	0.0	0.0	0.1	226 084.7	0.5	226 084.7
	0.0	0.6	425.6	224 732.1	10 578.8	224 732.1
	0.2	0.0	0.1	226 116.3	0.5	225 573.7
	0.2	0.2	0.9	225 362.5	616.3	225 099.8
	0.4	0.0	0.1	227 735.4	0.5	225 773.7
	0.4	0.2	1.7	226 108.5	4.9	225 316.2
	0.4	0.4	570.4	225 688.6	899.1	225 022.1
	0.6	0.0	0.1	228 248.4	0.5	227 094.8
	0.6	0.2	1.6	225 973.1	4.4	225 756.9
	0.6	0.4	1 479.2	225 596.8	1 960.9	224 984.4
SZ10	0.0	0.0	0.2	225 947.0	1.0	225 947.0
	0.0	0.6	178.2	225 505.4	883.3	225 388.1
	0.2	0.0	0.2	226 037.0	1.1	225 547.7
	0.2	0.2	63.7	225 666.4	175.1	225 449.2
	0.2	0.4	143.8	225 420.9	753.0	225 157.0
	0.2	0.6	188.6	225 824.1	976.9	225 399.7
	0.2	0.8	219.7	225 921.8	1 261.0	225 573.0
	0.4	0.2	7.0	226 161.3	184.7	225 415.5
	0.4	0.4	171.4	226 007.9	791.0	225 777.4
	0.4	0.6	184.8	226 271.2	991.1	225 639.2
	0.6	0.2	5.4	225 885.2	58.5	225 722.2
	0.6	0.4	141.5	226 103.9	723.6	225 513.7
SZ20	0.0	0.0	0.6	226 005.5	2.7	226 005.5
	0.0	0.2	225.3	225 786.4	1 159.5	225 256.2
	0.0	0.4	337.5	226 167.8	1 454.3	225 463.2
	0.0	0.6	330.9	225 836.1	1 637.5	225 806.7
	0.0	0.8	364.8	226 546.3	1 850.5	226 546.3
	0.2	0.0	0.5	226 070.2	2.8	225 685.7
	0.2	0.2	227.2	226 022.4	1 168.8	225 612.4
	0.2	0.4	253.1	226 152.3	1 454.0	225 397.0
	0.2	0.6	267.7	226 246.7	1 538.4	226 148.1
	0.4	0.0	0.5	227 587.9	2.8	225 886.2
	0.4	0.2	236.5	226 219.0	1 186.2	225 668.0
	0.4	0.4	233.7	226 233.1	1 424.6	225 873.8
	0.4	0.6	315.5	226 916.2	1 502.8	226 062.1
	0.6	0.0	0.5	228 555.0	2.5	227 186.2
	0.6	0.2	195.4	226 322.5	971.2	226 034.0
	0.6	0.4	287.7	226 377.6	1 395.9	226 169.0
	0.6	0.6	317.4	227 279.9	1 562.4	226 539.3

Table A 2. Experimental results.

			$\gamma = 2$		$\gamma = 10$	
ID	α	β	Seconds	Obj. Val.	Seconds	Obj. Val
SZ10	0.0	0.0	0.2	225 947.0	1.1	225 947.0
	0.0	0.2	25.2	225 487.7	164.2	225 124.7
	0.0	0.4	4245.1	225 528.1	23 908.4	225 155.8
	0.2	0.0	0.2	226 037.0	1.1	225 547.7
	0.2	0.2	73.6	225 666.4	188.5	225 449.2
	0.2	0.4	4818.7	225 331.1	37 316.2	0.2
	0.4	0.0	0.2	227 731.1	1.1	225 621.8
	0.4	0.2	7.2	226 161.3	2282.9	225 387.6
	0.4	0.4	5124.5	225 842.3	20 791.7	225 567.7
	0.6	0.0	0.2	227 926.8	1.0	227 034.3
	0.6	0.2	5.6	225 885.2	60.9	225 722.2
	0.8	0.0	0.2	227 491.7	0.9	227 477.5
	0.8	0.2	207.3	226 616.7	233.5	225 854.9

Table A.3. Experimental results (CPLEX node limit = 250 000).

best solution is negligible by comparison in an efficient implementation. The column labelled 'Obj. Val.' is the SMIP objective function value. The CPLEX integer node limit set at 20 000. Table A 3 shows the second set of results for SZ10 where the node limit is 250 000.

The data for figure 3 was generated to facilitate an understanding of the sensitivity of the GRASP2 algorithm to its parameters α and β. For this figure, GRASP2 is said to be undominated by CDE if CDE cannot find a solution in less time that is better than the solution found by GRASP2. Since the interest here is in GRASP2 parameter sensitivities, far more GRASP2 runs were used than CDE runs. Although CDE is not very sensitive to the number of nodes it is allowed to expand (which is the major determinant of its run time) because of the combinatorial nature of its branching tree, the reader is nonetheless cautioned against using this figure to predict dominance of GRASP2.

References

AVERBAKH, I. L., 1990, An additive method for optimization of two-stage stochastic systems with discrete variables. *Soviet Journal of Computer and System Sciences*, **28**, 161–165.

BIRGE, J., and WETS, R. J.-B. (editors), 1991, Stochastic programming *Annals of OR*, Vols. 30 and 31.

BITRAN, G. R., and DASU, S., 1992, Ordering policies in an environment of stochastic yields and substitutable demands. *Operations Research*, **40**, 990–1017.

CPLEX, 1994, *Using the CPLEX Callable Library*, ILOG CPLEX Division, Suite 200, 889 Alder Avenue, Incline Village, NV 89451, USA.

FEO, T. A., and RESENDE, M., 1988, A probabilistic heuristic for a computationally difficult set covering problem. *Operations Research Letters*, **8**, 67–71.

FOURER, R., GAY, D. E., and KERNIGHAN, B. W., 1993, *AMPL* (San Francisco, CA: Scientific Press).

INFANGER, G., 1994, *Planning under uncertainty: solving large scale stochastic linear programs* (Danvers, MA: Scientific Press).

MCGILLIVAY, A. R., and SILVER, E. A., 1978, Some concepts for inventory control under substitutable demand *Infor*, **16**, 47–63.

NOLTEMEIER, H., 1970, Sensitivitatas analyse bei dikreten linearen optimierungs problemen. In: *Lecture Notes in Operations Research and Mathematical Systems*, Volume 30, edited by M. Bechmann and H. P. Kunzi (Berlin: Springer)

PENTICO, D. W., 1974, The Assortment problem with probabilistic demands. *Management Science*, **21**, 286–290; 1976, The Assortment problem with nonlinear cost functions. *Operations Research*, **24**, 1129–1142.

ROCKAFELLER, R. T., and WETS, R. J-B., 1992, Scenario and policy aggregation in optimization under uncertainty. *Mathematics of OR*, **16**, 119–147.

SADOWSKI, W., 1959, A few remarks on the assortment problems. *Management Science*, **6**, 13–24.

TRYFOS, P., 1985, On the optimal choice of sizes. *Operations Research*, **33**, 678–684.

WOLFSON, M. L., 1965, Selecting the best length to stock. *Operations Research*, **13**, 570–585.

ZANGWILL, W. I., 1968, Minimum concave cost flows in certain networks. *Management Science*, **14**, 429–450.