1995

The development and testing of new, single and multiple echelon, dynamic, capacitated, lot sizing heuristics

Robert Jay McCoy
Iowa State University

The development and testing of new, single and multiple echelon, dynamic, capacitated, lot sizing heuristics

by

Robert Jay McCoy

A Dissertation Submitted to the
Graduate Faculty in Partial Fulfillment of the
Requirements for the Degree of
DOCTOR OF PHILOSOPHY

Department: Industrial and Manufacturing Systems Engineering
Major: Industrial Engineering

Approved:

Signature was redacted for privacy.

Signature was redacted for privacy.

In Charge of Major Work

Signature was redacted for privacy.

For the Major Department

Signature was redacted for privacy.

For the Graduate College

Iowa State University
Ames, Iowa

1995

UMI Number: 9531769

TABLE OF CONTENTS

ACKNOWLEDGMENTS

The following are acknowledged:

- The US Department of Education for the fellowship and funding support.
- Dr. Montag and the ISU IMSE Department for their assistance and funding support.
- Dr. Gemmill for his support and assistance as my major advisor.
- Doctors Barta, David, Flynn, and Wacker for serving on my ISU research committee.
- Doctors Trigeiro (MITRE), Diaby (U. of Conn.), Thizy (SUNY) and Van Wassenhove (INSEAD) for their test problems, heuristics and problem generators.
- My wife, Marisa, and parents for their love and emotional support.

GENERAL INTRODUCTION

Organization of the Dissertation

This dissertation introduces and tests three new heuristics and two random problem generators. These heuristics and problem generators are associated with single and multiple echelon, dynamic, capacitated lot sizing problems, with or without setup times.
The development, application and testing of these heuristics and generators are described in the four papers contained in Chapters I, II, III and IV.

The first paper discusses extensions to the basic Dixon and Silver (DS) heuristic that allow for improved solutions at a minor computational increase. These extensions include the use of newly developed perturbation factors and improvement algorithms. This paper was accepted for publication in the *3rd Annual Industrial Engineering Research Conference Proceedings* and was presented at the Atlanta (GA) conference in May, 1994.

The second paper is associated with the development and testing of a new, single echelon heuristic, called the MG heuristic. This heuristic, unlike the DS heuristic, is capable of solving problems that include significant levels of setup time. Furthermore, unlike heuristics that use Lagrangean relaxation with subgradient optimization, it is a fast heuristic. Additionally, the second paper describes an extensively modified, single echelon, random problem generator. This paper is currently under review by the *Computers and Industrial Engineering* journal.

The third paper provides additional test results associated with the MG heuristic and further specifics relating to the single echelon random problem generator. This paper will be presented at the 4th Annual Industrial Engineering Research Conference in Nashville (TN), May, 1995, and is pending publication in the conference proceedings.

The final paper deals with a new heuristic solution methodology that is directed towards solving multiple echelon problems with assembly product structures, with allowances for setup times and capacity constraints on one or more echelons. This paper is in review by the scheduling and logistics focus area of the *IIE Transactions* journal.

The heuristic discussed in the fourth and final paper is a practical heuristic that integrates many of the findings and methodologies discussed in the first three papers. Thus, it uses a sequential, top down approach that uses the single echelon MG heuristic and could be modified for implementation into existing MRP production planning systems. This new, multiple echelon heuristic, called the MELS heuristic, incorporates: (a) two different item cost adjustment methodologies; (b) an upward adjustment (feedback) methodology that is used by the heuristic when higher echelon lot sizing decisions "overload" the capacity constraint of a lower echelon; and (c) a search procedure (i.e., simulated annealing) for optimizing the holding cost adjustment factor used for each echelon. Additionally, the fourth paper provides: (1) a mathematical definition of the multiple echelon problem; (2) an introduction and description of a new, multiple echelon, random problem generator; (3) insight into the cause and magnitude of cost penalties that are incurred when inter-echelon cost impacts are ignored; (4) indications regarding the effectiveness of item cost adjustments; and (5) a brief introduction to simulated annealing, as well as a description of its application to multiple echelon lot sizing.

In this dissertation, the Background, Literature Review and Motivations subsections are first presented prior to the incorporation of the four papers discussed above. In these subsections, the references cited refer to those that are listed at the end of the dissertation in the Bibliography section. Additional references are listed at the end of each paper (chapter).

Background

The Capacitated Lot Sizing Problem (CLSP) is a common and important industrial inventory control problem that involves determining medium-range production planning lot sizes for items that share a common capacity constraint. It is a single echelon problem that consists of scheduling for multiple periods the production timing and quantity of multiple items so as to minimize the sum of both setup and linear holding costs while satisfying demands, without backlogs, and adhering to the capacity constraint. The capacity constraint and product demands may be time-varying (dynamic), but are assumed to be deterministic, even though they are often based on forecasts. Furthermore, item setup costs are incurred in each period that a particular item is produced. Additionally, the problem is significantly complicated by the inclusion of positive item setup times (which consume capacity). However, CLSP considers only lot sizing decisions, not short term job sequencing.

Another major complication is associated with multiple echelon, capacitated lot sizing problems, i.e., when the lot size decision made at one echelon of the manufacturing process (e.g., assembly) critically affects inter-related lot sizing decisions required at a lower echelon (e.g., fabrication). These types of problems have been referred to as 'cascading lot sizing problems'. A multiple item, Material Requirements Planning (MRP) environment associated with a job shop manufacturing firm having multiple, interdependent work centers using a medium-range planning horizon and relatively long planning periods (weekly or monthly) is one example where these types of lot sizing problems must routinely be solved.

These types of production planning problems are extremely complex. The single item, single stage, non-capacitated version of the problem without setup times can be solved efficiently using Wagner and Whitin's methodology (1958), but just adding a capacity constraint converts the problem to one that is NP-hard, Dixon (1979). Consequently, both single and multiple echelon CLSP research attention has primarily focused on the development of heuristics rather than optimization methodologies. As a result, over the last twenty years, a number of heuristics have been developed. However, until just recently, much of the complexity of the general lot sizing problem has been neglected (e.g. positive setup times, multiple echelons, etc.). And currently, there does not exist a heuristic methodology which has been designed to handle the general lot sizing environment described above.

Determining if problems are feasible is a major complication of dealing with positive setup times -- the issue of determining feasibility in this environment is an NP-Complete problem, Trigeiro et al. (1989). Consequently, few researchers have studied these types of problems--even just measuring how well a particular heuristic finds feasible solutions is a difficult task, without considering the quality of its solutions (w.r.t. cost).

At the start of dissertation research, the Lagrangean relaxation heuristic of Diaby et al. (1992a; 1992b) appeared to offer significant potential for solving single echelon problems, with or without setup times and overtime. However, after spending a significant amount of time working with the FORTRAN code supplied by the primary author, it was concluded that their approach is not robust over a wide variety of problem characteristics -- their methodology often would not generate a solution to problems that were shown to be feasible by other, more robust lot sizing procedures, e.g., the TTM heuristic of Trigeiro et al., (1989). This was discussed with the primary author and several randomly generated test problems were provided to him as examples.

Literature Review

In just the last twenty years, many dozens of articles have been written relating to single and multiple echelon, dynamic, capacitated lot sizing. An overview of some of the most relevant, recent articles is provided below.

In 1975, Eisenhut extended the well-known Silver-Meal heuristic to solve single echelon, capacitated, multi-item problems. Lambrecht and VanderVeken (1979) extended the work of Eisenhut and developed a heuristic that guarantees feasibility (if feasibility is possible) of the resulting lot-sizing solution. In 1981, Dixon and Silver published a relatively simple (and fast), 'greedy' heuristic that built upon previous enhancements to the Silver-Meal heuristic and uses a forward look-ahead to ensure feasibility. Since then, this heuristic has received considerable, favorable attention as both a stand-alone solution methodology and as a starting point for more complicated, computationally expensive methodologies. For example, both the Thizy and Van Wassenhove (1985) and Diaby et al. (1992a and 1992b) Lagrangean relaxation approaches use Dixon and Silver's solution to initialize dual costs associated with a transportation network formulation of the lot sizing problem. Additionally, other available, less well established heuristics that are somewhat competitive with Dixon and Silver's heuristic include the DPA (Dogramaci et al. 1981) and the ABCX (Maes and Van Wassenhove 1986) heuristics. And finally, Trigeiro's Dual Cost Heuristic (1989) is noteworthy because it, along with Diaby's Heuristic, is one of the few heuristics that permit the inclusion of significant product set up times. Of these methodologies, all are related to single echelon models.

For general reviews of the research related to multiple echelon production-inventory systems, see Collier (1982) and Goyal/Gunasekaran (1990). (Note: In place of the word 'echelon' some papers and dissertations use the words 'stage' or 'level'.) The following paragraphs will briefly discuss a few of the key heuristic methodologies.

Regarding heuristic methodologies for multiple echelon, non-capacitated lot sizing of products with time varying demand, a number of researchers have proposed sequential lot sizing heuristics that compute lot sizes one echelon at a time. Among these researchers are McLaren (1976), McLaren and Whybark (1976), Biggs et al (1977), Graves (1981), Peng (1985), and Blackburn and Millen (1982, 1984, & 1985). Rather than ignore upper and/or lower echelon impacts (as is commonly done in industrial practice), these heuristics use adjusted setup and holding costs that partially take into account the effect that a lot sizing decision made at one stage has on the other stages in the system. The major weaknesses of all these heuristics are their inability to include multiple items (products) with shared capacity constraints, their limitations on acceptable product structures (e.g., part commonality causes problems), and their omission of setup times. However, if shared capacity constraints are non-binding and product structures are acceptable, the use of any of the above multiple echelon heuristics, rather than simply using a single echelon heuristic iteratively down the product structure, will usually lead to significantly improved cost performance (e.g., a 5% to 15% reduction). Furthermore, Gupta and Keung (1990) concluded that Blackburn and Millen's approach is the most computationally effective, sequential lot sizing technique. Finally, Afentakis (1987) and Roundy (1993) have been active multiple echelon, CLSP researchers, but their approaches do not consider capacity constraints.

Research on the capacity constrained version of the multiple echelon, multiple item, dynamic demand CLSP problem has been relatively limited in comparison with the amount of research directed towards non-capacitated problems. Much of the research to date has focused on mathematical programming formulations of the problem. For example, Steinberg and Napier (1980) suggest a generalized network formulation, but their approach is only appropriate for very small product structures due to the very high computational requirements of mixed integer programming. This limitation also applies to the work of Billington (1983), which has the additional limitation of not allowing for setup costs. Additionally, Bahl and Ritzman (1984) also used a mathematical programming approach , but it simplifies the problem by using only fixed ordering interval schedules and its solution procedures will generate production schedules with fractional production setups.

Lambrecht and VanderEecken (1978) developed an algorithm, but it is applicable only to serial, single end item systems where the capacity constraint is binding on just the end item. Ramsay and Rardin (1983) also developed four different heuristics for serial systems. Their approaches allow multiple capacity constraints, but they utilize some very restrictive assumptions that are not consistent with realistic production environments. Blackburn and Millen (1984) developed a heuristic that is applicable to both serial and assembly systems with a single end item and allows capacity constraints on all stages, but the constraints must be constant (not time varying). Maes and Van Wassenhove (1991) use the cost modifications of Blackburn and Millen to solve multiple item, multi-capacitated, lot sizing problems in a serial production environment under dynamic demand conditions. Their results were mixed for the small test problems utilized. Consequently, they acknowledged a need to investigate other capacitated cost modification procedures and extend the testing to include larger problems which more closely approximate industry environments. Also, all the above approaches are limited by their inability to handle setup times.

Kuik et al.(1993) proposed linear programming (LP), simulated annealing (SA) and Tabu Search (TS) heuristics for solving lot sizing problems relating to assembly systems. In their study, they reduced the overall problem complexity by limiting the production structures to six or seven items spread over three echelons, only the middle echelon was capacitated, all three levels used the same time between order (TBO) levels (either 2, 3, or 4) to generate item setup costs, and no setup times were allowed. They concluded that the TS and SA heuristics outperformed the LP heuristics and that SA slightly outperformed TS. However, for even their fastest heuristic, the six and seven item problems averaged about 90 to 100 seconds of Sun 3 workstation CPU time. With respect to solution quality, the best method (SA) generated solutions that averaged 12 to 23 percent above the lower bound obtained by solving the LP relaxation of the problem. No comparison to results obtained with single level heuristics was provided.

In Billington et al. (1994) the authors study capacitated multiple echelon serial systems and solve associated lot sizing problems using modified single echelon heuristics such as Dixon and Silver that are applied sequentially to each production echelon (top down). They modified the single level heuristics' feasibility routines to work in multiple echelon environments and used several of the Blackburn and Millen (1982a/b, 1984, and 1985) cost adjustment procedures (e.g., KBB and KCC). Besides limiting their study to serial product structures with no more than 12 end items and 5 echelons (60 items), they also used a simplifying assumption that the items have proportional processing times across echelons, i.e., if item A requires twice as much capacity as item B on echelon 1, then A must also require twice as much capacity as B on all other echelons. This rather severe restriction allows for the implementation of relatively simple multiple echelon feasibility checks. Nevertheless, their results indicate that the Blackburn and Millen cost adjustment procedures can provide a significant enhancement to lot sizing heuristic performance.

Motivations

Unfortunately, little of the CLSP research of the last twenty years has made its way into industry practice. Thus, the general motivation behind the research discussed in this dissertation was to derive and test heuristics directed towards providing practical, medium-range planning tools for dealing with realistically complex, commonly occurring lot sizing problems such as one would find in a typical job shop, production environment.

Dixon and Silver's heuristic has proven to be a quick, computationally efficient heuristic that generates feasible (if possible), good solutions to single stage, multiple item, multiple period, dynamic capacitated lot sizing problems. However, it does not allow for setup times--this severely limits the industrial applicability of Dixon and Silver's heuristic. Consequently, since no other fast heuristics are available to solve these types of more complicated lot sizing problems, a primary goal of this research was to develop a heuristic lot sizing approach that would be competitive with the DS heuristic with respect to computational times and solution quality, as well as be capable of handling the very common problem of lot sizing items with significant setup times. This is a major extension because setup times convert the problem into one with non-linear capacity constraints.

Another primary motivation of the dissertation research was to extend single echelon heuristics so that they are capable of solving realistically sized, multiple echelon, multi-capacitated problems with assembly product structures. This capability is important because most industrial problems exist within the framework of a multiple echelon environment and, unfortunately, industrial lot sizing practice tends to ignore this complexity. That is, each echelon is solved sequentially without considering lower or upper echelon cost impacts; then, laborious trial and error adjustments are made to satisfy any binding capacity constraints. Consequently, a goal was to provide a more powerful, less labor intensive, integrated methodology for dealing with these types of problems.

Finally, the testing of single and multiple echelon heuristics is critical. Previous research has tended to use small, unrealistic problems. Consequently, a primary goal was the development of better or new random problem generators capable of producing realistic lot sizing problems that allow variation in number of items and periods, capacity utilization, average EOQ time between setups, demand variation, and setup time.

CHAPTER I. AN EXTENDED DIXON AND SILVER HEURISTIC FOR SOLVING DYNAMIC, CAPACITATED LOT-SIZING PROBLEMS

A paper published in the *3rd International IE Research Conference Proceedings*

Robert J. McCoy and Douglas D. Gemmill
Industrial and Manufacturing Systems Engineering Department
Iowa State University, Ames, Iowa

ABSTRACT
This paper presents an extension to the Dixon and Silver lot-sizing heuristic that offers improved solution cost performance at a small computational expense.

INTRODUCTION
A common and important industrial inventory control problem involves determining production lot-sizes for multiple items that share a common capacity constraint. Specifically, the problem consists of scheduling for multiple periods the production timing and quantity of multiple items so as to minimize the sum of both setup and linear holding costs while satisfying demands (without backlogs) and adhering to the capacity constraint. The capacity constraint and item demands may be time-varying, but are assumed to be deterministic. A Material Requirements Planning (MRP) context with "rolling horizon" scheduling is one example of a production planning environment where these types of problems must routinely be solved.

Due to the complexity of the problem (it is known to be NP-hard [1]), research attention has primarily focused on the development of heuristics rather than optimization methodologies. Consequently, over the last twenty years, a number of heuristics have been developed. A few of these heuristics are listed below, but for a more detailed review of the problem and associated solution methodologies, the reader is referred to two reviews [2,3].

In 1975, Eisenhut [4] extended the well-known Silver-Meal heuristic to solve capacitated, multi-item problems. Lambrecht and VanderVeken [5] extended the work of Eisenhut to guarantee feasibility (if feasibility is possible) of the resulting lot-sizing solution. In 1981, Dixon and Silver [6] published a relatively simple (and fast), 'greedy' heuristic that built upon previous enhancements to the Silver-Meal heuristic and uses a forward look-ahead to ensure feasibility. Since then, this heuristic has received considerable, favorable attention as a stand-alone solution methodology ([2] and [3]) and as a starting point for more complicated, computationally expensive methodologies. For example, both the Thizy and Van Wassenhove [7] and Diaby et al. [8], Lagrangean Relaxation approaches use Dixon and Silver's solution to initialize dual costs associated with a transportation network formulation of the lot-sizing problem.

In this paper, the general solution methodology of Dixon and Silver will be extended. The resulting Extended Dixon-Silver (EDS) Heuristic, while more complicated than Dixon and Silver's original, is still relatively simple and fast. Furthermore, the extended heuristic generated better solutions for seven out of eight lot-sizing problems taken from the literature. Additionally, better results were obtained on all four of the suitable, "real world" problems found in the literature--three from Dixon and Silver's original paper [6] and the other from Maes and Van Wassenhove [3]. These results are provided later in this paper.

SUMMARY OF THE EXTENDED DIXON-SILVER HEURISTIC

The EDS heuristic is composed of two main algorithms: (1) an algorithm that uses the basic Dixon and Silver Heuristic's logic--but modifies the marginal benefit calculation to allow for the iterative creation of multiple solutions; and (2) the new Forward and Backward Adjustment improvement algorithms. The basic Dixon and Silver algorithm is a greedy heuristic that uses modified Silver-Meal criteria that attempts to minimize the average cost of each lot per unit of time. The modification to the Silver-Meal criteria is necessary only if capacity is limited and there is competition for a limited resource--thus, it is not possible that all lot-sizes can be increased such that the average cost per unit of time is minimized. In this case, the greedy heuristic increases the lot-size of the item for which a single period increase in its time supply results in the largest average cost decrease *per unit of capacity* (e.g., hours) expended. Furthermore, the heuristic uses a look-ahead criterion to achieve feasibility (if possible). The reader may refer to the original article by Dixon and Silver [6] for a more detailed discussion of the basic heuristic and its associated flowchart.

The EDS Heuristic uses the same general solution logic as the original Dixon and Silver Heuristic, hereafter called the 'D&S' Heuristic. However, the EDS implementation differs in two primary ways from the original implementation. The first primary difference relates to a modification of the equations used to calculate the marginal benefit of increasing the lot size of each particular item (referred to as U_i in reference [6]). In general terms, each time the marginal benefit is calculated, a perturbation cost factor is multiplied by the hours of planned, consumed capacity associated with each item's potential lot size. This cost is added to the average cost calculations and often has the effect of slightly modifying the solutions generated by the EDS heuristic, i.e., the perturbation cost factor has the effect of modifying (in close or tie breaker situations) the choice of the next item selected for increasing its lot size. Thus, due to the rapid solution times associated with the heuristic, this allows the user to iteratively generate multiple solutions, with the lowest cost solution selected for implementation. On the basis of experimentation, perturbation cost values in the range of -4.0 to +4.0 were found to be most helpful in solving the data sets provided in 'RESULTS'. However, because the code associated with this perturbation cost factor is space intensive to describe, no additional attention will be devoted to the differences. Rather, the EDS Heuristic code will be provided to interested readers.

The new improvement algorithm is the other primary difference between the EDS heuristic and the original D&S Heuristic. The D&S Heuristic uses a simple improvement subroutine that checks the lot-sizing solution generated by the basic heuristic, starting with period two and proceeding to the end of the planning horizon. It looks for lots that could be eliminated by consolidating them into existing lots that occur in earlier periods with sufficient capacity to cover the additional production quantity. Of course, this consolidation only occurs if the additional holding cost incurred is less than the setup cost saved.

The EDS Heuristic uses a more complex improvement algorithm. This algorithm starts with the feasible lot-sizing solution generated by the basic algorithm and looks for improvement in the final solution cost by adjusting certain lot-sizes (complete or partial) forward and/or backward in time. The first of the two main sections of the improvement algorithm begins by *checking for extra item inventory that is needlessly being carried and could be shifted to a later production period and lower total planned costs.* An example of how this excess inventory for a particular item could be generated by the basic algorithm is as follows: (1) The algorithm uses the basic Dixon and Silver logic (which may include the addition of a perturbation cost factor) to determine lot-sizes for a particular production period, e.g., period one. In determining the lot-sizes for one of the items, e.g., item 'A', the algorithm determines that it is desirable to supply period two's demand for item A by production in period one. That is, the greedy algorithm determines that it is cost beneficial and feasible to eliminate the setup of item A in period two by instead producing in period one. (2) After finishing the determination of lot-sizes in period one for all items and completing the forward-looking feasibility check, the basic algorithm moves on to consider period two and determines the most cost effective lot-sizes. Once again, the algorithm does a forward-looking feasibility check. However, the basic D&S Heuristic may have difficulty at this point--that is, the forward-looking feasibility check may determine that additional current period production, while not cost effective, is required to ensure future production period feasibility. In this example, if the heuristic determines that item A has the smallest marginal increase in average costs per unit of capacity absorbed, then it will schedule production of item A in period two. Therefore, the item A lot-size decision regarding period one is no longer sound because an item A setup is now planned for period two *while* at the same time extra holding costs will be incurred. Thus, holding costs may be reduced by shifting some production from period one to period two.

The *Forward Adjustment Algorithm* is the first section of the improvement algorithm and compensates for the type of situation described above. The detailed flow chart associated with this algorithm is provided in Figure 1. The flow chart is relatively self-explanatory, but step eight requires a brief clarification. Specifically, the algorithm in steps two to seven shifts a maximum of one item per period forward in time one period. Consequently, if any shifting occurred in steps two to seven, then additional shifting within the revised available capacity may be possible. Thus, step 8 directs the algorithm to go back to step one for additional repetition(s).

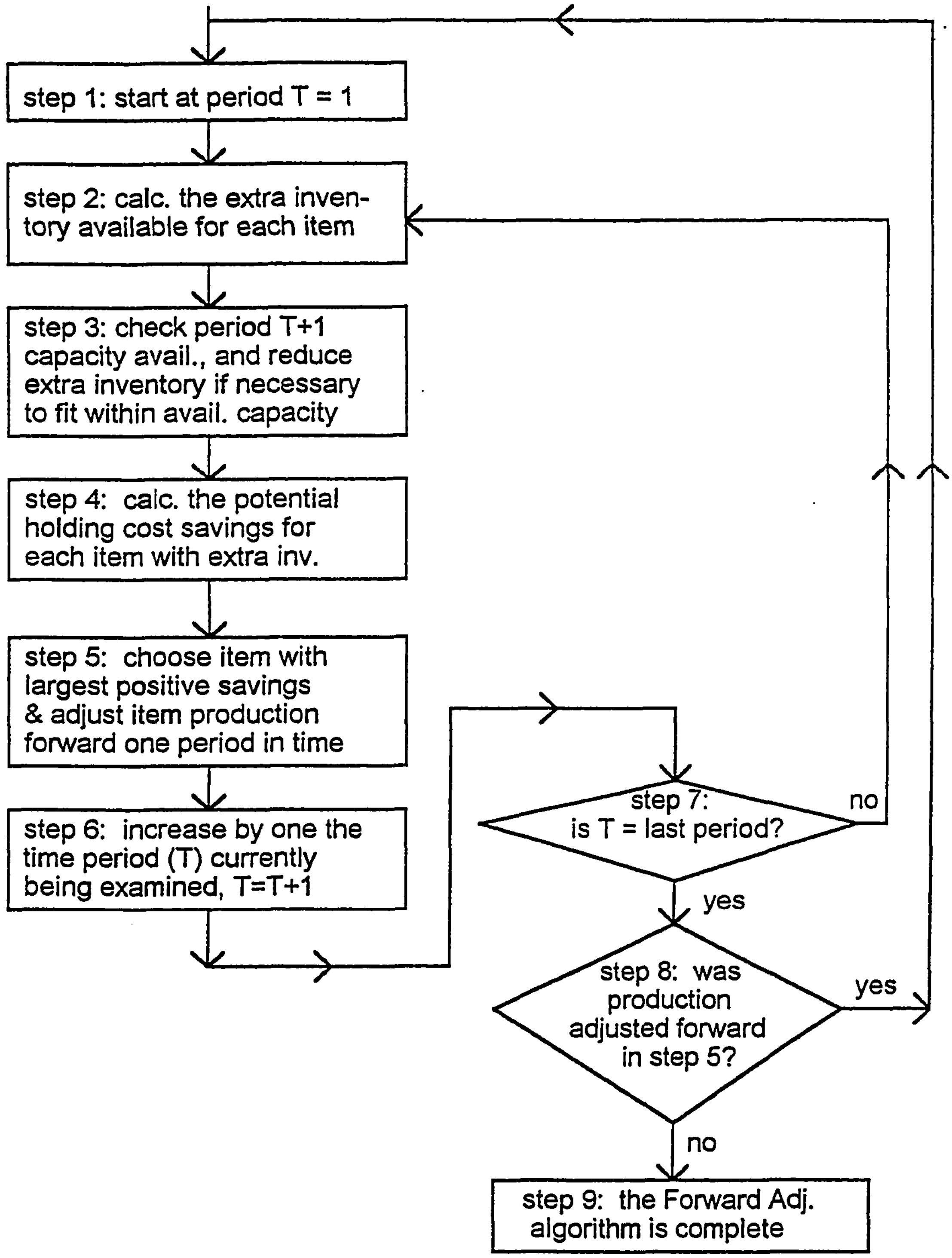

Figure 1: Forward Adjustment Algorithm Flowchart

The second main section of the improvement algorithm is called the *Backward Adjustment Algorithm*. It deals with cost effectively eliminating some production setups by shifting production of one or more items back to an earlier production period. An example of a situation where this iteration of the improvement algorithm may result in reduced costs is the situation discussed above in the Forward Adjustment explanation. That is, capacity constraints have caused the planning formation of a small non-economic production lot that could be shifted in time (in this case backward) and result in a lower overall cost. However, unlike the simple period by period check made by the improvement algorithm of Dixon and Silver, this Backward Adjustment algorithm differs in two major respects: (1) it is a multi-pass algorithm that reviews *all* planned lot-sizes for the largest potential savings; and (2) it does not restrict itself to simply eliminating a particular lot-size and shifting it back in time to a period that already has a lot scheduled. This last difference is particularly important because it opens up improvement possibilities that are not available to the D&S improvement algorithm.

To aid in understanding the Backward Adjustment Algorithm, the detailed flow chart of Figure 2 is offered. Furthermore, additional explanation regarding step 6 is provided: Step 6: Calculate the overall savings (if any) associated with the elimination considered in step 5. The methodology for calculating the savings is as follows:

(A) Calculate the additional holding cost for the item in question resulting from eliminating the future lot-size (in period E) and producing in the current period (T) being analyzed.

(B) Calculate the capacity shortage penalty associated with the elimination of production in period E. If sufficient capacity exists in period T (where the production would be moved), the penalty is zero. However, if the move is only feasible due to cumulative (unused) available capacity, then the calculations are more complex because one must estimate the holding cost penalty associated with moving production of one or more items earlier in time so as to free

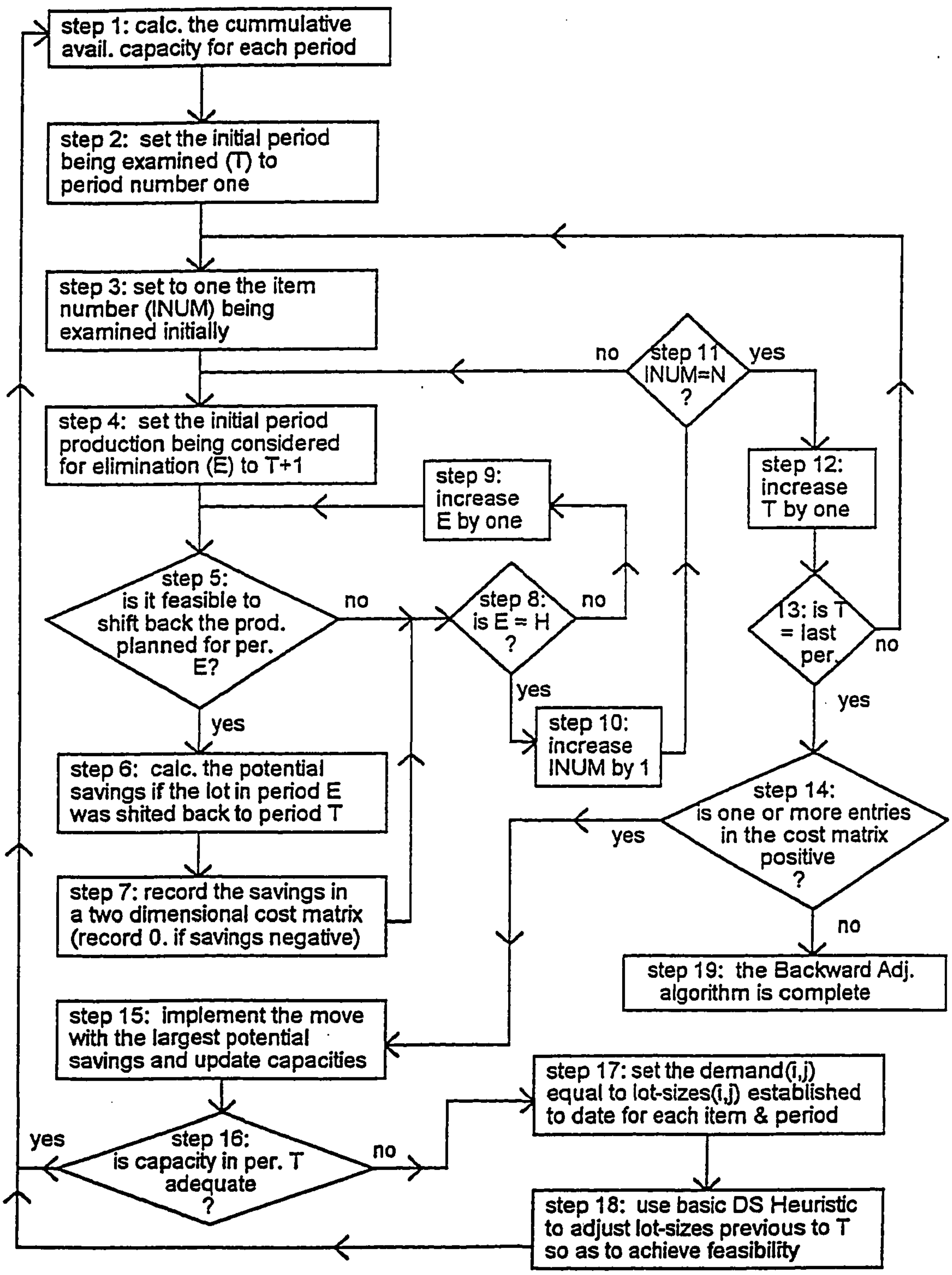

Figure 2: Backward Adjustment Algorithm Flowchart

up sufficient capacity. (Note: estimation is required because, without actually rescheduling all the previous periods, it is not known a priori which item(s) would be moved.) The approach taken by the algorithm is to use the minimum holding cost. This is a best case estimate that will maximize the number of possible changes considered (and increase the potential cost performance and computational requirements of the algorithm).

(C) Subtract the holding costs calculated in A and B from the savings resulting from eliminating the future lot-size (i.e., the setup cost for the item). Divide this amount by the capacity impact of the lot-size in period E being considered for backward movement.

The main sections of the new heuristic have now been described. The configuration of the EDS heuristic is to run the basic algorithm first (with a zero perturbation cost factor), followed by the Forward Adjustment algorithm, then the Backward Adjustment algorithm, and concluding with a review (and possible adjustment) of the lot-sizing solution using the Forward Adjustment algorithm. Then, at the discretion of the user, additional iterations of the algorithm may be run--each iteration using either a positive or negative perturbation value. This iteration will often produce a superior (lower cost) solution at the expense of additional computer time.

RESULTS

A review of the literature identified eight, multi-item, capacitated, single-level, dynamic lot-sizing problems that included published Dixon and Silver solution values and sufficient problem data so that the EDS Heuristic could be applied and its solution cost determined. Three of the problems (DS1-DS3) are 'real world' problems taken from the original Dixon and Silver article ([6], Machines 1 to 3). Four others (TVW1-TVW4) were taken from Thizy and Van Wassenhove [7]. The final problem (MVW) was taken from Maes and Van Wassenhove

[3]. It is reported to be an actual industrial problem supplied by a large Belgian plastic's manufacturer. The eight problems range in size from eight to twenty items and a total of eight to thirteen periods.

The best results of applying multiple iterations of the EDS Heuristic to the eight problems are shown in Table 1. For each problem, a total of 9 iterations was run (with perturbation cost values of -4, -3, ... 3, 4). Also included are the associated solution costs of the EDS Heuristic (with one iteration at a perturbation value of 0.0), the D&S Heuristic, and the optimal solution (for TVW1-4 and DS3).

TABLE 1:
COMPARISON--HEURISTIC SOLUTION QUALITY

Prob. Name	9 Iter. EDS(*)	1 Iter. EDS	D&S	Optimal
TVW1	8480 (- 1)	8650	8710	8430
TVW2	7940 (+4)	8030	7930	7910
TVW3	7610 (+2)	7910	7970	7610
TVW4	7520 (+1)	7630	8000	7520
MVW	12652 (0)	12652	13113	n.a.
DS1	93314 (+4)	94636	95537	n.a.
DS2	71712 (- 4)	72849	73161	n.a.
DS3	5854 (+1)	5867	5944	5808

* ( ) Equals integer perturbation cost value associated with the best solution identified.

As Table 1 shows, on seven out of the eight problems, the EDS Heuristic provided better solutions than Dixon and Silver. Additionally, it appears that the computational advantage of the EDS heuristic is greatest on problems with 'looser' (but still binding) capacity constraints and relatively high setup costs, e.g., TVW3 and TVW4.

Computation times are also an important aspect of heuristic performance. The times relating to the D&S Heuristic have been well documented [3,9]. This heuristic, as well as the EDS

Heuristic, may be categorized as a simple and fast heuristic that requires little computer memory. However, they also exhibit a typical trade-off between solution speed and solution accuracy. That is, the D&S heuristic is faster and has a poorer performance, while the EDS Heuristic is slower, but has a better overall performance. However, on problems of this size, the EDS heuristic is very fast--on a DECstation 5100 workstation, the 9 iterations of each problem could be solved in a *maximum of less than 2 CPU seconds per problem.* In contrast, the optimum solution times (on a VAX 750) range from 34 to 4,909 CPU seconds per problem [10]. These optimal solution times are excessive for the majority of users requiring to solve numerous capacitated lot-sizing problems on a repetitive basis--hence, the importance of heuristic solution procedures.

The relatively fast computational times of the EDS heuristic and the solution cost results shown in Table 1 indicate that the EDS Heuristic is a competitive alternative to the Dixon and Silver heuristic. On the eight problems tested, the EDS Heuristic, with both nine iterations and just one iteration, beat the results of the D&S heuristic on seven out of the eight problems. And, the EDS heuristic with nine iterations provided a total average reduction of 2.9 percent from the eight solutions provided by the D&S heuristic. Furthermore, on the five problems with an available optimal solution value, the EDS heuristic hit the optimal on two of the problems and missed the optimal on the other three by *at most* 0.8 percent. In contrast, the D&S heuristic did not hit the optimal on any of the five, it missed the optimal by 6.4 percent on one problem, and its solutions averaged 3.4 percent over the optimal.

The EDS computer code and the test problems are available upon request.

RESEARCH DIRECTIONS

Work is currently underway to test the new EDS heuristic against additional problem sets and extend its general solution methodology to enable it to handle capacitated lot-sizing problems *with* setup times. Furthermore, future research will investigate extending the EDS heuristic to solve *multiple stage [echelon]*, capacitated lot-sizing problems.

CONCLUSION

In this paper, an extension to a fast and simple heuristic for solving multi-item, multi-period, single-level, dynamic, capacitated lot-sizing problems was presented and applied to eight problems found in the literature. It was shown that the Extended Dixon-Silver (EDS) Heuristic provided better solutions than the original Dixon and Silver Heuristic on seven of the eight problems and, on the five problems with available optimal solutions, it generated solutions that hit the optimal on two of the problems and deviated from the optimal on the other three by at most 0.8 percent. Therefore, it is a reasonable alternative for users wishing to increase their lot-sizing solution accuracy at a minor computational expense.

REFERENCES

[1] Dixon, P.S. (1979) Multi-item lot-sizing with limited capacity. Unpublished Ph.D. dissertation, Dept. of Mgmt. Sci., *U. of Waterloo*, Waterloo, Ontario.

[2] Bahl, H.C. and Ritzman, L.P. and Gupta, J.N.D. (1987) Determining lot sizes and resource requirements. *Opns. Res.* **35**, 329-345.

[3] Maes, J and Van Wassenhove, L (1988) Multi-item single-level capacitated dynamic lot-sizing heuristics: a general review. *J. Opl. Res. Soc.* **39**, 991-1004.

[4] Eisenhut, P.S. (1975) A dynamic lot sizing algorithm with capacity constraints. *AIIE Trans.* 7, 170-176.

[5] Lambrecht, M.R. and VanderVeken, H (1979) Heuristic procedures for the single operation multi-item loading problem. *AIIE Trans.* **11**, 319-326.

[6] Dixon, P.S. and Silver, E.A. (1981) A heuristic solution procedure for the multi-item, single-level, limited-capacity, lot-sizing problem. *J. Opns. Mgmt.* **2**, 23-40.

[7] Thizy, J.M. and Van Wassenhove, L. N. (1985) Lagrangean relaxation for the multi-item capacitated lot sizing problem, a heuristic approach. *AIIE Trans.* **17**, 308-313.

[8] Diaby, M. and Bahl, H.C. and Karwan, M.H. and Zionts, S. (1992) A Lagrangean relaxation approach for very-large scale capacitated lot-sizing. *Mgmt. Sci.* **38**, 1329-40.

[9] Trigeiro, W.W. (1987) A Dual-cost heuristic for the capacitated lot sizing problem. *IIE Trans.*, Mar. '87, 67-72.

[10] Eppen, G.D. and Martin, K.P. (1987) Solving multi-item CLSP using variable redefinition. *Opns. Res.* **35**, 832-848.

CHAPTER II. A FAST HEURISTIC FOR LARGE-SCALE CAPACITATED LOT SIZING PROBLEMS, WITH OR WITHOUT SETUP TIMES

A paper submitted to *Computers and Industrial Engineering*

Robert J. McCoy and Douglas D. Gemmill

Industrial and Manufacturing Systems Engineering Department

Iowa State University, Ames, IA 50011, USA

Abstract

The development and testing of a fast heuristic, called the MG heuristic, is discussed. This new heuristic is capable of solving multiple item, multiple period, dynamic capacitated lot sizing problems, with or without setup times. Comparison testing of the MG heuristic against other leading heuristics used large-scale, realistically sized problems. For even the largest group of problems tested, 4000 items and 25 periods, its approximate average CPU time (on a DECstation 5000/200 workstation) was 1.0 minute. And, for all 216 problems tested, the MG heuristic's average solution costs were just 0.86% higher than the best (Lagrangean relaxation) heuristic against which it was tested, at 0.026 the computation time.

Keywords: Lot Sizing; Production Planning; Material Requirements Planning; and Job Shop.

1. Introduction

The Capacitated Lot Sizing (CLS) problem is an important industrial inventory control problem that involves determining medium-range production planning lot sizes for items that share a common capacity constraint. A material requirements planning (MRP) system is a common example of a situation requiring the frequent solving of CLS problems. Specifically, the problem consists of scheduling for multiple periods the production timing and quantity of multiple items so as to minimize the sum of both setup and linear holding costs while satisfying demands (without backlogs) and adhering to the capacities associated with a single resource. The capacity constraint and product demands may be time-varying, but are assumed to be deterministic, even though they are often based on forecasts. Furthermore, item setup costs are incurred in each period that a particular item is produced. The problem is complicated by the inclusion of item setup times (which consume capacity). For a mathematical definition of the problem, the reader is referred to [1].

These types of production planning problems are extremely complex. The single item, single stage, non-capacitated version of the problem can be solved efficiently using Wagner and Whitin's methodology [2], but just adding a capacity constraint converts the problem to one that has been shown to be NP-hard [3]. Consequently, CLS research has primarily focused on the development of heuristics rather than optimization methodologies. As a result, a number of heuristics have been developed over the last twenty years, but most do not allow for the inclusion of setup times. For a detailed review of the CLS problem without setup times and associated solution methodologies, the reader is referred to two reviews [4 and 5].

In spite of the common occurrence of CLS problems with significant levels of setup times, the list of heuristics available to solve these types of problems is short and is limited to computationally complex heuristics. The list includes TTM heuristic ([1 and 6], using Lagrangean relaxation), Subgradient Lagrangean Relaxation (SLR) heuristic [7 and 8], and Billington's heuristic [9]. The TTM heuristic tested superior to the Billington heuristic [6] and the computational complexity of the Billington heuristic makes it inappropriate for realistically sized CLS problems. Therefore, Billington's heuristic will not be discussed further.

Trigeiro et al. [1] includes significant detail relating to the issues and complexity associated with solving CLS problems that include setup times. Using a factorial design associated with their TTM heuristic testing, they made the following conclusion based on an analysis of the solution gaps (deviation from the Lagrangean lower bound to their solution):

problems with high setup costs relative to holding costs, few items and tight capacity constraints tended to have increased solution gaps and are therefore more difficult problems. Furthermore, they indicated that because large problems are easier to solve than small problems (but require more computation time), previous research based on small problems may not provide results representative of typical (large) CLS problems encountered in industrial settings. They concluded that simple, fast heuristic algorithms may thus be more useful in practice than suggested by previous research findings.

Trigeiro et al. [1] also discussed the difficulty of solving problems with setup times. They wrote, "It is a grave error to state that setup time is a simple extension of setup cost. For example, the problem of determining whether a feasible solution exists goes from trivial to NP complete when setup time is added." Therefore, the issue of determining feasibility is significant, because there is no easy way to determine if a feasible solution exists for a given problem that includes setup time.

One prime motivation behind the research described in this paper was to help fill a perceived void in available CLS heuristics. That is, the need for a fast, robust heuristic that would provide a practical, medium-range planning tool for solving realistically complex, commonly occurring lot sizing problems, with (or without) significant levels of setup time, such as one would find in a typical job shop environment. Computationally complex heuristics, such as the TTM heuristic, are available that do explicitly account for setup times. Their primary drawback is excessive computation time for realistically sized CLS problems. Currently, for large-scale problems without setup times, the Dixon and Silver heuristic [10], among others, has filled the need for a fast heuristic that can be expected to generate a reasonably good solution in a reasonable amount of computation time. From a practitioner's viewpoint in an actual production setting, given the uncertainties usually associated with a typical "real world" production problem (e.g., approximate setup and holding costs, uncertain demand forecasts, and approximate capacity constraints), the DS solutions are generally adequate. However, in many production environments (and even in many factories that strive for JIT adherence), setup times are significant and must be taken into account. Unfortunately, current industry practice associated with constructing a feasible production schedule that reflects setup times often involves a significant amount of labor intensive, iterative 'trial and error' effort -- thus the need for a fast CLS heuristic capable of solving problems with setup times.

Another motivation behind this research was to test the solution quality and computational performance of several leading CLS heuristics, as well as the new MG heuristic, on randomly generated problems more representative of the "real world" (in terms of both size and problem characteristics) than those commonly used in past research. In the following sections, a description of the new heuristic, the MG heuristic, and its solution methodology will first be provided. Then, the testing process, including random problem generation, and associated results will be discussed.

2. The MG Heuristic

The new MG heuristic for solving large-scale CLS problems has three main sections: (A) Wagner-Whitin (WW) algorithm and a feasibility seeking subroutine (referred to below as the "smoothing" subroutine); (B) a modified Dixon and Silver heuristic (only used when the previous procedure, section A, does not result in an initial feasible solution); and (C) improvement algorithms. The MG heuristic starts by ignoring the capacity restrictions and uses the basic WW dynamic programming algorithm to optimally solve the resulting uncapacitated lot sizing problem. (Note: minor improvements in overall computational speed could be obtained by incorporating one of the dynamic programming refinements recently suggested, e.g., [11].) Then, after reinstalling the capacity restrictions, a feasibility check is performed. Typically, except for trivial problems, the WW solution is infeasible. If it is feasible, the MG heuristic ends with the retained WW (optimal) solution.

To seek feasibility using the initial WW solution as the starting point, the MG heuristic uses a modified version of the "smoothing" subroutine from the TTM heuristic, Trigeiro et al. [1]. This modified code does not utilize dual costs of capacity, as does the original code. It merely seeks to obtain a feasible solution that satisfies capacity constraints using the WW generated lot sizes as a starting point. It does this by making, if necessary, multiple backward and forward passes to shift full or partial lot sizes out of periods that are overloaded with respect to capacity. (For additional details relating to this methodology, refer to the above reference article.) However, this smoothing subroutine does not guarantee a feasible solution, particularly for tightly constrained problems with significant levels of setup time. If the "smoothed" WW solution is not feasible, the MG heuristic seeks an initial feasible solution by using a modified version of the DS heuristic. Our modifications to the basic DS heuristic enable it to solve problems with setup time and are discussed in Appendix I.

For large-scale CLS problems, the modified DS heuristic is only used when the initial WW and smoothing process does not identify a feasible solution. This does not commonly occur for problems with<u>out</u> setup time, but when it does, use of the DS heuristic guarantees a feasible solution will be found (if feasibility is possible). For CLS problems <u>with</u> setup time, using the modified DS heuristic to obtain feasibility is more complicated, but the general idea associated with its use is relatively simple. On marginally feasible problems with setup times, achieving feasibility is enhanced by reducing the number of setups, thereby reducing total setup time requirements. The MG heuristic uses the modified DS heuristic and temporarily increases the setup costs for all items. Then, the DS solution procedure will tend to maximize the capacity use at the start of the production schedule and, because of the inflated setup costs, it will tend to reduce the number of setups scheduled. Thus the amount of capacity expended on setups is decreased and the likelihood of identifying an initial feasible starting solution is enhanced. Then, if an initial feasible solution is identified, the original setup costs are reinstated and the improvement algorithms (discussed below) are applied.

The third (and last), main section of the MG heuristic involves the use of three improvement algorithms. Their use follows the attainment of an initial feasible solution by the WW and smoothing algorithms or, if required, the modified DS heuristic. These improvement algorithms are run in series and include a modified version of the improvement algorithm supplied with the basic DS heuristic and new forward and backward adjustment improvement algorithms. These algorithms adjust the lot sizes of the initial feasible solution with the objective of maintaining feasibility while lowering total costs (setup and holding). After running the improvement algorithms, the MG heuristic terminates.

The first of the improvement algorithms used by the MG heuristic is a modification of the DS improvement algorithm supplied by Dr. Van Wassenhove, which implements possibilities 1 and 4 of the improvement algorithm described in [10]. The reader is referred to this article for an overview of the improvement logic associated with the algorithm. We modified the basic DS improvement algorithm to account for the impact of setup time and improve its computational efficiency. For an overview of the logic associated with the improvement in computational efficiency, see Appendix II.

In addition to the modified improvement algorithm described above, the MG heuristic uses two other improvement algorithms. These algorithms also start with the best, feasible lot sizing solution generated to that point in time and look for improvement in the final solution cost by adjusting certain lot sizes (complete or partial) forward or backward in time. The first of these improvement algorithms, called the 'Forward Adjustment' algorithm, begins by checking for extra item inventory that is needlessly being carried and that could be shifted to a later production period to lower total planned costs. It is a period by period (starting in period one), multiple pass, review. It seeks to lower total costs by reducing holding costs, possibly at the expense of additional setup cost and time.

The second improvement algorithm is called the 'Backward Adjustment' algorithm. It deals with cost effectively eliminating some production setups by shifting production of one or more items back to an earlier production period. Unlike the basic improvement algorithm of Dixon and Silver, the Backward Adjustment algorithm does not restrict itself to simply eliminating a particular lot size and shifting it back in time to a period that already has a lot scheduled and that has adequate capacity. If adequate incremental capacity exists and it appears cost effective to do so, the algorithm will overload a particular period's capacity and then use the modified DS heuristic to regain feasibility by shifting production into an earlier production period(s). This opens up improvement possibilities that are not available to the basic DS improvement algorithm. Detailed discussion of both the forward and backward improvement algorithms, as well as flowcharts, are provided in [12].

3. Heuristic Testing

In addition to the MG heuristic, testing incorporated the following CLS heuristics: DS [10], DPA [13], and TTM [1 and 6]. Large-scale testing of the TTM heuristic was desired -- previously reported testing was limited to problems of no more than 36 items. Computer implementations of the DS and DPA heuristics were supplied by Dr. Luk Van Wassenhove, INSEAD. The TTM heuristic was supplied by Dr. Trigeiro, MITRE Corporation.

Based on previous, initial testing [14], other CLS heuristics such as SLR [7 and 8], LV [15] and TVW [16] were not included in the testing discussed in this paper. Our preliminary testing indicated that the LV heuristic, while marginally faster than the DS heuristic, tended to generate solutions with significantly higher cost than those generated by the DS heuristic. The TVW heuristic was omitted because of slow computation times and and the superiority of the

TTM heuristic. The performance of the SLR heuristic, while designed for large-scale testing, was disappointing. Our preliminary testing indicated that it works well on some problems, but its performance is variable. Without resorting to a 'trial and error' approach to setting the user supplied run control parameters, the SLR heuristic failed to consistently identify feasible solutions to feasible CLS problems (without setup time). We concluded that the heuristic's run control parameter settings are critical to the performance of the heuristic and are of questionable robustness over a variety of CLS problem characteristics.

The random problem generator used for our testing was a modified version of the generator used by Diaby et al. [7] for their large-scale testing. The reader is referred to their article for details relating to the basic generator. The significant modifications to the generator are: (A) Production time, rather than being constant at 1.0 per unit demand, was allowed to vary normally around a mean value of 1.0; (B) The user was provided the capability of influencing the average setup times, setup costs and carrying costs values for a particular problem -- thus the user can approximately specify, using program run control parameters, the average level of setup time and the average time between orders (TBO). Note: TBO is defined using the standard mathematical definition, see [5]; (C) Setup cost calculations for each item were modified to provide positive correlation between the level of setup time and the setup cost. Thus, the setup costs were generated as follows:

AVE(I)=(ST(I) * 10. + RM * 200.) * SCF; and

SC(I)=20. + AVE(I) * (1.0 + Z / 6.0);

where: I = item number; ST = setup time; RM = random number;

SCF = setup cost factor; SC = setup cost; Z = normal random variate;

and (D) The methodology of calculating each period's capacity constraint was modified to provide more realistic capacity profiles (refer to [14]). The original generator tends to lump a disproportionate amount of total capacity into the first period, with the rest of the production period's capacity set to a lower, constant level. Our modifications reduce the amount of first period (upward) bias and generates production period capacities that vary normally around an overall CLS problem average. Appendix III provides an overview of our capacity determination methodology.

Our testing was designed to determine the relative performance of the tested heuristics on randomly generated large-scale problems. The testing was conducted on four major categories of problems: (1) low TBO problems without setup time; (2) high TBO problems without setup; (3) low TBO problems with setup time; and (4) high TBO problems with setup time. Within each of the four categories, the problem generator was used to vary the problem size (from 250 to 4000 items and from 25 to 50 periods) and the approximate, overall, capacity utilization level. This allowed us to study the effect of problem size and capacity utilization level on solution quality and computation time.

Use of problems without setup time allowed us to test the DS and DPA heuristics along with the MG and TTM heuristics. The two categories of problems with setup time included only problems with setup time utilization greater than 10%, i.e., in the final solution, more than 10% of the total capacity available was used to setup items. This ensured that the total setup time was at a significant level.

TBO was segregated into two levels. The low level was defined as TBO less than 2.0. The high level was defined as TBO between the value of 2.5 and 5.0. The reason for the gap between 2.0 and 2.5 was to provide a significant difference between the two levels.

Within each of the four setup/TBO categories, 54 problems were randomly generated, for a total of 216 problems. Of the 54 problems in each category, thirty were 250 items and 25 periods in size. Of these thirty problems, in order to assess the impact of overall capacity utilization on solution quality and time, we generated twelve high utilization, twelve medium utilization and six low utilization problems. Average capacity utilization was calculated over all but the final four production periods. The last four periods were omitted to reduce the effects of batching at the end of the production horizon. Based on preliminary testing, we set the following definitions of capacity utilization ranges: low -- utilization in the range of 70% to less than 80%; medium -- utilization in the range of 80% to less than 90%; and high -- utilization in the range of 90% to 98%. Problems with average utilization less than 70% were thrown away because they tended to be trivial problems. On the other end, problems with over 98% average utilization levels (over a long time horizon) were marginally feasible problems that were not commonly generated and tended to produce erratic heuristic performances. Thus, their inclusion would have created outliers, both with respect to solution quality and computation time. The impact of high capacity utilization on solution quality will be discussed later in more detail.

In addition to the thirty 250 * 25 (item * period) problems generated in each category, twelve 1000 * 25, six 1000 * 50 and six 4000 * 25 problems were also generated to assess the impact of problem size on solution quality and particularly solution time. The capacity utilization target for these problems was a moderate 88%, and to facilitate comparison between size groupings, problems falling outside the range of 80% to 95% were discarded.

In order to generate the 216 problems needed for testing, the random problem generator's run control parameters were set as follows:

(A) The demand variability factor was set to a high level, i.e. two times the variability used by Diaby et al. [7]. This increased the "lumpiness" of the item demands, which is more representative of "real life" MRP component demand schedules [17].

(B) For problems specified to have setup time, the setup time factor was set to generate problems that utilized, on average, about 22% of capacity (the actual range of values for the 108 problems with setup time was from 10-37%).

(C) The setup cost and carrying cost factors were varied over a range of values that tended to generate problems with average TBO values within the category ranges previously specified. Note that item TBO values often varied significantly around the overall problem TBO average.

(D) The capacity constraint tightness factor was adjusted so as to influence the overall capacity utilization level and "fall" into the aforementioned categories.

The use of the capacity constraint tightness factor could not be used to specify a particular utilization level. This is due to random variation in the individual period capacity constraints, variation in the setup and production times per item, and varying item setup and holding costs (which influences the number of item setups justified). Thus after each problem was generated, the MG heuristic was used to determine the amount of capacity utilization. For a particular setup/TBO category, if the calculated utilization fell outside the ranges specified above or if the required number of problems in a particular utilization range had already been obtained, the problem was discarded. The one exception to this is in the case of problems with setup times and a problem size of 250 items. Then, both the MG and TTM heuristics were applied to the problem to evaluate their relative effectiveness in finding a feasible solution to constrained problems.

For problems that include significant levels of setup time, determining if problems are feasible is a major complication. As indicated earlier in the paper, the issue of determining feasibility in this environment is an NP-Complete problem. Trigeiro et al. [1] and Diaby et al. [7] handled this problem in the following manner. They use the output of their own, independently, developed random problem generator and apply their heuristic (only) to the resulting problems. Then, if their method does not generate a feasible solution, they throw the problem away -- this of course biases their results, but the extent of the bias is unknown. This bias and the number of CLS problems with questionable feasibility can be reduced by applying the guidelines contained in Appendix IV. Also, by applying both the MG and TTM heuristics to the problems with setup time, the testing included in this paper is less biased.

4. Heuristic Test Results

Tables 1, 2, 3, and 4 provide the test results for each of the four major problem categories. The results for each category include: (1) solution quality, expressed as the average deviation between the solutions of the MG heuristic and the other tested heuristics; and (2) average computation time (in CPU seconds) on a DECstation 5000/200 workstation. The heuristic abbreviations are as previously indicated. The 'LB gap' (lower bound gap) is defined as the percent difference between the TTM Lagrangean lower bound and the best solution achieved. Also provided in the tables are the range of solution ratios (where, for example, MG/TTM represents the MG solution cost divided by the TTM solution cost) and the CPU time ratios (where, for example, TTM/MG represents the average TTM computation time divided by the average MG time).

With respect to overall solution quality, the computationally complex TTM heuristic outperformed the other heuristics tested. A number of paired-t statistical tests, at a 95% confidence level, was conducted on the results included in each of the four categories of problems associated with the data in Tables 1 to 4. These tests all indicated a *statistically significant* difference between the solution quality of the TTM heuristic and the other heuristics tested (However, whether the differences are of *practical significance* is another issue that must be determined by the user).

With respect to theoretical lower bounds, the TTM's solution quality is also quite good. On average, for the 108 problems with<u>out</u> setup time, its solutions were less than one percent above the lower bound cost. For the 108 problems <u>with</u> setup times, the corresponding value

TABLE 1: LOT-SIZING PROBLEMS WITHOUT SETUP TIMES
LOW TBO (0.50-2.0)

COMPARISON OF THE MG, TTM, DS AND DPA HEURISTICS:

• SOLUTION QUALITY:

ITEMS & PERIODS	NO. OF PROB.	RANGE OF SOLUTION RATIOS MG/TTM	MG/DS	MG/DPA	AVERAGE PERCENT DEVIATION MG vs. TTM	MG vs. DS	MG vs. DPA	LB GAP
250 & 25								
UTIL: HIGH	12	0.995-1.055	0.980-1.024	0.975-0.993	1.04	-0.85	-1.48	1.85
UTIL: MED	12	1.000-1.025	0.979-1.004	0.981-0.998	0.87	-0.94	-0.97	0.18
UTIL: LOW	6	1.001-1.005	0.977-0.984	0.984-0.996	0.31	-2.12	-0.83	0.02
1000 &25*	12	1.002-1.020	0.981-1.00	n.a.	0.92	-1.17	n.a.	0.45
1000 & 50*	6	0.998-1.008	0.975-0.985	n.a.	0.19	-1.87	n.a.	0.98
4000 & 25*	6	0.998-1.014	0.974-0.992	n.a.	0.48	-1.67	n.a.	0.68
	54			GRAND AVERAGE	**0.74%**	**-1.29%**	**-1.15%**	**0.74%**

• COMPUTATION TIME:

ITEMS & PERIODS	NO. OF PROB.	AVERAGE (DEC 5000) CPU SECONDS MG	TTM	DS	DPA	CPU TIME RATIO TTM/MG	DS/MG	DPA/MG.
250 & 25								
UTIL: **	24	2.6	75.1	7.5	164.7	28.9	2.9	63.3
UTIL: LOW	6	2.5	68.0	6.8	204.5	27.2	2.7	81.8
1000 & 25*	12	12.3	400.9	91.7	n.a.	32.6	7.5	n.a.
1000 & 50*	6	35.5	1946.1	586.6	n.a.	54.8	16.5	n.a.
4000 & 25 *	6	35.2	3490.5	1641.0	n.a.	99.2	46.6	n.a.
	54				AVERAGE	**40.2**	**10.3**	**67.0**

* UTILIZATION IN THE RANGE OF 80-95%
** BOTH MEDIUM AND HIGH UTILIZATION

TABLE 2: LOT-SIZING PROBLEMS WITH SETUP TIMES
LOW TBO (0.50-2.0)

COMPARISON OF THE MG AND TTM HEURISTICS:

- **SOLUTION QUALITY:**

PROBLEM DESCRIPTION	NO. OF PROB.	SOLUTION RATIO (MG/TTM) AVERAGE	SOLUTION RATIO (MG/TTM) RANGE	AVERAGE PERCENT DEVIATION MG vs. TTM	AVERAGE PERCENT DEVIATION LOWER BOUND
250 ITEMS * 25 PERIODS					
UTILIZATION: HIGH	12	1.0197	1.0002-1.0380	1.97	0.67
UTILIZATION: MED.	12	1.0025	0.9994-1.0080	0.25	0.13
UTILIZATION: LOW	6	1.0003	0.9980-1.0007	-0.03	0.27
1000 ITEMS * 25 PERIODS*	12	1.0142	1.0019-1.0625	1.14	0.10
1000 ITEMS * 50 PERIODS*	6	1.0037	1.0013-1.0080	0.37	0.05
4000 ITEMS * 25 PERIODS*	6	1.0050	1.0002-1.0201	0.50	0.14
	54		GRAND AVERAGE	**0.85%**	**0.25%**

- **COMPUTATION TIME:**

PROBLEM DESCRIPTION	NO. OF PROBLEMS	AVERAGE (DEC 5000) CPU SEC. MG	AVERAGE (DEC 5000) CPU SEC. TTM		CPU TIME RATIO (TTM/MG)
250 ITEMS * 25 PERIODS					
UTILIZATION: MED&HIGH	24	3.3	113.8		34.5
UTILIZATION: LOW	6	2.3	73.0		31.7
1000 ITEMS * 25 PERIODS*	12	13.3	505.1		38.0
1000 ITEMS * 50 PERIODS*	6	37.4	2013.5		53.8
4000 ITEMS * 25 PERIODS*	6	89.4	2620.3		29.3
	54			AVERAGE	**36.6**

* UTILIZATION IN THE RANGE OF 80-95%

TABLE 3: LOT-SIZING PROBLEMS WITHOUT SETUP TIMES
HIGH TBO (2.50-5.00)

COMPARISON OF THE MG, TTM, DS AND DPA HEURISTICS:

• **SOLUTION QUALITY:**

ITEMS & PERIODS	NO. OF PROB.	RANGE OF SOLUTION RATIOS: MG/TTM	MG/DS	MG/DPA	AVERAGE PERCENT DEVIATION: MG vs. TTM	MG vs. DS	MG vs. DPA	LB GAP
250 & 25								
UTIL: HIGH	12	0.992-1.012	0.966-1.002	0.875-0.989	0.24	-1.07	-5.71	2.09
UTIL: MED	12	1.000-1.005	0.986-0.999	0.925-0.998	0.21	-0.70	-1.84	0.51
UTIL: LOW	6	0.999-1.001	0.988-0.999	0.972-0.997	0.03	-0.77	-0.90	0.07
1000 &25*	12	1.000-1.010	0.990-0.998	0.926-0.997	0.20	-0.66	-2.06	0.67
1000 & 50*	6	1.000-1.005	0.987-0.997	n.a.	0.17	-0.66	n.a.	0.54
4000 & 25*	6	1.002-1.009	0.991-0.996	n.a.	0.34	-0.60	n.a.	0.98
	54			GRAND AVERAGE	**0.20%**	**-0.77%**	**-2.87%**	**0.90%**

• **COMPUTATION TIME:**

ITEMS & PERIODS	NO. OF PROB.	AVERAGE (DEC 5000) CPU SECONDS: MG	TTM	DS	DPA	CPU TIME RATIO: TTM/MG	DS/MG	DPA/MG
250 & 25								
UTIL: **	24	3.4	81.5	3.2	12.8	24.4	0.9	3.8
UTIL: LOW	6	2.0	35.7	4.0	29.0	17.9	2.0	14.5
1000 & 25*	12	9.5	415.5	54.3	307.6	43.7	5.3	29.9
1000 & 50*	6	37.3	1992.5	269.2	n.a.	53.4	7.2	n.a.
4000 & 25 *	6	68.0	3304.6	838.3	n.a.	48.6	12.3	n.a.
	54				AVERAGE	**33.9**	**4.0**	**12.8**

* UTILIZATION IN THE RANGE OF 80-95%
** BOTH MEDIUM AND HIGH UTILIZATION

TABLE 4: LOT-SIZING PROBLEMS WITH SETUP TIMES
HIGH TBO (2.50-5.00)

COMPARISON OF THE MG AND TTM HEURISTICS:

- **SOLUTION QUALITY:**

PROBLEM DESCRIPTION	NO. OF PROB.	SOLUTION RATIO (MG/TTM) AVERAGE	SOLUTION RATIO (MG/TTM) RANGE	AVERAGE PERCENT DEVIATION MG vs. TTM	AVERAGE PERCENT DEVIATION LOWER BOUND
250 ITEMS * 25 PERIODS					
UTILIZATION: HIGH	12	1.0251	1.0028-1.1490	3.04	0.57
UTILIZATION: MED.	12	1.0181	1.0001-1.1400	1.81	0.12
UTILIZATION: LOW	6	1.0027	1.0012-1.0042	0.27	0.02
1000 ITEMS * 25 PERIODS*	12	1.0136	1.0036-1.0425	1.36	0.13
1000 ITEMS * 50 PERIODS*	6	1.0037	1.0016-1.0287	0.87	0.10
4000 ITEMS * 25 PERIODS*	6	1.0108	1.013-1.0252	1.08	0.09
	54		GRAND AVERAGE	1.63%	0.21%

- **COMPUTATION TIME:**

PROBLEM DESCRIPTION	NO. OF PROBLEMS	AVERAGE (DEC 5000) CPU SEC. MG	AVERAGE (DEC 5000) CPU SEC. TTM		CPU TIME RATIO (TTM/MG)
250 ITEMS * 25 PERIODS					
UTILIZATION: MED&HIGH	24	2.5	110.0		44.0
UTILIZATION: LOW	6	2.4	100.0		41.7
1000 ITEMS * 25 PERIODS*	12	12.3	478.0		38.9
1000 ITEMS * 50 PERIODS*	6	38.2	2367.6		62.0
4000 ITEMS * 25 PERIODS*	6	58.8	2697.2		45.9
	54			AVERAGE	44.8

* UTILIZATION IN THE RANGE OF 80-95%

was less than one-quarter percent. That solution gaps decrease for problems with setup time is consistent with the findings of Trigeiro et al [1].

On the 108 (low and high TBO) problems without setup times (see Tables 1 and 3), the MG heuristic outperformed both the DS and DPA heuristics by about 1% and 2%, respectively. In general, the DPA heuristic tended to perform poorly compared to the other heuristics tested -- particularly for low TBO problems with high capacity utilization, where its solutions averaged 5.71% higher than the MG heuristic on the problems (sized 250 * 25) to which it was applied. On the other hand, the relative performance of the DS heuristic deteriorated on problems with low capacity utilization and high TBO (see Table 3). This finding is consistent with Maes and Van Wassenhove [5] whose results indicated that when comparing the DS and DPA performance, the DPA performs best on low utilization problems with sparse demands (high TBO) and the DS heuristic performs better on high utilization problems, regardless of TBO level. Since our test problems tended to be relatively tightly constrained, it is not surprising that the DS heuristic outperformed the DPA heuristic.

As indicated, the MG heuristic typically outperformed the DS heuristic for both high and low TBO categories of problems without setup time. Furthermore, as will be discussed in greater detail below, the MG required less computation time. However, in analyzing the data associated with the high TBO problems, we found that for the problems with both average TBO and capacity utilization at the highest end of their respective ranges, the DS solutions tended to be competitive with those of the MG heuristic.

The impact of capacity utilization on the lower bound gap is very apparent. That is, the higher the capacity utilization, the higher the lower bound gap and, in general, the greater the performance differences between the heuristics.

The impact of problem size on the lower bound gap is inversely related, i.e. larger problems tend to have smaller gaps [1]. This impact can be seen in the results relating to the problems with setup time (Tables 2 and 4). However, on problems without setup time, this impact is somewhat difficult to see. This may be accounted for by random differences in average capacity utilization for each of the size groupings (i.e., capacity utilization effects may have masked problem size effects).

One area of MG heuristic weakness was identified by our testing. For marginally feasible, high TBO problems with setup times, the MG performance relative to the TTM heuristic decreases. These problems can be easily identified by applying the MG heuristic and noting which problems, in order to obtain feasibility, require the use of the modified DS heuristic using temporarily inflated setup costs. When the MG heuristic was required to use its DS heuristic based feasibility attainment procedure, the resulting solutions tended to provide the greatest deviation from the TTM heuristic's solution values. However, this procedure tends to do well finding a feasible solution to marginally feasible problems. In fact, in the process of obtaining the 108 (high and low TBO) problems with setup time, it failed to find a feasible solution to just one problem that was feasible (as determined by using the TTM heuristic).

There were significant differences between the four heuristics' performances with respect to computation time. For three of the four major categories of problems tested, the TTM heuristic, not surprisingly, was the slowest tested. For the 4000 * 25 problems, over all four categories, its average computation time was over 3000 seconds (50 minutes). In contrast, the MG heuristic averaged about 1 minute to solve these same problems. For the fourth major category, high TBO without setup time, the DPA heuristic was significantly slower than the other heuristics tested. For example, on the 250 * 25 item problems, it required 67 times the MG heuristic computation time. Furthermore, its computational complexity is poorer than the other heuristics. Therefore, because the solution quality of the DPA answers does not justify the computation expense, it was not tested on the larger sizes of problems.

While the MG heuristic uses the WW algorithm, it is possible to decrease its average computation time by using the Silver-Meal (SM) algorithm in its place. So, we experimented with a version of the MG heuristic that used the SM algorithm in place of the WW algorithm. This resulting heuristic was used on a subset of the problems tested and approximately a 10 to 20 percent average reduction in total computation time was achieved, depending on the length of the planning horizon. However, there was also a decrease in solution quality associated with using the SM algorithm, rather than WW. For example, on the six, 1000 * 50, low TBO problems tested, use of the SM algorithm in the MG heuristic reduced the total average computation time by about 16% (i.e., 6 seconds), while increasing the average solution deviation between the TTM and the MG heuristics from 0.37% to 0.63%. But, it was also noticed that use of the SM algorithm could occasionally lead to a better solution and/or increase total MG computation time (both results due to the impact of the improvement algorithms).

Prior to our testing, we anticipated that the DS heuristic would be the fastest of the heuristics. Thus, the relatively slow computation time of the DS heuristic, particularly on high TBO problems, was surprising. Consequently, we analyzed the DS times in greater detail. Of the total DS computation time, about 70% was due to its improvement algorithm. Therefore, a significant time reduction would be achieved by eliminating this algorithm. However, the average improvement resulting from utilizing the improvement algorithm was about 1.4% and 0.8% for the high and low TBO problems, respectively. As a result, eliminating the algorithm would more than double the percent deviation between the DS and MG heuristics, and the solution time of the basic DS heuristic would still be greater than that of the MG heuristic.

Generally, with respect to both solution quality and computation time, the MG heuristic dominated the performance of the DPA heuristic and, on average, it outperformed the DS heuristic. As previously noted, it was only for problems with both very high TBO and capacity utilization that the DS heuristic's solutions were competitive, but always at the expense of additional computation time. Furthermore, in comparison to the TTM heuristic, the MG heuristic tended to generate similiar solution costs at a small fraction of the TTM computation time. However, for problems with significant setup time, high TBO, and very high capacity utilization levels (i.e., marginally feasible problems), the TTM heuristic tended to produce significantly better solutions, but always at a major computational increase.

The code used in our testing as well as the problem generator are available from the authors.

5. Conclusion

This paper presented the MG heuristic, a fast heuristic for solving CLS problems, with or without setup time. Also presented was large-scale testing that evaluated the performance of the MG heuristic and three other leading heuristics on realistic CLS problems. Testing used 216 randomly generated problems that included several groupings of problem sizes as well as varied levels of capacity utilization and average TBO. One-half the problems tested were problems with significant levels of setup time.

Test results were favorable for the MG heuristic. On the randomly generated problems with<u>out</u> setup times, overall results of the testing (for both high and low TBO problems) indicate that the MG heuristic yielded cost solutions 1.03% better, on average, than the next best fast heuristic (DS), and it did so at a reduced computational expense. Furthermore, on the same problems, the MG heuristic achieved solutions that averaged just 0.47% above the solutions achieved by the computationally complex heuristic tested (TTM), and on average it obtained the solutions at 0.027 the computation time.

On the problems with setup time, the MG heuristic also performed well. In comparison with TTM, it generated solutions that were on average (for both low and high TBO problems) 1.24% higher, at only 0.024 the average computation time.

Overall, on all 216 problems tested, the MG heuristic achieved feasible cost solutions just 0.86% higher than the TTM heuristic. Furthermore, for even the largest problems tested (4000 items and 25 periods), the computation time for the MG heuristic averaged (over 24 problems) about one minute of CPU time on a DECstation 5000/200 workstation. This indicates that the MG heuristic is fast enough and accurate enough for most "real world" CLS problems, with or without setup time.

Acknowledgments:

The authors would like to thank Doctors Trigeiro, Diaby, Thizy and Van Wassenhove for their programs, problem generators and CLS problems.

References

1. Trigeiro, W.W., Thomas, L.J. and McClain, J.O., 1989. "Capacitated lot sizing with setup times", Management Science, vol. 35, no. 3, 353-366.
2. Wagner, H. and Whitin, T.M., 1958. "A dynamic version of the economic lot size model", Management Science, vol. 5, no. 2, 89-96.
3. Dixon, P.S., 1979. "Multi-item lot-sizing with limited capacity", Unpublished Ph.D. dissertation, Dept. of Management Sci., U. of Waterloo, Waterloo, Ontario.
4. Bahl, H.C., Ritzman, L.P. and Gupta, J.N.D., 1987. "Determining lot sizes and resource requirements : a review", Operations Research, vol. 35, 329-345.

5. Maes, J. and Van Wassenhove, L., 1988. "Multi-item single-level capacitated dynamic lot-sizing heuristics: a general review", Journal of the OR Society, vol. 39, 991-1004.

6. Trigeiro, W.W., 1987. "A Dual-cost heuristic for the capacitated lot sizing problem", IIE Transactions, March, 67-72.

7. Diaby, M., Bahl, H.C., Karwan, M.H. and Zionts, S., 1992. "A Lagrangean relaxation approach for very-large scale capacitated lot-sizing". Management Science 38, 1329-40.

8. Diaby, M., Bahl, H.C., Karwan, M.H. and Zionts, S., 1992. "Capacitated lot-sizing and scheduling by Lagrangean relaxation", European Journal of Operational Research, 59, 444-458.

9. Billington, Peter J., 1983. "Mathematical approaches to capacity-constrained MRP systems: review, formulation and reduction", Management Science, 29, 1126-1141.

10. Dixon, P.S. and Silver, E.A., 1981. "A heuristic solution procedure for the multi-item, single-level, limited-capacity, lot-sizing problem", Journal of Operations Management, vol. 2, no. 1, August, 23-40.

11. Federgruen, A. and Tzur, M, 1989. "A simple forward algorithm to solve dynamic lot sizing models with n periods in *O(n log n)* time", working paper, Graduate School of Business, Columbia University, New York.

12. McCoy, R.J. and Gemmill, D.D., 1994. "An Extended Dixon and Silver heuristic for solving multi-item, dynamic, capacitated lot sizing problems", Proceedings of the 3rd Annual Industrial Engineering Research Conference, 611-616, Atlanta.

13. Dogramaci, A., Panayiotopoulos, J.C. and Adam, N.R., 1981. "The dynamic lot-sizing problem for multiple items under limited capacity", AIIE Trans., vol. 13, no. 4, 294-303.

14. McCoy, R.J. and Gemmill, D.D., 1995. "A Wagner-Whitin based heuristic for solving dynamic, capacitated lot sizing problems", Proceedings of the 4th Annual Industrial Engineering Research Conference, in review, Nashville.

15. Lambrecht, M.R. and VanderVeken, H., 1979. "Heuristic procedures for the single operation multi-item loading problem", AIIE Transactions, 11, 319-326.

16. Thizy, J.M. and Van Wassenhove, L. N., 1985. "Lagrangean relaxation for the multi-item capacitated lot sizing problem, a heuristic approach", AIIE Transactions, 17, 308-313.

17. Blackburn, J.D. and Millen, R.A., 1980. "Heuristic lot-sizing performance in a rolling-schedule environment", Decision Science, vol. 11, 691-701.

Appendix I -- Modifications of the DS Heuristic

The basic DS heuristic (which was supplied by Dr. Luk Van Wassenhove, INSEAD) was modified to include the capability of solving CLS problems that include setup times. The reader is referred to the original article [10] for a detailed discussion of the basic DS heuristic.

While significant modifications to the code were required to extend the basic DS heuristic to include the capability of solving problems with setup times, most of the changes were more of an accounting nature, e.g., change the calculation of capacity consumption, modify the marginal benefit calculations to reflect the correct amount of capacity consumed by a particular lot size, etc.. However, the primary complexity was related to the issue of the forward feasibility check. The issue, for problems with setup times, is as follows: What amount of capacity should be reserved for setup time in future periods to avoid an infeasible overload of capacity? Several approaches were examined, but our approach uses lot-for-lot setup times, i.e., for each demand in forward planning periods, a corresponding setup time is associated. This is a conservative approach that initially assumes that no batching will occur (which lessens setup time requirements). Its major problem is that for tightly constrained problems, the modified DS heuristic will tend to make a number of non-economic batching decisions and maximize the use of production capacity in the earlier production periods. However, it tends to generate an initial feasible solution as its starting point and did not require significant amounts of lot size manipulation to achieve feasibility. Also, the problem of the modified DS heuristic moving more production than necessary into the earlier production periods tends to be corrected by the MG heuristic's forward improvement algorithm.

Appendix II -- DS Algorithm Computational Improvement

The computational efficiency of the DS improvement algorithm was increased by reducing needless computations. Specifically, after each lot size elimination, the unmodified DS improvement algorithm incorporates the new slack capacity information for the affected periods and recalculates the potential cost savings for every potential lot elimination. We avoid this complete recalculation effort by partitioning the lot elimination possibilities into two subsets: (1) those whose calculations are not valid after the last elimination was implemented; and (2) those whose expected cost savings are still valid. This partitioning is based on where the potential lot elimination is in the lot size production matrix (items * production periods) relative to the location of the lot just previously eliminated. If the movement of the eliminated lot (e.g., from period 7 to period 4) does not overlap the potential movement of the possible lot elimination (e.g., from period 12 to period 9, or from period 3 to period 1), then the previous cost calculation for the possible lot elimination is still valid -- thus no need to recalculate.

Appendix III -- Random Problem Generator Capacity Determination

Our random problem generator was modified to use the average capacity calculated by the original generator, Diaby et al. [7], for periods two and up and use this as the capacity basis for all the periods. The modified generator calculates the capacity level for each period as being normally distributed around the average, with the variance level at one-sixth the average value. Then, as a feasibility check for period one only, the generator adds the total production time of the items demanded in period one plus any required period one setup time and compares this value to the first period's randomly generated capacity level. If the required level is higher than the capacity level, then the first period capacity level is increased to 1.03 times the required level and the amount of the increase is reduced uniformly from the last three production periods (so as to not change the total capacity). Unfortunately, for tightly constrained problems, this does tend to produce a small upward bias to the first period's capacity level, but it reduces the number of infeasible problems generated and the amount of upward bias is significantly less than the amount of bias contained in the unmodified generator.

Appendix IV -- CLS Problem Feasibility/Infeasibility

CLS problems with setup time where feasibility is questionable should only include those with the following characteristics: (1) the cumulative sum of processing time and lot-for-lot setup time in periods one to the end of the production horizon exceeds the cumulative capacity for at least one of the periods -- otherwise the problem is clearly feasible; (2) the sum of processing time and setup time for the items demanded in period one is less than the available capacity in period one -- otherwise the problem is clearly infeasible; and (3) for periods one through the end of the production horizon, the cumulative total of the processing time per period (without backlogging and the inclusion of setup time) is less than the cumulative amount of capacity available per period -- otherwise the problem is clearly infeasible.

CHAPTER III. A WAGNER-WHITIN BASED HEURISTIC FOR SOLVING DYNAMIC, CAPACITATED, LOT SIZING PROBLEMS

A paper submitted to the *4th International IE Research Conference*

Robert J. McCoy and Douglas D. Gemmill
Industrial and Manufacturing Systems Engineering Department
Iowa State University, Ames, IA 50011

This paper discusses the development and testing of a fast, new heuristic for solving multiple item, multiple period, dynamic capacitated lot sizing problems.

Keywords: Lot Sizing, MRP, Heuristic.

INTRODUCTION

The Capacitated Lot Sizing (CLS) problem is a common and important industrial inventory control problem that involves determining medium-range production planning lot sizes for items that share a common capacity constraint, e.g., in a MRP production environment. Specifically, the problem consists of scheduling for multiple periods the production timing and quantity of multiple items so as to minimize the sum of both setup and linear holding costs while satisfying demands (without backlogs) and adhering to the capacity constraints associated with a single resource. The capacity constraints and product demands may be time-varying, but are assumed to be deterministic. Furthermore, item setup cost and setup time (if required) are incurred in each period that a particular item is produced. For a mathematical definition of the CLS problem without setup times and a detailed review of associated solution methodologies, the reader is referred to two reviews (Bahl et al., 1987; and Maes and Van Wassenhove, 1988).

Presented below is a discussion of a new heuristic for solving CLS problems. Hereafter, this heuristic will be referred to as the MG heuristic. Following this, we provide test results associated with applying the MG heuristic and a number of other heuristics to both a set of CLS problems available from the liturature and new, randomly generated, large-scale problems.

THE MG HEURISTIC

The new MG heuristic has three main sections: (1) Wagner-Whitin algorithm and feasibility attainment subroutine; (2) a modified Dixon and Silver (1981) heuristic; and (3) improvement algorithms. Note: The MG heuristic is capable of solving CLS problems with significant levels of setup time. However, due to space limitations, this capability will not be discussed.

The MG heuristic starts by ignoring the capacity restrictions and uses the basic Wagner-Whitin (WW) dynamic programming algorithm to optimally solve the resulting uncapacitated lot sizing problem. Then, after reinstalling the capacity restrictions, a feasibility check is performed. Typically, except for trivial problems, the WW solution is infeasible. If it is feasible, the heuristic ends with the retained WW solution.

The heuristic then utilizes a modified version of the Trigeiro (1989) "smoothing" subroutine from their TTM heuristic to seek feasibility. This modified code does not utilize dual prices of capacity, as does the original code. It merely seeks to obtain an initial feasible solution that satisfies capacity constraints using the WW generated lot sizes as a starting point. It does this by making, if necessary, multiple backward and forward passes to shift full or partial lot sizes out of periods that are overloaded with respect to capacity. However, the smoothing routine does not guarantee a feasible solution will be found (our tests indicate that a feasible solution is generally found). If the "smoothed" WW solution is not feasible, then feasibility is sought using the basic Dixon and Silver (DS) heuristic (with a perturbation factor of 0.0 -- see discussion below). Use of the DS heuristic guarantees a feasible solution will be found to a feasible problem.

The next step of the MG heuristic depends on the size of the CLS problem being solved. If the CLS problem is 200 items or larger in size, the MG heuristic proceeds directly to the improvement algorithms. However, if it is a small problem, the modified DS heuristic is used. Note: for larger CLS problems, the DS heuristic is only used if the WW and related "smoothing" procedure fails to obtain an initial feasible solution.

The basic DS heuristic (which was supplied by Dr. Luk Van Wassenhove, INSEAD) was modified to allow for the generation of multiple, feasible solutions to a particular CLS problem through the use of what we refer to as perturbation factors. (It was also modified to solve CLS problems that include setup times.) Because the DS heuristic quickly solves small CLS problems, this modification often provides the user with improved solutions at a small computational cost. The reader is referred to the original article by Dixon and Silver (1981) for a more detailed discussion of their basic heuristic.

The perturbation factors included in the modified DS code are simply cost multipliers applied to the holding cost portion of the lot sizing utility benefit calculations used in the DS heuristic. Each time the marginal benefit of batching a particular production lot is calculated, the perturbation cost factor often has the effect of modifying (in close or tie breaker situations) the choice of the next item selected for increasing its lot size, as well as influencing the total amount of batching. The use of multiple factors allows the user to iteratively generate multiple initial feasible solutions, apply improvement algorithms, and retain the lowest cost solution obtained. Perturbation factors of -3, -2, . . . , 2, 3 were used for the phase 1 testing discussed later in this paper. Note that the use of a perturbation factor of 0.0 means that the modified DS heuristic will obtain the same solution as the original DS heuristic.

Phase 2 testing, because the testing involved large problems, did not use perturbation factors. More detail on the use of perturbation factors in conjunction with the DS heuristic is provided in McCoy and Gemmill (1994).

The third (and last) main section of the MG heuristic involves the use of three improvement algorithms. These improvement algorithms are run in series and include a modified version of the improvement algorithm supplied with the basic DS heuristic and forward and backward adjustment improvement algorithms. These algorithms adjust the lot sizes contained in the best feasible solution achieved to that point with the objective of maintaining feasibility while lowering total costs (setup and holding). After running the improvement algorithms, the MG heuristic recalls the best solution achieved and terminates.

The first of the improvement algorithms used by the MG heuristic is a modified version of the DS improvement algorithm supplied by Dr. Van Wassenhove, which implements possibilities 1 and 4 of the improvement algorithm described in Dixon and Silver (1981). It was modified to account for the impact of setup time and improve its computational efficiency.

In addition to the basic DS improvement algorithm, the MG heuristic uses two other improvement algorithms. These algorithms also start with the best, feasible lot sizing solution generated to that point in time and look for improvement in the final solution cost by adjusting certain lot sizes (complete or partial) forward or backward in time. The first of these improvement algorithms, called the 'Forward Adjustment' algorithm, begins by checking for extra item inventory that is needlessly being carried and that could be shifted to a later production period to lower total planned costs. It is a period by period, multiple pass, review (starting in period one). It seeks to lower total costs by reducing holding costs -- possibly at the expense of additional setup cost, unlike the basic DS improvement algorithm.

The last of the three improvement algorithms is called the 'Backward Adjustment' algorithm. It deals with cost effectively eliminating some production setups by shifting production of one or more items back to an earlier production period. Unlike the basic improvement algorithm of Dixon and Silver, the Backward Adjustment algorithm does not restrict itself to simply eliminating a particular lot size and shifting it back in time to a period that already has a lot scheduled and that has adequate capacity. If adequate incremental capacity exists and it appears cost effective to do so, the algorithm will overload a particular period's capacity and then use the modified DS heuristic to regain feasibility by shifting production into an earlier production period(s). This opens up improvement possibilities that are not available to the basic DS improvement algorithm. A more detailed discussion of both the forward and backward improvement algorithms, as well as flowcharts, are provided in McCoy and Gemmill (1994).

PHASE 1 TESTING

The first phase of testing used a set of 15 problems from Eppen and Martin (1985), called the Eppen and Martin, or E&M, problems. These relatively small problems (without setup time) were used to establish a performance baseline for the tested CLS heuristics on a known set of readily available problems. These 15 problems were segregated into two categories, TVW* and DS*. TVW* includes the following problems: TVW1; TVW2; TVW3; TVW4; TVW50; TVW100; and TVW150. DS* includes the following problems: DSBASE; DS115L; DS110L; DS105L; DS100M; DS199T; DS198T; and DS200M. The size of these 15 problems varies from 8 to 200 items and from 8 to 13 periods, with the largest problem being 200 items and 10 periods.

The CLS heuristics tested in phase 1 include the following: LV (Lambrecht and VanderVeken, 1979), DS (Dixon and Silver, 1981), DPA (Dogramaci, et al., 1981), TTM

(Trigeiro, Thomas, and McClain, 1989), TVW (Thizy and Van Wassenhove, 1985), SLR (Diaby et al., 1992) with two sets of run control parameters, and the MG heuristic. Additionally, the integer programming branch and bound solutions of Eppen and Martin (E&M), 1985, are provided. Computer implementations of the LV, DS and DPA heuristics were supplied by Dr. Luk Van Wassenhove, INSEAD. The other heuristics were supplied by their respective authors.

The SLR heuristic requires explanation. The SLR heuristic is controlled by ten run control parameters (step size, reduction coefficient, improvement factor, transportation factor, etc.). Using the parameters listed in Diaby et al. (1992), we were not able to reproduce the published results. This is due to an error in the paper. The transportation factor for all problems except DS200M should be 0.9 rather than the published 0.8 (page 1333). Also, in order for the heuristic to obtain a feasible solution for DS200M, a transportation factor of 0.05 was recommended for this one problem only [Diaby, private communication, 1994].

Using these two transportation factors as indicated, we were able to duplicate their published results. However, since it is not possible to know a priori which parameter to use on a newly generated or previously unstudied problem, we twice applied the SLR heuristic to all 15 E&M problems -- using transportation factors of 0.9 and 0.05. Use of these two factors often resulted in different solutions and computation times. Thus, our SLR test results are grouped under two names, SLR1 and SLR2.

Table I summarizes the results of phase 1 testing. Results are segregated by solution quality (average percent deviation from the best available solution) and computation time (total CPU seconds on a DECstation 5000/200). Not surprisingly, the E&M solutions were the lowest (average) cost. However, the computation time of this integer programming approach is excessive. Eppen and Martin (1985) report requiring 38,125 seconds (10.6

TABLE I
EPPEN AND MARTIN BASELINE PROBLEMS

AVERAGE PERCENT DEVIATION FROM THE BEST SOLUTION AVAILABLE:

PROB.	NO. OF PROB.	LV	DS	DPA	MG	TTM	E&M	TVW	SLR1	SLR2
TVW*:	7	**5.02**	**4.09**	**4.45**	**1.23**	**0.53**	**0.22**	**2.14**	**5.11**	**4.22**
DS*:	8	**1.94**	**1.41**	**0.77**	**0.47**	**0.21**	**0.02**	N/A	**0.42**	**0.66**

TOTAL (DECSTATION 5000) CPU SECONDS REQUIRED TO SOLVE ALL 15 PROBLEMS:

PROB.	NO. OF PROB.	LV	DS	DPA	MG	TTM	E&M	TVW	SLR1	SLR2
TVW*:	7	0.7	0.8	6.8	7.8	15.3	N/A	6.5	10.3	50.9
DS*:	8	1.6	1.7	2.7	11.4	49.7	N/A	N/A	58.0	74.3
TOTAL	15	**2.3**	**2.5**	**9.5**	**19.2**	**65.0**	N/A (1)	N/A	**68.3**	**125.2**

(1) Eppen and Martin solution times on a VAX 750 totaled 38,125. seconds

hours) of VAX 750 CPU time to solve the 15 relatively small problems. This amount of time highlights the importance of using heuristic solution procedures.

The next best sets of solutions were provided by the TTM heuristic, using 100 iterations of their primal-dual solution methodology (Lagrangean dual costs are iteratively updated by subgradient optimization) as they recommended for problems without setup time (Trigeiro et al., 1989). Its solutions deviated from the best available solution by an average of only 0.53% and 0.21%, for TVW and DS problems, respectively. Furthermore, the total computation time for all 15 problems is a much more reasonable 65.0 seconds of DECstation 5000/200 CPU time.

The MG heuristic is one step below the TTM heuristic in terms of solution quality. On these small problems, using the "smoothed" Wagner-Whitin procedure as well as 7 iterations of the modified DS subroutines contained in the MG heuristic, the MG heuristic achieved solutions that deviated from the best available by 1.23% and 0.47%, respectively, on the TVW and DS problems. And, its total computation time was 19.2 seconds. In contrast, the LV and DS heuristics were the fastest tested -- either required less than 2.5 seconds of total CPU time. However, the 'price' of this speed is reduced solution quality -- their average percentage deviation was three to four times greater than the MG heuristic's deviation.

In comparison with the DS and LV heuristics, the DPA heuristic did relatively well on the DS group of problems and relatively poorly on the TVW problems (in both solution quality and computation time). The primary reason for this is the poor performance of the DPA heuristic on the relatively tightly constrained TVW1 and TVW2 problems (DPA works best on loosely constrained problems).

On average, the computationally complex TVW and SLR Lagrangean Relaxation heuristics did not perform as well as the TTM or MG heuristics, particularly on the TVW problems. The TTM heuristic's superior performance relative to the TVW heuristic is consistent with the results reported by Trigeiro et al., (1987). However, the SLR1 and SLR2 heuristic results are somewhat surprising. The heuristic performed very poorly on the TVW1, TVW2 and TVW3 problems. And, more significantly, neither SLR1 nor SLR2 achieved feasible solutions on all 15 problems. SLR1, with a transportation factor of 0.90, did not identify a feasible solution to DS200M; SLR2, with a transportation factor of 0.05, did not identify a feasible solution to DS100M and DS198T. Furthermore, as can be seen in Table I, the solutions and computation times achieved by the SLR heuristic were highly affected by changing just the one run control parameter. Consequently, the robustness of the SLR approach is questionable.

On the basis of the overall results provided in Table I, we included the DS, MG, and TTM heuristics into the phase 2 testing (which used larger, randomly generated problems). The LV heuristic was omitted because it was only marginally faster than the DS heuristic, yet tended to produce poorer solutions. The TVW and SLR heuristics were omitted because, on average, they did not perform as well or as reliably as the TTM heuristic, which produced very good lower bounds in addition to good solutions. Furthermore, testing of the TTM heuristic on larger problems was desired -- previously reported testing (Trigeiro, et. al., 1987 and 1989) was limited to problems of no more than 36 items. Finally, the DPA heuristic was omitted because its computational complexity is such that it is not well suited for large problems.

PROBLEM GENERATOR

The random problem generator used for Phase 2 testing was a modified version of the generator used by Diaby, et al. (1992) for their very large scale testing. The reader is referred to their article for details relating to the basic generator. With respect to generating problems without setup time, our significant modifications to the problem generator are as follows:

(1) Production time, rather than being constant at 1.0 per unit demand was allowed to vary normally around a mean value of 1.0. (Variable production times are more common in actual production settings.)

(2) We allow the user to approximately specify the average setup cost and carrying cost levels by varying a setup cost factor and a carrying cost factor, respectively. These two factors, which must be positive real numbers, are included in the problem generator run control file. These run control factors allow the user to influence the generated problem's level of time between orders (TBO), defined using the standard mathematical definition (see Maes and Van Wassenhove, 1988).

In addition to the setup cost and carrying cost factors, our problem generator retains the original generator's capability of modifying two other run control factors: demand variability and capacity constraint tightness. For our testing, the capacity constraint tightness factor was set to generate relatively tightly constrained problems (target utilization was 90%) and the demand variability factor was set to a high level, i.e. two times the variability used by Diaby, et al. (1992). This increased the 'lumpiness' of the item demands and contributed to problems that are more representative of 'real life' MRP component demand schedules.

PHASE 2 HEURISTIC TESTING

Our phase 2 testing was designed to determine the relative performance of the three best heuristics from phase 1 on larger, randomly generated, problems. The testing involved the random generation of 12 problems without setup time, each containing 500 items and 26 production periods. After omitting the last four production periods (to reduce end of horizon effects), their capacity utilization ranged from 88% to 95%, with an average value of 90.6%. The overall TBO of these problems ranged from 2.71 to 3.97, with an average value of 3.52.

Note that item TBO values, as would be expected in a typical production environment, often varied significantly around their overall problem TBO average.

As before in phase 1 testing, the TTM heuristic was run for 100 iterations. However, because the phase 2 problems are all large problems, the MG heuristic only used its modified DS heuristic subroutines (with a 0.0 perturbation factor) if the initial, "smoothed" WW solution was not feasible.

Phase 2 test results are provided in Table II. The results are segregated into two areas: (1) solution quality; and (2) computation time on a DECstation 5000/200 workstation. The heuristic abbreviations are as previously indicated. The 'LB gap' (lower bound gap) is defined as the percentage difference between the Lagrangean lower bound and the best solution achieved.

With respect to overall solution quality, the computationally complex TTM heuristic outperformed the MG and DS heuristics. Furthermore, its solution quality is quite good. For the 12 problems, its solutions averaged significantly less than one percent above the lower bound cost. However, the solutions achieved by the MG heuristic were also quite good -- its cost solutions averaged just 0.61% above the solutions achieved by the TTM heuristic and it generated average solution costs that were 1.37% lower than those of the DS heuristic.

There were significant differences among the three heuristics' performances with respect to computation time. The TTM heuristic, not surprisingly, was the slowest tested -- its average computation time was 181 seconds. In contrast, the MG heuristic averaged 5.5 seconds. But, we were surprised by the relatively slow computation time of the DS heuristic. Prior to our testing, we anticipated that the DS heuristic would be the fastest of the Phase 2 heuristics. Consequently, we analyzed the DS computation times in greater detail. Of the total DS time, about 63% was due to the improvement algorithm. Therefore, eliminating this

TABLE II
RANDOMLY GENERATED PROBLEMS
500 ITEMS AND 26 PERIODS (1)

COMPARISON OF THE MG, TTM AND DS HEURISTICS:

- **SOLUTION QUALITY:**

NO. OF	AVERAGE PERCENT DEVIATION			RANGE OF SOLUTION RATIOS	
PROB.	MG vs. TTM	MG vs. DS	L.B. GAP	MG/TTM	MG/DS
12	**0.61**	**-1.37**	**0.88**	**0.9962-1.0156**	**0.9816-0.9914**

- **COMPUTATION TIME:**

NO. OF	AVERAGE CPU SECONDS (2)			CPU TIME RATIOS	
PROB.	MG	TTM	DS	TTM/MG	DS/MG
12	**5.5**	**181.0**	**26.6**	**32.9**	**4.8**

(1) TBO Average Equals 3.52 and Average Capacity Utilization Equals 90.6%
(2) On a DECstation 5000/200 Workstation

algorithm would significantly reduce the DS computation time. However, use of the improvement algorithm reduced average DS solution costs by 1.1%. As a result, eliminating the algorithm would significantly increase the average percent deviation between the DS and MG heuristics (from 1.37% to 2.47%), and the average computation time of the basic DS heuristic would still be greater than that of the MG heuristic (9.6 seconds for DS versus 5.5 seconds for MG).

CONCLUSION

In this paper, a fast, new CLS solution mtheodology, called the MG heuristic, was presented and test results provided. These test results were favorable for the MG heuristic. On average, for all 27 problems tested, its solutions were 2.1% better than those of the well-known DS heuristic and just 0.7% above the solutions achieved by the computationally complex TTM heuristic. And, on the larger, randomly generated problems, the MG heuristic required about 21% and 3% of the DS and TTM heuristics' computation time, respectively. While additional testing is required, these results seem to indicate that the MG heuristic is fast enough and accurate enough to be used in realistically sized, "real life" production environments.

Acknowledgments:

The authors would like to thank Doctors Trigeiro, Diaby, Thizy and Van Wassenhove for their heuristic programs, problem generators and/or CLS problems.

REFERENCES

Bahl, H.C. and Ritzman, L.P. and Gupta, J.N.D. Determining lot sizes and resource requirements : a review, *Opns. Research,* Vol. **35**, 329-345, (1987).

Diaby, M. and Bahl, H.C. and Karwan, M.H. and Zionts, S., A Lagrangean relaxation approach for very-large scale capacitated lot-sizing. *Mgmt. Sci.* **38**, 1329-1340.(1992)

Dixon, P.S. and Silver, E.A., heuristic solution procedure for the multi-item, single-level, limited-capacity, lotsizing problem. *J. Opns. Mgmt.* **2**, 23-40, (1981).

Dogramaci, A., Panayiotopoulos, J.C. and Adam, N.R., The dynamic lot-sizing problem for multiple items under limited capacity, *AIIE Trans.*, Vol. **13**, No. 4, 294-303, (1981)

Eppen, G.D. and Martin, K.P., Solving multi-item capacitated lot-sizing problems using variable redefinition. *Opns. Research,* Vol. **35**, 832-848, (1987).

Lambrecht, M.R. and VanderVeken, H., Heuristic procedures for the single operation multi-item loading problem. *AIIE Trans.* **11**, 319-326, (1979).

Maes, J. and Van Wassenhove, L., Multi-item single-level capacitated dynamic lot-sizing heuristics: a review. *J. Opl. Res. Soc.*, Vol. **39**, 991-1004, (1988).

McCoy, R.J. and Gemmill, D.D., An Extended Dixon and Silver heuristic for solving multi-item, dynamic, capacitated lot sizing problems, *Proceedings of the 3rd Annual Industrial Engineering Research Conference*, 611-616, (1994).

Thizy, J.M. and Van Wassenhove, L. N., Lagrangean relaxation for the multi-item capacitated lot sizing problem, a heuristic approach. *AIIE Trans.*, **17**, 308-313, (1985).

Trigeiro, W.W., A Dual-cost heuristic for the capacitated lot sizing problem. *IIE Trans.*, March, 67- 72, (1987).

Trigeiro, W.W., Thomas, L.J. and McClain, J.O., Capacitated lot sizing with setup times. *Mgmt.Science*, Vol. **35**, No. 3, 353-366, (1989).

CHAPTER IV. MULTIPLE ECHELON, DYNAMIC, CAPACITATED LOT SIZING HEURISTIC USING COST ADJUSTMENTS AND SIMULATED ANNEALING

A paper submitted to *IIE Transactions*

Robert J. McCoy and Douglas D. Gemmill

Industrial and Manufacturing Systems Engineering Department

Iowa State University

Ames, IA 50011

In this paper, we investigate a new lot sizing heuristic for multiple echelon, assembly production systems with capacity constraints. More specifically, the new heuristic utilizes multiple iterations of a sequential top-down approach that combines single echelon, dynamic, capacitated approaches with a feedback mechanism to higher echelons. Additionally, the heuristic uses two cost modification procedures. First, it incorporates the KCC procedure developed by Blackburn and Millen. Second, holding cost adjustment factors, one for each echelon, are available for application to each item on a particular echelon. These factors were developed by the authors and assist the heuristic in finding a feasible solution to capacitated, problems. Then, the best (lowest solution cost) combination of factors is explored with a simulated annealing heuristic. In comparison to heuristics with<u>out</u> cost adjustments and feedback mechanisms, significant cost reductions were obtained for both capacitated and non-capacitated, randomly generated problems.

Production lot sizing in a multiple echelon production system is a common and important task. This general problem is commonly known as the cascading lot sizing problem. A material requirements planning (MRP) system is an example of the type of environment where these tasks must routinely be performed. As discussed in Billington et al. [2, 3], Kuik et al. [9], Maes and Van Wassenhove [10], and Blackburn and Millen [4], this type of production planning problem is very difficult due to the complex interdependencies that exist between lot size decisions made on each echelon. This is particularly true for assembly product structures with capacity constraints on one or more of the lower echelons.

The conventional MRP solution methodology handles the capacitated multiple echelon problem in a rather primitive manner. Using this methodology, capacity restrictions are initially dropped. Then, lot size decisions are generally determined using simple lot sizing (non-capacitated) heuristics, starting at the top echelon (the one associated with independent, end item demand) and proceeding sequentially down the product structure. Lot sizing decisions are made one echelon at a time, with the production lot sizing decisions at each echelon determining the dependent demand schedule of the echelon just below it. Then, capacity constraints are reimposed and Capacity Requirements Planning (CRP) procedures are followed. These CRP procedures check the MRP generated lot sizes and determine if the plan is feasible with respect to capacity restrictions. Unfortunately, it often is not, and an iterative (trial and error), labor intensive process of using MRP/CRP procedures is required before a feasible plan is generated.

In this paper, we consider the multiple echelon, capacitated lot sizing problem for assembly product structures. Furthermore, we allow various levels of capacity restrictions on each of the production echelons, lumpy demand, positive item setup costs and times, and variable item processing times. Additionally, we allow item capacity absorption ratios to vary from echelon to echelon. However, we do not consider overtime work or item lead-time considerations.

The multiple item, multiple echelon capacitated lot sizing problem for assembly product structures consists of scheduling production of all end items and their components on M total echelons over a horizon of T periods. Demands for the end items on echelon one are deterministically known and no shortages are allowed at any echelon. The objective is to minimize the sum of setup and inventory holding costs for all items on all echelons without creating item shortages or violating any capacity constraint associated with a particular period at each echelon. Consistent with the formulations of Billington et al. [3], but without provisions for lead times, the mathematical problem is as follows:

Objective Function: $\min \sum_{i=1}^{N} \sum_{t=1}^{T} (sc_i Y_{it} + hc_i I_{it})$

Subject To:

$$I_{i,t-1} + X_{it} - I_{it} - \sum_{j<i} U_{ij} X_{it} = d_{it} \qquad i = 1, \ldots, N; \qquad t = 1, \ldots, T;$$

$$\sum_{i \in K} (st_i Y_{it} + pt_i X_{it}) \leq C_{kt} \qquad K = k = 1, \ldots, M; \qquad t = 1, \ldots, T.$$

$$X_{it} - B\, Y_{it} \leq 0 \qquad i = 1, \ldots, N; \quad t = 1, \ldots, T;$$
$$X_{it},\ I_{it} \geq 0 \qquad i = 1, \ldots, N; \quad t = 1, \ldots, T;$$
$$Y_{it} \in \{0,1\} \qquad i = 1, \ldots, N; \quad t = 1, \ldots, T;$$
$$U_{ij} \in \{0,1\} \qquad i = 1, \ldots, N; \quad j = 1, \ldots, N;$$

where:

- N = Number of items in the production system
- sc_i = Set-up cost for end item i
- st_i = Set-up time for end item i
- pt_i = Processing time required to produce one unit of item i
- hc_i = Holding cost for end item i
- d_{it} = External demand for item i in period t
- K = Set of items on echelon number equal to K
- C_{kt} = Maximum available capacity at echelon k in period t
- Y_{it} = Set-up variable of item i in period t (i.e., 1 = setup; 0 = no setup)
- U_{ij} = Usage variable related to whether item i is required to make j
- X_{it} = Production quantity of item i in period t
- I_{it} = Amount of item i in inventory from period t to $t+1$
- B = Big positive number

Note that in the above formulation, we do not allow for component part commonality. That is, no part may have more than one parent (higher echelon) part. Specifically, the number of units from any echelon k required in the production of one unit at the immediate successor echelon (k-1) is assumed to be constant and equal to either zero or one. As noted in Kuik et al. [9], this causes no loss in generality since any assembly problem with usage factors that are multiples of one unit can be easily transformed into an equivalent assembly problem with usage factors no greater than one.

Capacity restrictions significantly complicate multiple echelon lot sizing. The amount of capacity restriction on each echelon relative to the other echelons is critical, solution feasibility is difficult to determine and the addition of setup time is significant because it consumes a portion of the limited capacity. However, most production situations are capacitated and significant amounts of setup time are often required. Thus, while optimum solutions are not obtainable for realistically sized problems, improved lot sizing tools for this environment are required.

A number of procedures for solving capacitated, multiple echelon problem have been proposed. For research prior to 1987, the reader is referred to Bahl et al. [1]. Since 1987, Kuik et al. [9] proposed linear programming (LP), simulated annealing (SA) and Tabu Search (TS) heuristics for solving lot sizing problems relating to assembly systems. In their study, they reduced the overall problem complexity by limiting the production structures to six or seven items spread over three echelons, only the middle echelon was capacitated, all three levels used the same time between order (TBO) levels (either 2, 3, or 4) to generate item setup costs, and no setup times were allowed. They concluded that the TS and SA heuristics outperformed the LP heuristics and that SA slightly outperformed TS. However, for even their fastest heuristic, the six and seven item problems averaged about 90 to 100 seconds of Sun 3 workstation CPU time. With respect to solution quality, the best method (SA) generated solutions that averaged 12 to 23 percent above the lower bound obtained by solving the LP relaxation of the problem. No comparison to results obtained with single level heuristics was provided.

In Billington et al. [2], the authors study capacitated multiple echelon serial systems and solve associated lot sizing problems using modified single echelon heuristics such as Dixon and Silver that are applied sequentially to each production echelon (top down). They modified the single level heuristics' feasibility routines to work in multiple echelon environments and used several of the Blackburn and Millen [4, 5] cost adjustment procedures (e.g., KBB and KCC). Besides limiting their study to serial product structures with no more than 12 end items and 5 echelons (60 items), they also used a simplifying assumption that the items have proportional processing times across echelons, i.e., if item A requires twice as much capacity as item B on echelon 1, then A must also require twice as much capacity as B on all other echelons. This rather severe restriction allows for the implementation of relatively simple multiple echelon feasibility checks. Nevertheless, their results indicate that the Blackburn and Millen cost adjustment procedures can provide a significant enhancement to lot sizing heuristic performance.

Heuristic For Capacitated Assembly Systems

The multiple echelon lot sizing heuristic developed as part of our research is hereafter referred to as the MELS heuristic. It uses multiple iterations of single echelon, capacitated, lot sizing heuristics within its overall solution methodology. The specifics of which heuristics were used will be provided later. As with conventional MRP approaches, the heuristic solution procedure starts with the top (end item) echelon of the product structure and sequentially solves the echelon lot sizing problems. For echelons below the top echelon, if infeasibilities result from higher echelon decisions, a subroutine is called that adjusts higher level lot sizes (at one or more echelons) forward in time, without creating shortages. The purpose of the upward, forward, adjustment is to decrease capacity requirements in the earlier production periods of the lower level(s). It is a myopic, greedy adjustment that seeks to move an adequate amount of production forward in time with the minimum amount of cost. At the completion of lot sizing at the lowest echelon of the assembly product structure, if upward adjustments of lot sizes were required, the heuristic does a final review of all created lot sizes. This review starts at the top echelon and proceeds down the product structure, one echelon at a time. It performs the following tasks: (1) checks for and attempts to correct capacity overloads, first through forward, then backward, production shifts; (2) eliminates non-economic setups by a forward production shift, if such shifts do not violate capacity or demand constraints; and (3) establishes the correct dependent item demand schedules for each echelon (below the top level). Then, following this review, if a feasible solution has been obtained, the heuristic calculates the total multiple echelon cost.

The heuristic also uses (or has the capability of using) two cost modification procedures. First, the KCC cost adjustment methodology of Blackburn and Millen [4, 5] was included in the heuristic. Although it was not designed for multiple end item, dynamic capacitated problems, we included this multiple echelon methodology because we believed it offered the potential for improved solutions, particularly for problems without tight capacity constraints.

The KCC method assumes constant item demand and does not take capacity restrictions into account. It attempts to consider the inter-relationships in lot size decisions across echelons. The logic behind their approach is related to the fact that since setups at a particular echelon generate demand for the echelon just below, lot sizes at the lower echelon are impacted. Thus, their method modifies both setup and holding costs used in upper echelon lot sizing decisions in order to at least partially reflect cost impacts on lower echelons. Consequently, their methodology reduces the myopic behavior of sequentially applied, single echelon, lot sizing heuristics.

The KCC approach uses K-factors, which are estimates of the number of orders of a parent part which will be combined at the children's level. More specifically, if the echelons are numbered sequentially from one, two, ..., starting at the top echelon, the K_{ij} (K-factor for part i on echelon j) is an estimate of the number of orders of the parent part of i (on echelon j-1) which will be combined into a single order at echelon j. Note that K_{ij} may not be less than 1, otherwise this would imply that shortages will be created. Mathematically, the KCC methodology is as follows:

$$K_{ij} = \max\left\{((\hat{S}_{ij}/S_{P_{ij}})(e_{P_{ij}}/\hat{e}_{ij}))^2,\ 1\right\}; \qquad e_{ij} = hc_{ij} - \sum_{m \in C_{ij}} (hc_m)$$

$$\hat{S}_{P_{ij}} = S_{P_{ij}} + \sum_{m \in C_{ij}} (\hat{S}_m / K_m); \qquad \hat{e}_{P_{ij}} = e_{P_{ij}} - \sum_{m \in C_{ij}} (\hat{e}_m / K_m)$$

where:

C_{ij} = children of item i on level j
P_{ij} = single immediate successor (parent) of item i on echelon j
S_{ij} = unmodified setup or ordering cost for item i on echelon j
hc_{ij} = full value holding cost for item i on echelon j
e_{ij} = unmodified echelon holding cost for item i on echelon j
$\overline{D}_{ij}$ = average item i demand on echelon j
$\hat{S}_{P_{ij}}$ = modified setup or ordering cost for item i at echelon j
$\hat{e}_{P_{ij}}$ = modified echelon holding cost for the parent item of item i on echelon j

The order of calculation of the modified cost $\hat{S}$ and $\hat{e}$ are j=M, M-1, ... 1. Therefore, the values of K_{ij} consider cost information from all its predecessor echelons.

When the KCC modification is used, the modified setup and holding costs that it calculates are temporarily used by the MELS heuristic to sequentially develop the single echelon lot sizes. Then, at the completion of the lot sizing tasks, the modified costs are replaced by the original costs and the total multiple echelon cost is calculated.

The second cost modification included in our multiple echelon heuristic is a technique of using holding cost adjustment factors. For a problem with M echelons there are M factors, one for each echelon. These adjustment factors (F_j) are applied to each item's holding cost as follows:

$$\hat{hc}_{ij} = hc_{ij} * (1.0 + F_j) \qquad i = 1, ..., N_j; \;\; j = 1, ..., M$$

where:

M = total number of echelons
N_j = number of items on echelon j
$\hat{hc}_{ij}$ = revised holding cost for item i on echelon j
hc_{ij} = initial holding cost without F_j adjustment for item i on echelon j
F_j = holding cost adjustment factor for echelon j

These adjustment factors are real numbers. For our test problems, they were in the range of ± 40.0, and usually ± 10.0. These factors were developed for use in the MELS heuristic to assist it in finding feasible, low cost solutions to capacitated problems. The rationale behind their use is to influence the single level capacitated heuristic(s), used within the multiple echelon methodology, to shift lot sizes to the left (with a negative F_j value, which encourages larger lot sizes) or to the right (with a positive F_j value, which encourages smaller lot sizes). Note that the use of F_j values, as well as the KCC modifications, only impacts production lot sizing decisions and feedback mechanisms. When total multiple echelon costs are calculated, the original, actual setup and holding costs are used.

Two examples of situations that benefit from the use of the holding cost adjustment factors are provided. The first example situation benefits from using a positive value for echelon 1; the second example benefits from a negative value. Both examples are for simple two echelon problems and references to time between orders (TBO) refer to the average calculated value for all items on an echelon, with TBO calculated using the standard (classical) definition, see Maes and Van Wassenhove [10], and unmodified setup and full value holding costs.

(1) In this example, the capacity constraint on the lower echelon is relatively tight, and the time between orders (TBO) on the top (first) echelon is relatively high (e.g., 5). On the top echelon, without applying a F_1 value, a significant amount of production is scheduled early in the production plan due to its high TBO. This then results in a capacity overload in

the early production periods of the second echelon. Consequently, the application of a positive F_1 value to the items of the top echelon will tend to reduce batching and shift the lower echelon dependent demand to the right. Thus, the opportunity of locating a feasible multiple echelon solution is increased.

(2) In this example, the KCC modification was not used and TBO is constant (e.g., 3 and 3) or increasing (e.g., 3 to 4). On the top echelon, by applying a negative F_1 value, the amount of batching will increase (i.e., lot sizes increase in size) and production will be shifted to the left. This will tend to increase actual costs for the top echelon, but overall costs over both echelons may decrease. More specifically, in this situation the application of a negative F_1 value will tend to have the following impacts on the top echelon: (a) actual holding costs will tend to increase; (b) setup costs will tend to decrease; and (c) the net (holding and setup) costs will tend to increase. However, associated with these impacts, the lower (second) echelon's net holding and setup cost tends to be reduced. Consequently, overall actual costs may decrease in spite of increased costs on echelon 1.

Our preliminary multiple echelon heuristic testing using the echelon holding cost adjustment factors (F_j) provided some insight on when their use was most beneficial and their typical order of magnitude and sign (i.e., positive or negative). We found that these aspects were related to whether or not we initially used the KCC modification procedure, whether the problem is capacitated and whether the TBO of the various echelons was increasing or decreasing. We will first address the issue of when one can expect a benefit from using F_j values. When binding lower echelon capacity constraints are present, then the use of positive F_j values often leads to a feasible solution being identified and/or lower overall solution costs. But, when the KCC modification is used and the problem is relatively loosely constrained (or non-capacitated), little benefit results from using non-zero F_j values. Also, with the use of KCC modifications, little benefit appears to result from the use of negative F_j values.

However, when the KCC modification is not used, our heuristic generally benefits from the use of non-zero F_j values, regardless of capacity constraint.

Regarding the general magnitude and sign of the actual F_j values that typically yield the lowest cost (feasible) solution, the tighter the capacity constraints on lower echelons, the greater the positive magnitude of the most beneficial value(s) of the F_j values on the higher echelon(s). Furthermore, when the KCC modification was not used, the magnitude of the F_j values are generally less positive than when it was used. In fact, for problems without binding capacity restrictions, when the KCC modification was not used, the 'best' F_j values (all) tend to be negative.

Preliminary testing of our multiple echelon heuristic experimented with using three single level heuristic solution methodologies. The three heuristics are: (1) TTM, see Trigeiro et al. [13]; (2) DS, see Dixon and Silver [7]; and (3) MG, see McCoy and Gemmill [11, 12]. These three were selected based on single level testing reported in the McCoy and Gemmill references listed above. Briefly, the TTM heuristic proved to be a reliable Lagrangean relaxation based heuristic suitable for problems with or without setup times and costs. Its main drawback is its requirement of rather large computation times for medium and large scale problems. The DS heuristic was included in preliminary testing because it is a well known, good heuristic for solving problems without setup times. Finally, the MG heuristic was included because it is a quick heuristic that is capable of handling lot sizing problems with or without setup time. On problems without setup time, it tended to outperform the DS heuristic with respect to solution quality and, on large scale problems, it was generally faster. Also, in comparison testing with the TTM heuristic, particularly on large scale problems, its solution quality was quite good and its computation times were significantly lower. For example, as reported in McCoy and Gemmill [11], for 216 single echelon, capacitated problems that varied in size from 250 to 4000 items and from 25 to 50 production periods, the

MG heuristic's average solution costs were just 0.86% higher than the TTM heuristic's costs and its computation time was 0.026 that of the TTM heuristic.

Based on our preliminary, multiple echelon testing, we choose to use both the TTM and MG heuristics as part of our MELS heuristic. That is, in its sequential solution of single echelon problems, our heuristic uses the TTM heuristic when the single echelon problem is smaller than 40 items, and it uses the MG heuristic for larger problems. Thus, the solution of a multiple echelon problem may involve the use of both the TTM and MG heuristics. The rationale behind using both was that the TTM heuristic tends to produce better quality solutions (particularly for very small problems), and for small problems the extra computational penalty is not great. The MG is then used on all other problems in order to take advantage of its computational speed and relatively good solution quality.

Our MELS heuristic allows the user to determine if the KCC cost modifications are incorporated. Thus, we were able to test the effectiveness of the procedure on randomly generated problems. When it is chosen for use, the procedure is called by the heuristic just once, before the start of the sequential echelon by echelon solution procedure and has an insignificant impact on computation times. Our preliminary test results indicated that using the KCC procedure yielded significant cost improvements on relatively loosely capacitated problems, and minor improvements on tightly constrained problems. However, more formal testing was desired.

For non-capacitated problems, when the KCC modification procedure is used, an F_j value of 0.0 for j=1, ... , M (the total number of levels) is satisfactory. However, except for this one case, the "best" choice of individual F_j values for a particular, previously unstudied, problem is not known. Consequently, for all problems that are capacitated or where the KCC cost modification was <u>not</u> used, the choice of F_j values is a combinatorial optimization problem. We choose to solve this problem by using a modified SA algorithm.

Prior to a description of the SA stochastic search process, a brief description of the MELS heuristic's method of determining an initial search starting point is as follows. The MELS heuristic first attempts to solve the lot sizing problem with the use of F_j values equal to 0.0 (for j=1,...,M). If a feasible solution was not achieved using these F_j values, all F_j values are incremented by 1.0 and the process repeats until either a feasible solution is found or the F_j level reaches 40.0. If at the 40.0 level a feasible solution has not yet been found, then the heuristic terminates. If a feasible solution was found, its associated F_j values are the starting point of the SA procedure

Modified Simulated Annealing Search Procedure

SA is a stochastic search technique that was discovered as a result of simulating the cooling of materials from higher to lower level energy states. It has received considerable attention since 1983, when Kirkpatrick et al. [8] published the seminal paper on the topic. Since then, it has been proven a powerful technique for solving combinatorial optimization problems. It is essentially a relatively simple technique that constructs a sequence of solutions (a walk) through the set of permissible solutions called the state space. Four basic elements are required: (1) a state space definition and an initial starting state within the space; (2) a transition mechanism that defines neighboring states (within the state space) and allows the procedure to consider moving from the current state to one of the randomly selected neighboring states; (3) an acceptance mechanism and associated control parameters to determine whether the potential move to the neighboring state should be accepted; and (4) a termination mechanism to end the search. Each of these four elements will now be discussed with respect the MELS heuristic:

(1) State space and initial starting state: The state space for the F_j values at each echelon is the collection of discrete points starting at 0.0 and extending in the negative and positive directions in increments of 0.1 (i.e., ... ,-0.3, -0.2, -0.1, 0.0, 0.1, 0.2, 0.3, ...). This state space was chosen based on preliminary testing. The initial starting state is determined as follows: starting with the F_j values associated with the first feasible solution obtained as previously described (e.g., {1.0, 1.0, 1.0, 1.0} for a 4 echelon problem), the heuristic starts with the lowest (bottom) echelon and decreases the F_j value by the *hc_incr* amount provided in (2) below and solves the resulting multiple echelon lot sizing problem. This repeats as long as cost improvements result or until the F_j value reaches 0.0 or less. Then, this process repeats for each higher level echelon and terminates with the F_j values associated with the best solution cost obtained. These F_j values are the starting point for the stochastic search process described below.

(2) Transition mechanism: To move from the current state to a neighboring state, the procedure picks one of the echelons at random and then randomly moves a holding cost increment, or *hc_incr*, in either the positive or negative direction. For example, if *hc_incr* is set to be 0.2, an acceptable move is from {1.0, 1.0, 1.0, 1.0} to {1.0, 1.0, 0.8, 1.0}).

The value of *hc_incr* is determined by the initial starting point where the heuristic first found a feasible solution (prior to SA). The rationale behind the determination of the *hc_incr* is to allow, for capacitated problems, approximately 5 to 10 possible points between the value of the initial feasible F_j value and 0.0. Specifically, Table 1 provides this information. Note: non-capacitated problems always found an initial feasible solution with all F_j values equal to 0.0 -- thus, for these problems, a *hc_incr* of 0.2 was always used.

Table 1: Determination of the Initial Holding Cost Increment Value

Initial Feasible Solution	*hc_incr*	Initial Feasible Solution	*hc_incr*
< 2.0	0.2	16.0 ≤ 24.0	3.2
2.0 ≤ 4.0	0.4	24.0 ≤ 32.0	4.8
4.0 ≤ 8.0	0.8	≥ 32.0	6.4
8.0 ≤ 16.0	1.6		

(3) Acceptance mechanism and associated control parameters: In our SA implementation, if the candidate set of F_j values results in an infeasible solution to the multiple echelon problem, the move is rejected. Conversely, all moves to neighboring candidate states are accepted if the actual multiple echelon solution cost associated with the candidate set of F_j values is less than the actual cost of the current set of F_j values. The acceptance of all moves that result in a cost improvement is consistent with local search techniques, e.g., hill climbing. However, to avoid becoming trapped in a local minimum, the SA procedure also accepts some candidate states even if their associated solution costs are higher than the current state -- this allows SA to explore other regions of the state space and thus is more likely to identify the global minimum, regardless of the starting state.

If Z_c is the value of the solution cost associated with the current state of F_j values and Z_n is the value of the cost associated with the potential neighboring state, then the candidate solution is accepted if $Z_n \leq Z_c$ or if

$$p < \exp\left\{\frac{1}{\text{temp}_l} * \frac{(Z_c - Z_n) * 100}{Z_c}\right\}, \text{ for } Z_n > Z_c$$

where p is the probability of accepting the neighboring state and is randomly selected from the U(0,1) distribution, and temp_l is a control parameter (discussed in more detail below).

If the candidate move is accepted, then Z_c is set equal to Z_n. If the candidate move is not accepted, then the Z_c remains the same. Regardless, the transition mechanism is called again and the process repeats.

Essentially, the value of $temp_l$ is determined by the user defined initial (non-negative) starting temperature (T_0) and a cooling schedule (CS). The determination of both the T_0 and CS is somewhat an art, but the principle idea is to systematically and stochastically explore the significant regions of the state space and gradually reduce the amount of upward hill climbing that is allowed so that one gradually settles into the most promising state space region and finishes the SA search at a low temperature that prohibits moves to higher cost states. Thus, depending upon the T_0 and CS used, the SA approach incorporates varying amounts of both global and local search.

For our MELS heuristic, we used a CS described as follows:

$$temp_l = T_0 * \beta^{l-1}, \qquad \text{for } l = 1, \ldots, MAX$$

where: $temp_l$ = specific temperature at each temperature level l
MAX = maximum number of temperature levels considered
β = cooling factor

Additionally, at each temperature level ($temp_l$, $l = 1,\ldots,MAX$), we consider up to L candidate moves, also sometimes referred to as a 'chain length' of L. However, if during the search, the heuristic accepts $0.75*L$ moves at a specific temperature level, it immediately proceeds to the next (lower) temperature level.

In our SA implementation, the values of T_0, β, MAX and L vary depending on the size of the problem and whether it is capacitated on at least one echelon. Problem size is defined based on the total number of items in the product structure, and is as follows: (a) small/medium -- less than 500 total items; and (b) large -- 500 or more items. These CS values are provided in Table 2.

Table 2: SA Cooling Schedules

Prob. Size	Capacitated	T_0	β	*MAX*	L
Small/Medium	Yes	3.0	.84	16	16
Large	No	3.0	.84	16	16
Large	Yes	1.0	.80	5	20

At any point in the SA procedure, if at a specific temperature level, no candidate solutions are accepted or if after 4 different temperature levels, a cost improvement of 1% or more was not obtained, the procedure immediately proceeds to the temperature level associated with *MAX*-3 (e.g., 16-3=13). Preliminary testing indicated that this reduces run times while tending to not have a major impact on solution quality. Additionally, regardless of how many temperature levels were examined by the SA procedure, at the *MAX*-3 level, the best combination of F_j values obtained so far are recalled and the *hc_incr* value previously used is reduced by 50%. This allows the SA procedure to "fine tune" the F_j values within the most promising local region.

Finally, because the SA solution technique often revisits states (particularly at lower temperatures), the MELS heuristic maintains a record of the F_j value combinations already visited along with their associated solution cost. Thus, the heuristic avoids recalculating the multiple echelon cost -- it simply recalls the cost from its records.

(4) Termination Mechanism: The SA procedure ends at the conclusion of the *L*th candidate move of the last (lowest) temperature level. At this point the best solution value is recalled.

Problem Generator

The problem generator used to test the MELS heuristic is a modified version of the generator used by Diaby et al. [6] for their large scale, single echelon testing. This generator produces problems of user specified size (number of items and production periods) with randomly generated item setup cost, setup time and holding cost, demands and various levels of capacity constraint tightness. It was twice modified. The first set of modifications are related to single echelon problem testing described in McCoy and Gemmill [11, 12]. These modifications: (a) allowed for variable item processing time and lumpy demands typical of MRP systems; (b) provided the user the capability to specify target (average) TBO values; (c) provided a stronger correlation between setup cost and time; and (d) provided more realistic capacity profiles (the original generator tended to lump a disproportionate amount of capacity into the first production periods).

The second set of modifications is related to multiple echelon problems. Our final problem generator allows the user to specify the total number of echelons desired, the number of items on the first echelon, the maximum number of predecessors that each successor (higher echelon) item may have and the probability of each predecessor. Thus, the generator is capable of producing problems that are: large or small, single or multiple echelon, tightly or loosely constrained , with serial or randomly generated assembly product structures, with or without item setup times and a target, average TBO value for each echelon. This last capability is important due to the critical importance that TBO structure has on multiple echelon lot sizing. To provide the user with an understanding of how the generator uses the target TBO values supplied by the user, the following is offered:

(a) Each item i on each echelon k of the multiple echelon problem receives a randomly generated item echelon holding cost, $e_{k,i} = 2.0 * RV$, where RV is randomly chosen from the U(0,1) distribution.

(b) The total holding cost, $hc_{k,i}$, for each particular item i on each echelon k is calculated, starting at the bottom echelon and summing up the individual echelon holding costs for each of its predecessors.

(c) Then the setup cost for each item i on echelon k, $sc_{k,i}$, is determined as follows:

$$TBO_{k,i} = Target_k + Z_{k,i} * (Target_k / 8.0)$$

$$sc_{k,i} = 0.5 * hc_{k,i} * \overline{D}_{k,i} * (TBO_{k,i})^2$$

where $\overline{D}_{k,i}$ = average item demand per period for item i on echelon k
$Z_{k,i}$ = normally distributed random variable for item i on echelon k
$Target_k$ = target TBO for each echelon k (user provided value)
$TBO_{k,i}$ = TBO level of item i on echelon k

The FORTRAN code associated with the problem generator described above is available from the authors.

Heuristic Test Plan

The two general objectives of testing the MELS heuristic were associated with testing the effectiveness of using the KCC cost modification <u>and</u> the holding cost adjustment factors on:

(1) large, <u>non</u>-capacitated problems with randomly generated, assembly, product structures and various TBO profiles (e.g., constant, increasing, decreasing, and random); and

(2) <u>capacitated</u> problems of various size, capacity profiles, and level of setup time.

The testing related to (1) allowed us to compare our results to previous results obtained by Blackburn and Millen, [4, 5], for smaller problems with a small set of less complex product structures and constant or less variable item demand. The capacitated testing associated with (2) is, to our knowledge, the most extensive of its kind.

All testing of the MELS heuristic used randomly generated problems that incorporated variable item production times and lumpy (MRP) demand. Additionally, the number of echelons considered varied from 3 to 5 and the number of periods in the production horizon was fixed at 18 for all problems generated and tested. All testing was conducted on a DecStation 5000 workstation.

As part of our testing, for each problem generated, we twice applied the MELS heuristic. The first application did not use the KCC modification; the second application did. For each application to a capacitated problem, we recorded the solution cost and computational time corresponding to three points in the solution process: (a) at the completion of the initial attempt to solve the problem with all F_j values equal to 0.0; (b) at the completion of finding an initial feasible solution; and (c) at the completion of the heuristic. Problems were discarded for which neither approaches (with and without KCC) found a feasible solution (For a discussion of the difficulties of determining if feasibility exists for a particular problem, see Trigeiro et al. [13], Kuik et al. [9] and Billington et al. [2].) For non-capacitated problems, points (a) and (b) are the same, i.e., a feasible solution was always obtained at point (a).

Details of our testing associated with the two objectives are provided in the next two sections.

Non-Capacitated Testing

Testing used four sets of randomly generated, large problems, with each set associated with a TBO profile. All problems were 5 echelon, assembly problems with 20 items on the first echelon. The TBO profiles are as follows:

Echelon	TBO PROFILES Constant	Increasing	Decreasing	Random
1	3.0	1.0	4.0	U(2,5)
2	3.0	2.0	3.5	U(2,5)
3	3.0	3.0	3.0	U(2,5)
4	3.0	4.0	2.5	U(2,5)
5	3.0	5.0	2.0	U(2,5)

where U(2,5) represents the uniform distribution between 2.0 and 5.0. For the 'random' TBO profile, the TBO for each item on each echelon was randomly chosen from the U(2,5) distribution. Thus, significant differences exist between the TBOs of items on a single echelon.

A total of 10 assembly problems was randomly generated for each of the four TBO profiles. Each of the 40 assembly problems allowed up to 3 predecessors for each successor (higher echelon) item, with the probability distribution of predecessors being: 1 -- 0.25; 2 -- 0.50; and 3 -- 0.25. Consequently, the expected value of the total number of items in each problem is 620 (or 31 items per each of the 20 end items). Finally, since setup times have no impact due to the lack of capacity constraints, they were not included in the problems generated.

Due to the absence of capacity constraints, the solution of these problems did not require the use of smoothing algorithms, upward adjustments, or improvement algorithms. Consequently, since both the MG and TTM procedures use the WW dynamic programming procedure, the solution procedure used by the MELS heuristic to solve these problems was equivalent to using WW (with modified holding and setup costs) for every item on each echelon.

Results of the non-capacitated testing are provided in Table 3.

Table 3: Non-Capacitated Testing -- Average Solution Costs

TBO Profile	Baseline Cost*	With F_j & SA (**)	With KCC Only(**)
Constant	1000	831 (16.9%; 802-880)	822 (17.8%; 796-874)
Increasing	1000	841 (15.9%; 825-904)	810 (19.0%; 801-874)
Decreasing	1000	900 (10.0%; 868-954)	896 (10.4%; 864-950)
Random	1000	850 (15.0%; 832-918)	842 (15.8%; 824-897)

* Baseline Cost: Without KCC modification and all F_j values equal to 0.0.

** Average percent reduction from the Baseline Cost; Range of solution costs

The results in Table 3 indicate that both the KCC modification and the use of holding cost adjustment factors (F_j) combined with SA provides significant cost reductions in comparison with using neither the KCC or F_j values. However, the computation time associated with using the KCC cost adjustment is relatively insignificant -- in comparison to the Baseline approach (without using KCC or 'F_j and SA'), which required about 3 CPU seconds, the increase in computation time was less than 1 second of CPU time. In contrast, the use of F_j and SA required about 190 CPU seconds. Consequently, the use of the KCC cost adjustment is recommended for non-capacitated problems. And, if the KCC cost modification is incorporated, there is no need for the use of F_j and SA. Our testing indicated that with the KCC cost modification, the additional use of F_j and SA requires a significant computation increase in order to sometimes achieve very minor cost improvements. Nevertheless, the test results summarized in Table 3 indicate that if the KCC modification is not used, the use of F_j values and SA does incorporate multiple echelon information into higher echelon lot sizing decisions and results in significant cost savings. This ability is critical for success in the commonly occurring, capacitated, multiple echelon situations where the KCC cost modification procedure, by itself, has difficulty.

Capacitated Testing

The capacitated testing used randomly generated, three to five echelon problems with random TBOs (U[2,5]) for each item on each echelon, 20 end items and the same number of potential predecessors (3), and probability distribution, as was used in the non-capacitated testing. Thus, the size of these problems had an expected value of 140 items, 300 items or 620 items, for the 3, 4 or 5 echelon problems, respectively. Additionally, this testing used problems of varying capacity profiles and level of setup time. The capacity profiles associated with the ten cases considered are provided in Table 4.

Table 4: Capacity Profiles

	Case									
	A	B	C	D	E	F	G	H	I	J
Echelon 1	T	U	U	T	L	M	M	M	S	S
Echelon 2	U	T	U	T	L	M	M	M	S	S
Echelon 3	U	U	T	T	L	M	M	M	S	S
Echelon 4	-	-	-	-	-	-	M	M	-	S
Echelon 5	-	-	-	-	-	-	-	M	-	-

where, T (tight) represents greater than 85% capacity utilization, M (moderate) represents 70 to 85% utilization, L (loose) represents 60 to 70 percent utilization, U represents unconstrained and S represents problems with a significant level of setup time and moderate capacity utilization. Consequently, problems for cases A to H did not include setup time, and those for cases I and J did.

For each of the ten cases, six problems were generated. And, for the 12 problems with setup time (cases I and J), the target setup time utilization was 15%, i.e., in the final (best) solution obtained, 15% of the total capacity is used to setup for production. The actual setup time utilization averaged 16.5% and 15.1% for cases I and J, respectively.

For cases A to C, we explored the impact of a tight capacity constraint on a single echelon of a three echelon problem. Cases D to H were used to investigate the impacts of tight (case D), loose (case E) or moderate (case F, G and H) capacity constraints on each of their echelons. Additionally, cases F, G and H also provide an indication of the solution quality and CPU impacts resulting from various problem sizes (3 to 5 echelons). Finally, cases I and J consider 3 and 4 echelon problems that contain significant levels of setup time.

Table 5 provides an overview of the quality of solution costs obtained for each of the ten, capacitated cases considered. In Table 5, the baseline cost is associated with the initial feasible solution obtained with<u>out</u> the KCC modification. This baseline cost was assigned a value of 1000. The last three columns of the table provide the average and range of solution values obtained for (1) the initial (feasible) solution obtained using the KCC modification; (2) the best solution obtained as a result of holding cost factor optimization (SA) when the KCC modification was not used; and (3) the best solution obtained after optimization when the KCC modification was used.

For cases A, B and C, where capacity restrictions were placed on only one echelon, a quick examination of the Table 5 information indicates that the use of the KCC modification, on average, resulted in slight solution cost improvements in both the initial feasible solution and the final (best) solution obtained. However, a more detailed analysis that included paired-t statistical tests indicates, at a 95% confidence level, the following: (1) If the problem is only constrained on the top echelon (case A), the use of the KCC modification provides a statistically significant benefit, but further attempts at cost reduction using SA provide little benefit; and (2) If the problem is contrained on the middle or lowest echelon (i.e., case B or C), then the KCC modification does not provide a statistically significant benefit, but the use of SA provides a significant reduction in total multiple echelon costs (from the initial feasible solution). Additionally, with respect to problem feasibility, for cases B and C, upward adjustment of lot sizes were required on all problems.

For Cases A to C, the total CPU time of the MELS heuristic averaged 100 seconds when the KCC modification was used. Its computation time was 8 seconds (8%) lower when KCC was not used. Of the total time, less than 1 second was required to obtain the KCC modified costs -- the remaining 7 (plus) seconds of the difference was related to different amounts of production smoothing and upward lot size adjustments required to obtain feasibility at each iteration of the MELS heuristic.

Table 5: Solution Cost Quality

	Initial Solution *		Final Best Solution Cost **	
Case	w/o KCC	with KCC	w/o KCC	with KCC
A	1000 ***	956 (928-986)	961 (934-983)	954 (927-975)
B	1000	978 (895-1007)	981 (955-999)	951 (864-988)
C	1000	987 (883-1046)	950 (785-993)	920 (854-1018)
D	1000	930 (820-1053)	877 (839-936)	879 (806-995)
E	1000	944 (869-997)	934 (899-976)	911 (827-964)
F	1000	981 (861-1053)	945 (865-995)	933 (843-994)
G	1000	920 (831-988)	908 (814-991)	870 (773-948)
H	1000	959 (873-1040)	904 (854-968)	895 (808-952)
I	1000	968 (862-1171)	916 (862-975)	910 (771-1043)
J	1000	967 (875-1096)	886 (848-922)	874 (820-926)
AVE.	1000	959 (870-1044)	927 (865-974)	910 (829-980)

* Initial feasible solution obtained (prior to optimization of F_j values).
** At the completion of the MELS heuristic (including SA).
*** Baseline cost is without the use of KCC and is scaled to equal 1000.
( ) Range of values

With respect to cases D to J (7 cases and 42 problems), where capacity restrictions are present on each echelon, the results indicate that with the KCC modification, approximately

4.7% better (lower) initial feasible solutions were obtained. However, after the F_j values are improved using SA, the difference shrinks to 1.4%. That is, the best (final) solutions obtained with KCC are 1.4% better, on average, than the best solutions obtained without KCC.

Cases F, G and H all involve moderate capacity restrictions on each of their echelons (3, 4, or 5, respectively). The results for these cases clearly indicate that the SA optimization of the F_j values yields greater benefit when the number of echelons is the greatest (cases G and H). Overall, the average cost reduction associated with using both the KCC modification and SA is 10.1% (versus the baseline solution).

For cases I and J (both with setup time), the results indicate that the use of SA also results in greater benefits when the number of echelons is greater. For case J (4 echelons), the amount of cost reduction associated with using both KCC and SA is about 12.6% lower than the baseline cost.

For cases D to J, the average computation times per problem, in CPU (DecStation 5000) seconds, are provided in Table 6.

Table 6: CPU Times

	Initial Feasible Solution *		Final (Total) Time **	
Case	w/o KCC	with KCC	w/o KCC	with KCC
D	18 (6-46)	34 (4-68)	403 (196-598)	392 (136-581)
E	6 (3-9)	8 (4-13)	239 (107-314)	277 (100-418)
F	13 (4-26)	21 (6-35)	261 (131-392)	281 (162-416)
G	13 (5-18)	25 (11-44)	570 (232-998)	574 (412-1000)
H	42 (16-86)	119 (54-215)	631 (300-954)	718 (293-1202)
I	14 (3-27)	19 (5-41)	296 (134-983)	310 (85-489)
J	18 (13-36)	42 (24-80)	777 (540-1193)	705 (290-1146)
AVE.	18 (7-35)	38 (15-71)	454 (244-798)	465 (211-750)

* Initial feasible solution obtained (prior to optimization of F_j values).
** At the completion of the MELS heuristic (including SA).
( ) Range of CPU computation times

The computation times for cases D to F, the cases without setup time but with capacity constraints on each of their three echelons, indicate that the tighter the capacity constraints (i.e., case D), the greater the CPU requirements. This is consistent with expectations.

Cases F (3 echelons), G (4 echelons) and H (5 echelons) all incorporate moderate capacity impacts and random item TBOs. Additionally, the three cases used the cooling schedules (CS) provided earlier (F and G used the same CS and H used a "quicker" CS). The above times for these cases show the obvious increase in times required as problem size increases. However, for even the five echelon case H (which averaged 655 items) with the use of the KCC modification, an average of less than 2 and 12 minutes of CPU time was required to obtain initial feasible solutions and the final (best) solutions, respectively.

CPU times for cases I and J , in contrast to similar sized cases without setup time (cases F and G -- 3 and 4 echelon problems, respectively), indicate that significant levels of setup time increase the amount of time required. This is not surprising given the additional amount of problem complexity imposed by the setup time.

The information in Table 7 summarizes the percentage cost reduction from the initial baseline solution without the KCC modification to the final (best) solutions obtained after completion of SA, with and without the KCC modification.

Table 7: Percent Cost Reduction From Initial Solution *

Case:	A	B	C	D	E	F	G	H	I	J	AVE
To Final w/o KCC	3.6	1.9	5.0	12.3	6.6	5.5	9.2	9.6	8.4	11.4	7.35
To Final with KCC	4.6	4.9	8.0	12.1	8.9	6.7	13.0	10.5	9.0	12.6	9.03

* Without KCC modification

When the KCC modification was not used, the inclusion of the SA procedure provided statistically significant cost reduction benefits on all ten cases. Furthermore, the results in Tables 5 and 7 generally indicate that the SA procedure, with or without the KCC modification, provides the greatest cost improvement on more tightly constrained problems with deeper product structures, e.g., cases D, H and J. However, a visual examination of the information in these tables indicates that when the MELS heuristic (including SA) is applied to capacitated problems, the use of the KCC modification does not always result in the lowest costs. The best example of this is case D, the case with tight capacity constraints on each of its three echelons. To further investigate this, a paired-t statistical test of the last two columns of Table 5 was conducted., using a 95% confidence level. These two columns are associated with the best final solutions obtained after SA, without or with the KCC modification. A paired-t test on all 60 problems included in Cases A to J indicated that the final best solutions with the KCC modification are statistically better than the solutions without the KCC modifications. A similar test on the 24 problems included in cases E, F, G and H also indicated the same conclusion. These four cases do not include setup time and have loose or moderate capacity restrictions on each of their echelons. However, it is important to note that for the 18 problems included in Cases D, I and J (problems which are tightly constrained or have both setup times and moderate capacity restrictions), the same type of test indicated that the use of the KCC modification produced no statistically significant difference in solution quality.

The above information combined with that of the non-capacitated testing (Table 3) indicates that the KCC modification provides significant benefit on problems that are constrained moderately, loosely, or not at all. However, after SA improvements to holding cost adjustment values (F_j), the KCC modification, on average, provides an <u>in</u>significant benefit on relatively tightly constrained problems, including those that are only moderately constrained, but also include significant setup times. Nevertheless, because the KCC modification typically requires such a small amount of computation time, its inclusion as an option in the MELS heuristic methodology is warranted. For example, on the most difficult problems (those with relatively tight capacity constraints, particularly those with setup times), the MELS heuristic could be easily modified to seek an initial feasible solution, both with and without the KCC modification. Then, the heuristic could proceed with the SA procedure using the combination of adjusted setup and holding costs that yielded the best initial feasible solution.

Conclusion

This paper described a practical approach for solving commonly occurring multiple echelon production lot sizing problems. Our approach of using lot sizing feedback information to higher echelons coupled with holding cost adjustment factors provides a practical means of significantly increasing the likelihood of finding feasible solutions to realistically sized, capacitated problems. Additionally, it was shown that significant cost improvements can be obtained through the use of a modified SA procedure to explore the combinatorial optimization of the holding cost adjustment factors. Finally, our research results indicate that the KCC cost modification provides significant benefit on unconstrained or loosely/moderately constrained problems and, on average, tends to provide insignificant cost benefits on more tightly constrained problems, particularly if the problems include significant levels of setup time.

References

[1] Bahl, H.C., Ritzman, L.P., and Gupta, J.N.D., "Determining lot sizes and resource requirements: a review", *Operations Research*, Vol. 35, 329-345 (1987).

[2] Billington, P. J., Blackburn, J, Maes, J, Millen, Van Wassenhove, L. N., "Mult-Item Lotsizing In Capacitated Multi_Stage Serial Systems", Vol. 26, No. 2, *IIE Transactions*, 12-18 (1994).

[3] Billington, P. J., "Mathematical approaches to capacity-constrained MRP systems: review, formulation and reduction", *Management Science*, 29, 1126-1141 (1983).

[4] Blackburn, J.D. and Millen, R.A., "Simultaneous Lotsizing and Capacity Planning In Multi-Stage Assembly Processes", *European Journal of Operations Research*, Vol. 16, 84-93 (1984).

[5] Blackburn, J.D. and Millen, R.A., (1982). Improved heuristics for multi-echelon requirements planning systems, *Mgmt. Science*, Vol. **28**, 44-56.

[6] Diaby, M., Bahl, H.C., Karwan, M.H. and Zionts, S., "A Lagrangean relaxation approach for very-large scale capacitated lot-sizing". *Management Science*, 38, 1329-1340 (1992).

[7] Dixon, P.S. and Silver, E.A., "A heuristic solution procedure for the mültí-item, singl-level, limited-capacity, lot-sizing problem", *Journal of Operations Management*, vol. 2, no. 1, August, 23-40 (1981).

[8] Kirpatrick, S., Gelatt, C.D., Vecchi, M.P., "Optimization by Simulated Annealing", *Science*, Vol. 220, 671-679 (1983).

[9] Kuik, R., Van Wassenhove, L.N., and Maes, J., "Linear Programming, Simulated Annealing and Tabu Search Heuristics for Lotsizing in Bottleneck Assembly Systems", *IIE Transactions*, Vol. 25, No. 1, pp 62-71 (1993).

[10] Maes, J. and Van Wassenhove, L., "Capacitated Dynamic Lotsizing Heuristics for Serial Systems", *International Journal of Production Research* , Vol. 29, No. 6, 1235-1249 (1991).

[11] McCoy, R.J. and Gemmill, D.D., "An Extended Dixon and Silver heuristic for solving multi-item, dynamic, capacitated lot sizing problems", *Proceedings of the 3rd Annual Industrial Engineering Research Conference*, 611-616, Atlanta (1994).

[12] McCoy, R.J. and Gemmill, D.D., "A Wagner-Whitin based heuristic for solving dynamic, capacitated lot sizing problems", *Proceedings of the 4th Annual Industrial Engineering Research Conference*, in review, Nashville (1995).

[13] Trigeiro, W.W., Thomas, L.J. and McClain, J.O., "Capacitated lot sizing with setup times", *Management Science*, vol. 35, no. 3, 353-366 (1989).

GENERAL CONCLUSIONS

First, an extension to a fast and simple heuristic for solving multi-item, multi-period, single-echelon, dynamic, capacitated lot-sizing problems was presented and applied to eight problems found in the literature. Each of these problems did not include setup time. It was shown that the Extended Dixon-Silver (EDS) Heuristic provided better solutions than the original Dixon and Silver heuristic on seven of the eight problems and, on the five problems with available optimal solutions, it generated solutions that hit the optimal on two of the problems and deviated from the optimal on the other three by at most 0.8 percent. Therefore, it is a reasonable alternative for users wishing to increase their lot-sizing solution accuracy at a minor computational expense.

Second, the MG heuristic was presented. This heuristic is a fast heuristic for solving single echelon CLS problems, with or without setup time. Also presented was large-scale testing that evaluated the performance of the MG heuristic and three other leading heuristics on realistic CLS problems. Testing used 216 randomly generated problems that included several groupings of problem sizes as well as varied levels of capacity utilization and average TBO. One-half the problems tested were problems with significant levels of setup time.

Test results were favorable for the MG heuristic. On the randomly generated problems with<u>out</u> setup times, overall results of the testing (for both high and low TBO problems) indicate that the MG heuristic yielded cost solutions 1.03% better, on average, than the next best fast heuristic (DS), and it did so at a reduced computational expense. Furthermore, on the same problems, the MG heuristic achieved solutions that averaged just 0.47% above the solutions achieved by the computationally complex heuristic tested (TTM), and on average it obtained the solutions at 0.027 the computation time.

On the problems with setup time, the MG heuristic also performed well. In comparison with TTM, it generated solutions that were on average (for both low and high TBO problems) 1.24% higher, at only 0.024 the average computation time.

Overall, on all 216 problems tested, the MG heuristic achieved feasible cost solutions just 0.86% higher than the TTM heuristic. Furthermore, for even the largest problems tested (4000 items and 25 periods), the computation time for the MG heuristic averaged (over 24 problems) about one minute of CPU time on a DECstation 5000/200 workstation. This indicates that the MG heuristic is fast enough and accurate enough for most "real world" CLS problems, with or without setup time.

Third, further discussion and test results associated with the MG heuristic and the random problem generator were presented. The test results were favorable for the MG heuristic. On average, for all 27 problems tested (15 from the liturature and 12 randomly generated), its solutions were 2.1% better than those of the well-known DS heuristic and just 0.7% above the solutions achieved by the computationally complex TTM heuristic. And, on the larger, randomly generated problems, the MG heuristic required about 21% and 3% of the DS and TTM heuristics' computation time, respectively. These results seem to indicate that the MG heuristic is fast enough and accurate enough to be used in realistically sized, "real life" production environments.

Fourth, a practical approach for solving commonly occurring multiple echelon production lot sizing problems was described and tested. It was shown that the newly developed, sequential, top-down approach of using lot sizing feedback information to higher echelons coupled with holding cost adjustment factors provides a practical means of significantly increasing the likelihood of finding feasible solutions to realistically sized, capacitated problems. Additionally, it was shown that significant cost improvements can be obtained through the use of a modified SA procedure to explore the combinatorial optimization of the holding cost adjustment factors. Finally, the research results indicate that the KCC cost modification provides significant benefit on unconstrained or loosely/moderately constrained problems and, on average, tends to provide insignificant cost benefits on more tightly constrained problems, particularly if the problems include significant levels of setup time.

Fifth, it is believed that the multiple echelon, random problem generator discussed in Chapter IV provides researchers with a practical means of generating realistic lot sizing problems. This generator permits the study of problems with various capacity constraint profiles, setup time levels, item demand schedules, product structures, etc..

While it is recognized that many unresolved CLSP research issues still remain, it is hoped that the research discussed in this dissertation contributed to reducing the gap between industry practice and academic research activity.

BIBLIOGRAPHY

Afentakis, P. (1987). A parallel heuristic algorithm for lot-sizing in multistage production systems, *IIE Transactions*, March, 34-41.

Bahl, H.C. and Ritzman, L.P. (1984). A cyclical scheduling heuristic for lot sizing with capacity constraints, *Int. Journal Prod. Research*, Vol. **22**, No. 5, 791-800.

Bahl, H.C. and Ritzman, L.P. and Gupta, J.N.D. (1987). Determining lot sizes and resource requirements: a review, *Opns. Research,* Vol. **35**, 329-345.

Biggs, J.R. (1979). Heuristic lot-sizing and sequencing rules in a multistage production inventory system, *Decision Sciences*, Vol. **10**, 96-115.

Billington, P. J., Blackburn, J, Maes, J, Millen, Van Wassenhove, L. N., Mult-Item Lotsizing In Capacitated Multi-Stage Serial Systems, Vol. 26, No. 2, *IIE Transactions*, 12-18 (1994).

Billington, P.J., McClain, J.O. and Thomas, L.J., (1983). Heuristics for multi-level lot sizing with a bottleneck, *Mgmt. Science*, **32**, 989-1006.

Blackburn, J.D. and Millen, R.A., (1980). Heuristic lot-sizing performance in a rolling-schedule environment, *Decision Science*, Vol. **11**, 691-701.

Blackburn, J.D. and Millen, R.A., (1982a). Improved heuristics for multi-echelon requirements planning systems, *Mgmt. Science*, Vol. **28**, 44-56.

Blackburn, J.D. and Millen, R.A., (1982b). The impact of a rolling schedule in a multi-level MRP systems, *Journal of Operations Mgmt.*, Vol. **2**, No. 2, 125-135.

Blackburn, J.D. and Millen, R.A., (1984). Simultaneous lot-sizing and capacity planning in multi-stage assembly systems, *European Journal of Operations Research*, Vol. **16**, No. 1, 84-93.

Blackburn, J.D. and Millen, R.A., (1985). An evaluation of heuristic performance in multi-stage lot-sizing systems, *Int'l Journal of Prod. Research*, Vol. **23**, No. 5, 857-866.

Collier, D.A. (1982). Research issues for multi-level lot sizing in MRP systems, *Journal of Operations Mgmt.*, Vol. **2**, No. 2, 113-123.

Diaby, M. and Bahl, H.C. and Karwan, M.H. and Zionts, S. (1992a). A Lagrangean relaxation approach for very-large scale capacitated lot-sizing. *Mgmt. Sci.* **38**, 1329-1340.

Diaby, M. and Bahl, H.C. and Karwan, M.H. and Zionts, S. (1992b) Capacitated lot-sizing and scheduling by Lagrangean relaxation, *European Journal of Operational Research*, **59**, 444-458.

Dixon, P.S. (1979). Multi-item lot-sizing with limited capacity. Unpublished Ph.D. dissertation, Dept. of Mgmt. Sci., *U. of Waterloo*, Waterloo, Ontario.

Dixon, P.S. and Silver, E.A. (1981). A heuristic solution procedure for the multi-item, single-level, limited-capacity, lot-sizing problem. *J. Opns. Mgmt.* **2**, 23-40.

Dogramaci, A., Panayiotopoulos, J.C. and Adam, N.R. (1981). The dynamic lot-sizing problem for multiple items under limited capacity, *AIIE Transactions*, Vol. **13**, No. 4, 294-303.

Eisenhut, P.S. (1975). A dynamic lot sizing algorithm with capacity constraints. *AIIE Trans.* 7, 170-176.

Eppen, G.D. and Martin, K.P. (1987). Solving multi-item capacitated lot-sizing problems using variable redefinition. *Opns. Research*, Vol. **35**, 832-848.

Goyal, S.K. and Gunasekaran, A. (1990). Multi-stage production-inventory systems, *European Journal of Operations Research*, Vol. **46**, 1-20.

Graves, S. (1981). Multi-stage lot sizing: an iterative procedure, multi-level production inventory control systems: theory and practice, in Schwarz, L. (Ed.), *TIMS Studies in Management Science*, Vol. **16**, North Holland, New York.

Gupta, Y.P. and Keung, Y. (1990). A review of multi-stage lot-sizing models, *Int'l Journal of Prod. Mgmt.*, Vol. **10**, No. 9, 57-73.

Gupta, Y.P., Keung, Y. and Gupta, M.C. (1992). Analysis of lot-sizing models for multi-stage systems, *Int'l. Journal of Prod. Research*, Vol. **30**, No. 4, 695-716.

Kuik, R., Van Wassenhove, L.N., and Maes, J., Linear Programming, Simulated Annealing and Tabu Search Heuristics for Lotsizing in Bottleneck Assembly Systems, *IIE Transactions*, Vol. 25, No. 1, pp 62-71 (1993).

Lambrecht, M.R. and VanderVeken, H. (1979). Heuristic procedures for the single operation multi-item loading problem. *AIIE Trans.* **11**, 319-326.

Maes, J. and Van Wassenhove, L. (1986). A simple heuristic for the multi-item single level capacitated lot sizing problem. *Operations Research Letters*, **4**, 265-274.

Maes, J. and Van Wassenhove, L. (1988). Multi-item single-level capacitated dynamic lot-sizing heuristics: a general review. *J. Opl. Res. Soc.*, Vol. **39**, 991-1004.

Maes, J. and Van Wassenhove, L. (1991). Capacitated dynamic lot sizing heuristics for serial systems, *Int'l Journal of Prod. Research*, Vol. **29**, No. 6, 1235-1249.

McCoy, R.J. and Gemmill, D.D. (1994a). An Extended Dixon and Silver heuristic for solving multi-item, dynamic, capacitated lot sizing problems, *Proceedings of the 3rd Annual Industrial Engineering Research Conference*.

McCoy, R.J. and Gemmill, D.D. (1995a). A Wagner-Whitin based heuristic for solving, dynamic, capacitated lot sizing problems, *Proceedings of the 4th Annual Industrial Engineering Research Conference*, pending publication.

McCoy, R.J. and Gemmill, D.D. (1995b). A fast heuristic for large scale, capacitated lot sizing problems, with or without setup times, *Comp. and Ind. Eng..*, in review.

McCoy, R.J. and Gemmill, D.D. (1995c). Multiple echelon, dynamic capacitated lot sizing huristic using cost adjustments and simulated annealing, *IIE Transactions*, in review.

McLaren, B.J. (1976). A study of multiple level lot-sizing techniques for Material Requirements Planning systems, unpublished Ph.D. dissertation, Purdue University.

McLaren, B.J. and Whybark, D.C. (1976). Multi-level lot-sizing procedures in a MRP environment, Discussion paper No. 64, School of Business, Indiana University.

Peng, K. (1985). Lot-sizing heuristics for multi-echelon Assembly systems, *Eng. Cost and Prod. Econ.*, Vol. **9**, 51-57.

Ramsay, T.E. and Rardin, R.R. (1983). Heuristics for multistage production planning problems, *Journal Operational Research Society*, Vol. **34**, 61-70.

Ritzman, L.P. and King, B.E. (1993). The relative significance of forecast errors in multistage manufacturing, *Journal of Operations Mgmt.*, **11**, 51-65.

Roundy, R.O. (1993). Efficient, effective lot sizing for multistage production systems, *Operations Research*, Vol. **41**, No. 2, 371-385.

Steinberg, E. and Napier, H.A. (1980). Optimal multi-level lot sizing for requirements planning systems, *Mgmt. Science*, Vol. **26**, No. 12, 1258-1271.

Thizy, J.M. and Van Wassenhove, L. N. (1985). Lagrangean relaxation for the multi-item capacitated lot sizing problem, a heuristic approach. *AIIE Trans.*, **17**, 308-313.

Trigeiro, W.W. (1987). A Dual-cost heuristic for the capacitated lot sizing problem. *IIE Trans.*, March, 67-72.

Trigeiro, W.W., Thomas, L.J. and McClain, J.O. (1989). Capacitated lot sizing with setup times, *Management Science*, Vol. **35**, No. 3, 353-366.

Wagner, H. and Whitin, T.M. (1958). A dynamic version of the economic lot size model. *Mgmt. Science*, Vol. **5**, No. 2, 89-96.