A Genetic Algorithm for the Capacitated Lot Sizing Problem with Setup Times

CHEN, Jiayi

A Thesis Submitted in Partial Fulfillment
of the Requirements for the Degree of
Master of Philosophy
in
Systems Engineering & Engineering Management

The Chinese University of Hong Kong
September 2009


香港中文大學
圖書館系統
15 FEB 2011
THE CHINESE UNIVERSITY OF HONG KONG
LIBRARY SYSTEM

Thesis/Assessment Committee

Professor Janny Leung (Chair)
Professor C.H. Cheng (Thesis Supervisor)
Professor Wai Lam (Committee Member)
Professor Jaideep Motwani (External Examiner)

Abstract of thesis entitled:

A genetic algorithm for the capacitated lot sizing problem with setup times

Submitted by CHEN, Jiayi

for the degree of Master of Philosophy in

Systems Engineering & Engineering Management

at The Chinese University of Hong Kong in September 2009

This research work describes a production planning problem that determines the production timings and quantities for multiple products over a finite number of time periods without violating capacity constraints. We consider this problem, which is often referred to as the capacitated lot sizing (CLS) problem, in the case with the significance of setup times. An MIP model is developed for a case without backordering, incorporating setup times and allowing for different types of production capacities, such as regular time, overtime and subcontracting. The formulation of our model is a variation of the traditional fixed-charge transportation problem. We develop a heuristic approach based on Genetic Algorithm (GA) to solve this CLS problem. A variety of test problems is developed for the 3-product, 2-capacity, 12-period-planning-horizon scenario. All test problems consider three

parameters: seasonality of demand, the tightness of capacity, and the setup cost and setup time level. The computational results show that our heuristic algorithm gives reliable results to large problems when comparing to lower bounds generated. This thesis concludes with an analysis of the limitations of our work and a discussion of further research directions.

摘要

本文討論以有限的生產資源滿足在一定計劃期間內多種產品需求的生產計劃問題，旨在幫助決策者制定生產計劃，確定生產啓動時間以及生產批量。該問題通常被稱爲有限資源下的生產批量問題，而在本文中，我們着重于帶有啓動時間的資源限制型生產批量問題的探討。我們建立了一個混合整數規劃模型，該模型不允許延遲交貨，考慮了生產的啓動時間，並且包含多種生產資源與方式，例如正常生產時間，加班生產和外包。我們的模型是傳統的固定成本運輸問題模型的變異。本論文中提出了一個基於遺傳算法的啓發式算法用來解決之前所提到的問題。我們設計了一個包含三种產品、兩种生產資源、十二個生產時期的問題，並用大量的實驗加以驗證算法的有效性。所有的實驗都引用了三個參數，即產品需求的季節性、生產資源的稀缺程度，以及生產啓動成本與啓動時間的水平。實驗計算結果顯示，通過與生成的下限作比較，可知我們的啓發式算法能可靠地對於大規模問題提供優良的運算結果。在本文的結論中，我們分析了所建立的模型和算法的局限性與不足，並對今後進一步的研究方向以及對本文的拓展提出了一定的建議。

Acknowledgement

I would like to take this opportunity to express my gratitude to my supervisor, Prof. C. H. Cheng. He brought me into the research area in the Engineering field and inspired me to pursue my initial interests in quantitive studies and thus made me enjoy the research mode of life. His generous guidance and patience helped me overcome whatever difficulty I faced during my MPhil studies. Moreover, his confidence in me as well as his numerous encouragement was really valuable in my research life. I have learnt quite a lot from his experience he shared with me, which I believe must affect me for a long time in my future career. In short, I regard him not only as an advisor to my research thesis but a great mentor in my life.

I am also grateful for the time and valuable suggestion that Prof. Janny Leung has given for my research work. She let me realize the significance of having an independent and a rigorous attitude towards research, and understand I must be responsible for my study as well as my life, so that enough effort is a must for achieving a goal.

I thank Prof. Duan Li, Prof. Wai Lam, and Prof. Xiaoqiang Cai in my department for teaching me basic knowledge related to my research during the course study. Their kind reply and support made me strengthen the academic skills and improve my work. I also thank Prof. Lawrence Leung from Faculty of Business Administration in CUHK for giving me a memorable course on Research

Methodology, which taught me how to be a real professional researcher but not just a mechanism in solving problems.

I am particularly thankful to my external examiner, Prof. Jaideep Motwani. He responded to the draft of my thesis quickly and his comments were extremely perceptive, helpful, and appropriate, which helped me a lot in revising and improving my thesis.

And I wish to give my thanks to my fellow colleagues and friends, Mr. Alan Au, Ms. Garnet Xiong, Ms. Joyce Tong, Ms. Sharon Wong, Mr. Chris Chan, Ms. Peiqiu Guan, Ms. Miranda Cheng, Ms. Cecia Chan, Mr. Beyond Li, Mr. Jacky Chu, and Mr. Forest Kwan. They left me a colourful postgraduate study life in Hong Kong and encouraged me to complete my thesis.

Special thanks should be given to Mr. Yizhong Lin, Mr. Zhening Li, Mr. Bo Chen, Mr. Jun Jiang and Mr. Chenchen Zhou, and especially to my dearest friend Mr. Yiyin Zhou, who provided me valuable suggestion and guides on mathematics and programming studies, and always supported me without hesitation, sincerely and generously.

Finally, I am deeply indebted to my parents for their unconditional love and confidence these years. And may this work be dedicated to my grandma who passed away the day I finalized the draft version of my thesis.

Without these people, I would not be able to survive this challenging but tough or even desperate MPhil study period.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This research focuses on a single-level capacitated lot-sizing problem with setup time consideration. The model is formulated as a variation of the fixed-charge transportation problem. This modeling approach was designed by Gilbert and Madan (1991) to deal with a case without setup time consideration. We extend their work to handle setup time situation. A genetic algorithm based heuristic is developed. Extensive computational tests are presented.

1.1 Introduction to the Capacitated Lot Sizing (CLS) problem

The Capacitated Lot Sizing (CLS) Problem is a production planning problem that determines the production timings and the quantities of multiple products to satisfy

demands over a finite number of periods without violating resource constraints. It aims to minimize the sum of production, setup and inventory costs. This problem is commonly encountered in repetitive manufacturing settings involving processes such as assembly and stamping. This multi-product CLS problem is shown to be NP-hard by Madan (1988). Most solution algorithms are heuristic based. In this work, we consider the CLS problem with the significance of setup times.

The modeling approach we discuss here differs from the traditional formulation in the aspect that it considers multiple production resource capacities. Many traditional models only consider a single source of capacity, i.e., regular time. Therefore, some temporarily adjusting means, like overtime and subcontracting, could not be taken into consideration explicitly. Moreover, setup times, which cannot be neglected for some manufacturing situations, are taken into consideration in our model. Many research studies just assume that they are insignificant.

1.2 Our contributions

The original model developed by Gilbert and Madan (1991) for CLS problems assumed that setup times were insignificant. However, in some industries and manufacturing situations, setup times cannot be reduced to zero. Due to the significance of the setup time, the assumption of zero setup times is not reasonable. Therefore, models without setup time consideration may have their limitations when they are applied to industry.

From a theoretical perspective, the inclusion of setup times in CLS problems could not be simply regarded as an extension of the problems without setup times. This is true even for the original problems without setup time consideration that are not tightly constrained in capacity. This assertion was supported by Trigerio et al. (1989) using an illustrative example. In their example, they showed that a problem may be unsolvable when setup times were added to the constraints. The importance of setup times would be discussed more in detail in the next chapter.

The application of the fixed-charge transportation problem to a case of CLS problems was given by Gilbert and Madan (1991). We follow their research direction and formulate the problem as a variation of the classical transportation problem, and extend their problem to consider setup times. The modeling structure they proposed has some advantages over the traditional models. For instance, CLS decision variables can be directly obtained from the model solutions, while many traditional models require further calculation to find the values of these variables. These advantages would be further discussed in Chapter 3.

Our computational results show that our heuristic algorithm produces solutions that are within the percentage difference of 4.34% on average, when compared with the lower bounds. Moreover, the result of our heuristic algorithm also outperforms the one developed by So (1997) by 0.29% on average, when setup times are forced to zero.

1.3 Organization of the thesis

This thesis comprises 6 chapters. This chapter is a brief introduction to our research topic. A literature review in the problem and its solution algorithms is addressed in Chapter 2. In Chapter 3, we propose our model, assumptions and notations. A comparison between our model and the traditional one is also provided. Chapter 4 describes the whole heuristic algorithm procedures. The design of experimental tests and computational results are shown in Chapter 5. The conclusion is drawn in the last chapter.

Chapter 2

Literature Review

We mainly review the literature on Capacitated Lot Sizing (CLS) problems in this chapter. A brief summary of the development of Genetic Algorithm (GA) is also presented.

2.1 Research in CLS problem

Production planning is an activity that considers the allocation of available production resources to best satisfy the market demands and production requirements over a certain planning time horizon. It could be classified into three parts from the point of managerial decision making, long-term, medium-term and short-term planning. The Lot Sizing problem is a typical type of medium-term production planning problem, which considers setups between production lots. When setup

times and/or costs are significant, this problem becomes interesting and encapsulates one of the key issues in production planning (Bahl et al., 1987).

Bahl et al. (1987) classified lot-sizing problems into four categories, according to two particularly important environment characteristics: the type of demand and presence or absence of resource constraints. Figure 1 below shows the four categories: single-level lot sizing without resource constraints (SLUR), single-level lot sizing with resource constraints (SLCR), multi-level lot sizing without resource constraints (MLUR), multi-level lot sizing with resource constraints (MLCR).

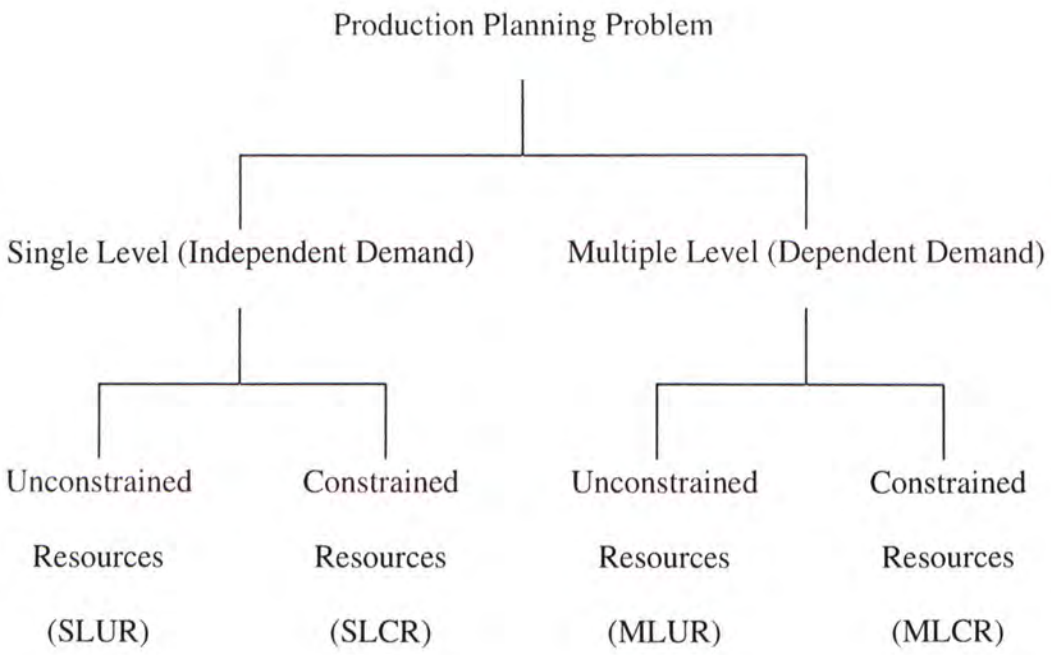

Figure 1: A classification of production planning (Bahl et al., 1987)

Karimi et al. (2003) addressed various characteristics that may affect the classification, modeling and complexity of lot-sizing problems. The planning horizon may be finite or infinite, which may be usually associated with a dynamic demand or a stationary demand, respectively. In addition, either dynamic or static demand could be classified into deterministic or probabilistic type, according to whether the value of demand is known in advance or not. If a production system deals with only simple final products or those materials without intermediate subassemblies, i.e., the demands of products are independent, we call it a single-level system. The multi-level problems refer to those cases that a parent-component relationship among items demand exists. Whether the resource capacities are with restriction or not will directly affect problem complexity.

Another important characteristic is setup structure. When it is independent of the sequence or previous periods, it is termed a simple structure. Otherwise it is called a complex one. Both structures may assume whether the setup cost and/or setup time are taken into consideration, and thus engendering different sub-problems. Inventory shortage is another significant factor, which may decide whether backordering is allowed or not. Also, the number of products and the features of products (i.e., perishable products) will affect the complexity of the whole problem as well.

Considering the scope of our research, the literature review in this section focuses on the SLCR problem with a finite-planning horizon and deterministic demand for multi-products. Literature on both cases with or without setup time, and both cases with or without backordering are also reviewed.

2.1.1 **Reviews in CLS problems**

There are several extensive reviews of lot-sizing research. Bahl et al. (1987) classified production planning problems into four major categories, outlining the research issues and reviewing selected contributions in all four categories. Their review covered the work by both practitioners and researchers. They assessed the strengths and weaknesses of these contributions, and suggested promising research avenues. Further, they asserted that capacity limitations, as well as other realities of plant environments, such as scrap, demand uncertainties or inaccurate data, were the big challenges of lot-sizing research.

Drexl and Kimms (1997) reviewed both the single-level and multi-level problems, focusing specifically on the dynamic lot-sizing problem and continuous time models. Mixed-integer programming models were given for both of them. They pointed out the importance of the consideration of positive setup times and backlogging cases in lot-sizing problems, and the linkage with other planning activities such as distribution planning and cutting and packing.

Karimi et al. (2003) focused their attention on the single-level lot-sizing problem. They addressed various characteristics that may affect the classification, modeling and complexity of lot-sizing problems, and summarized different algorithms and mathematical-based heuristics for both capacitated and uncapacitated models.

Brahimi et al. (2004) presented four different mathematical-programming formulations of the classical problems for both capacitated and uncapacitated versions. Some remarks about the CLS problem were worth mentioning according to their conclusion. For example, most of the extensions of the uncapacitated case, such

as backlogging, perishable inventory and time windows, had not been considered in the capacitated one. To follow up on their work, Brahimi et al. (2006) studied a new family of the CLS problems.

Jans and Degraeve (2008) gave an overview of recent developments in modeling single-level dynamic lot-sizing problems. But no solution approaches were discussed in their review paper. The single-item uncapacitated lot-sizing problem and the capacitated multi-item lot-sizing problem were discussed as two basic lot-sizing problems. Research works were summarized in operational and tactical/strategic versions. Five aspects: set-ups, characteristics of the production process, inventory, demand, and rolling horizon, were discussed for CLS problems. In the real industrial environment, models more global in scope coordinated planning, or models with some degree of uncertainty such as stochastic inventory and demand, are of huge interest.

2.1.2 Approaches and methods to solve the traditional CLS problems

Complexity theory as well as computational experiments indicates that most CLS problems are hard to solve. Because of the difficulties, various solution techniques have been proposed. Most researchers in the lot-sizing area use linear or non-linear mixed-integer programming based on dynamic-programming-based algorithms to

solve their formulated problems. Some reviews of these algorithms are presented here, and some representative studies on specific approaches are also outlined.

Maes and Wassenhove (1988) analysed and compared different heuristic approaches to solve the multi-item single-level CLS problem in their review paper. Both special-purpose methods and mathematical-programming-based heuristics were discussed. They also presented an extensive computational review to find relationships between the performance of the heuristic and the computational burden involved in finding the solution.

Staggemeier and Clark (2001) presented different CLS models covering several important aspects, such as capacity constraints, backlogs, setup costs and times, multiple machines and size of planning periods. They also listed major solution methods like mathematical programming, tabu search, simulated annealing and evolutionary algorithms.

Karimi et al. (2003) reviewed different algorithms and mathematical-based heuristics for both capacitated and uncapacitated cases. They concluded with some suggestions for further research, and highlighted the lack of fast and efficient heuristics for NP-hard CLS problems such as those with backlogging or setup times and setup carry-over. Also, some relatively new solution approaches such as tabu search, simulated annealing, and other meta-heuristics were suggested to solve these complex CLS problems.

Jans and Degraeve (2007) gave an overview on meta-heuristics for dynamic lot-sizing problems, such as tabu search, genetic algorithms and simulated annealing. The key components of these approaches, such as representation, evaluation, neighborhood definition and genetic operators, were discussed. Other solution

algorithms, like dynamic programming, cutting planes, Dantzig-Wolfe decomposition, Lagrange relaxation and dedicated heuristics, were also looked into for comparison. They also discussed general guidelines for computational experiments.

Quadt and Kuhn (2008) presented a literature review on problems that incorporate one of the following extensions to the CLS: backorders, setup carry-over, sequencing, and parallel machines. Formulations for each of the extensions were illustrated. The inclusion of setup times, multi-level product structures and overtime were also mentioned in this study.

In what follows, we discuss some representative papers on the approaches mentioned above. Although the most general model formulations still defy optimization when applied to realistically sized problems, many recent contributions are both innovative and promising.

Manne (1958) put forward an algorithm based on set partitioning. The formulation could be solved with linear programming by relaxing a 0-1 set partitioning problem to a linear programming problem and rounding off the resulting solutions. Dzielinski and Gomory (1965) used Dantzig-Wolfe (1961) decomposition principle to improve Manne's work. Dzielinski et al. (1963) also presented a simulation test for the lot-sizing programming. Zangwill (1966) used the concave cost network approach to analyse such problems. Eisenhut (1975) used a dynamic algorithm to consider a CLS problem with setup cost and holding cost. Although their algorithm could not always guarantee a feasible solution, their contribution was still significant since uncertain and fluctuating demand was allowed in their model. Lambrecht and Vanderveken

(1979) followed up Eisenhut's work and improved the marginal cost determination and incorporate feasibility assurance procedure. Later, Dixon and Silver (1981) developed a "Greedy" algorithm to guarantee a feasible solution.

Wagner and Whitin (1958) provided a dynamic programming algorithm to solve uncapacitated lot-sizing problems. Although Wagner-Whitin Algorithm provided optimal solutions, it required high computational resources. Silver and Meal (1969) developed faster heuristics to counter this problem, which was known as Silver-Meal Heuristic. They later (1973) used the heuristic to handle cases with deterministic time-varying demand rate and discrete opportunities for replenishment. Many (Ozdamar and Bozyel, 2000), carrying over of setups (Sox and Gao, 1999), multiple items (Eppen and Martin, 1987; Pfeiffer,1999; Sambasivan and Schmidt, 2002; and Wolsey, 2002), and supplier selection (C. Basnet and Janny M. Y. Leung, 2005).

Algorithms based on cutting planes method were also used to solve CLS problems. Barany et al. (1984) and Eppen and Martin (1987) exploited concept of cutting planes and branch-and-bound. Pochet and Wolsey (1988) proposed a shortest-path reformulation which was similar to Eppen and Martin's work. Janny M. Y. Leung et al. (1989) studied the polyhedral structure of an integer programming formulation of a single-item capacitated version, and used the results to develop solution methods for multi-item applications. The set of valid inequalities introduced could be used to develop an efficient cutting plane/branch-and-bound procedure. Hindi (1995) presented a branch-and-bound algorithm to solve the CLS problem.

Solving the CLS problem by Lagrangean relaxation and decomposition techniques is another research area in literature. Diaby et al. (1992) used this method to solve a very-large-scale CLS problem with setup times. Millar and Yang (1993) used

Lagrangean decomposition methodology to solve a CLS problem without production costs. They used a Lagrangean method heuristic to consider a CLS problem with backordering later in 1994.

There are also some papers recognizing the importance of setup times in the traditional modeling structures of CLS problems. Trigerio et al. (1989) used a heuristic based on Lagrangean relaxation to solve the CLS problem with setup times. Some researchers understood that there was a trend toward reducing setup times and argued that the amount of setup times in many industries cannot be minimized to zero, such as Manne (1958), Dzielinski and Gomory (1965), Lasdon and Terjung (1971), Bahl (1983), Trigerio (1989), Trigerio et al.(1989), Lozano et al. (1991), Pochet and Wolsey (1991), Diaby et al. (1991), Diaby et al. (1992), Salomon et al. (1997), Gopalakrishnan et al. (2001). Many researchers believed that the inclusion of setup times was simply an extension of the problem without setup times. Trigerio et al. (1989) rejected this assertion using an illustrative example, which would be discussed more in detail in next chapter. Therefore, the CLS problem with setup time consideration is still worthy of further investigation.

2.1.3 Research on Fixed-Charge-Transportation-typed models for CLS problems

The fixed-charge transportation problem requires each cell in the transportation tableau to have fixed cost and variable cost. Kennington (1976) did a computational study using a branch-and-bound code for the fixed-charge transportation problem.

Kennington and Unger (1976) proposed a new branch-and-bound algorithm for such a kind of problem. Hirsch and Dantzig (1968) concluded that an optimal solution could be found in an extreme point of the constraint set for any fixed charge problem. Balinski (1961) substituted an approximate linear objective function for the non-linear one to solve the problem. Cooper and Drebes (1967) proposed an extreme point heuristic based algorithm. Steinberg (1970) developed an adjacent extreme point heuristic to improve the efficiency. Walker (1976) also considered adjacent extreme points to extend Denzler (1969)'s work to degeneracy cases. Diaby (1991) gave a successive linear approxiamation procedure for the generalized fixed-charge transportation problem.

Gilbert and Madan (1991) proposed a model of CLS problem without backordering which was a variation of the fixed-charge transportation problem. The difference was that a fixed charge is shared by a group of cells in the transportation tableau, rather than a single cell, a single row, or a single column. Madan and Gilbert (1992) later presented an exact solution algorithm to the same problem without backordering. The solution procedure was based on the algorithm developed by Trotter and Shetty (1974) for general integer-programming problems. So (1997) improved Gilbert and Madan's (1991) algorithm and extended the model to a case with backordering consideration. Madan et al. (2001) improved their heuristic (1991) by using a better lower bounding procedure. And Cheng et al. (2001) extended their work to a backordering case.

The model is formulated as a variation of the fixed-charge transportation problem, rather than a traditional MIP model for the CLS problem mentioned before. The advantages of this type of model will be discussed more in detail in next chapter.

However, for this kind of modeling structure for CLS problems, few papers discussed the case with setup times. Our work is mainly following the key principles of these papers and considering a CLS problem with setup times.

2.2 Research in Genetic Algorithm (GA)

The aim of creating artificial intelligence and artificial life can be tracked back to the earliest pioneers, like Alan Turing, John von Neumann, and Norbert Wiener (Melanie, 1998). They contributed to not only calculating missile trajectories and deciphering military codes, but also modeling the brain, mimicking human learning, and simulating biological evolution. These research interests later, in the 1980s, had grown into three fields: neural networks, machine learning and evolutionary computation. Among all, Genetic Algorithms may be one of the most interesting developments.

The main idea in the evolutionary systems is to evolve a population of candidate solutions to a problem, using operations inspired by natural genetic variation and natural selection. Genetic Algorithms (GA) in particular became popular through the work of John Holland in the early 1970s, and particularly his book *Adaptation in Natural and Artificial Systems* (1975). GAs are implemented in a computer simulation in which a population of abstract representations (called chromosomes or the genotype of the genome) of candidate solutions (called individuals, creatures, or phenotypes) to an optimization problem evolves toward better solutions. Holland

(1975) presented the GA as an abstraction of biological evolution and gave a theoretical framework for adaptation under the GA in his book. He introduced a formalized framework for predicting the quality of the next generation, known as Holland's Schema Theorem. Research in GAs remained largely theoretical until the mid-1980s, when The First International Conference on Genetic Algorithms was held in Pittsburgh, Pennsylvania.

Holland had two goals: to improve the understanding of natural adaptation process, and to design artificial systems having properties similar to natural systems (Goldberg, 1988). The basic idea is as follows: the genetic pool of a given population potentially contains the solution, or a better solution, to a given adaptive problem. This solution is not "active" because the genetic combination on which it relies is split between several subjects. Only the association of different genomes can lead to the solution.

Holland method was especially effective because he not only considered the role of mutation (mutations improve very seldom the algorithms), but he also utilized genetic recombination, in other words, crossover (Emmeche, 1994) . The crossover of partial solutions greatly improves the capability of the algorithm to approach, and eventually find, the optimum.

Genetic algorithms find applications in bioinformatics, phylogenetics, computational science, engineering, economics, chemistry, manufacturing, mathematics, physics and other fields. We find that GA is a proper approach to solve our formulated model structure. We reviewed some general work on GA to better understand its main procedures. Grefenstette (1990) provided a general guide of the GENESIS system for function optimization based on genetic search techniques in

order to encourage the experimental use of genetic algorithms on realistic optimization problems. Khouja et al. (1998) investigated the use of GA for solving the Economic Lot Size Scheduling Problem (ELSP) which was formulated using the Basic Period (BP) approach with a formulation that was ideally suited for using GA. Hernandez and Suer (1999) proposed a GA approach for a single-item, and single-level lot-sizing problem with an uncapacitated, no shortages allowed case. Staggemeier and Clark (2001) listed some papers on the application of GA to lot-sizing problems, many of which were focusing on the production planning problem with a short-range decision-making dimension on sequencing and shop floor control, such as Potts and Wassenhove (1992), Hyun and Kim (1998), and Sikora (1996), or on the multi-level problems, such as Kim & Kim (1996), Kimms (1999) and Ip et al. (2000). Few papers are on the application of GA on the medium-range dimension of CLS problems, which is beyond the scope for our work.

2.3 Conclusion

We reviewed the literature mainly on the single-level CLS problem. CLS problems with setup time consideration is an interesting area that is worth further study. We also reviewed some papers on GA studies. The application of GA method to some kinds of CLS problems may be promising.

Chapter 3

Problem Description and Formulation

The formulation of the problem will be presented in this chapter. A comparison between our formulation and a traditional one in the literature is also discussed.

3.1 The formulation

In this section, we present our formulation for the CLS problem with setup time consideration. This problem involves J products, K types of production capacities, and T time periods. We denote k as types of capacities, k=1, 2,…, K. The cost of capacity k is P_k. We also require the cost of capacity k always be less than that of capacity k+1 (i.e., $P_k < P_{k+1}$, where k=1, 2,…, K). Each product needs a setup for a period in which it is produced. The cost of a setup for product j is S_j. The downtime consumed by a setup operation is significant (i.e., the setup time t_j for product j is

greater than zero). Each product may be produced using one type of capacity source or a combination of these sources. b_j is an absorption rate in order to indicate the amount of capacity required to produce one unit of product j. Each product j has a holding cost of H_j per unit per period. Backordering is not allowed. Further, we assume that there are no beginning inventories. If beginning inventories do exist, they can be used to satisfy the demands in the first period. This approach will ensure no beginning inventories and result in the reduced demands in the first period.

To formulate the model, we define:

J	Number of products
K	Number of types of production capacities available
T	Number of time periods in the planning horizon
D_{jn}	Demand of product j in period n
C_{km}	Units of capacity type k available in period m (capacity is measured in time units)
S_j	Setup cost of product j
t_j	Setup time of product j

H_j Unit holding cost per period of product j

P_k Capacity cost per unit of capacity type

b_j An absorption rate that indicates the amount of capacity required to produce a unit of product j

Y_{jm} A binary variable that indicates the presence or absence of a setup for product j in period m

X_{jkmn} The quantity of product j produced using capacity source k in period m, used to meet the demand in period n

The problem *P1* is formulated as follows:

P1: Minimize

$$\sum_{j=1}^{J}\sum_{k=1}^{K}\sum_{m=1}^{T}\sum_{n=m}^{T}(n-m)(X_{jkmn} * H_j) + \sum_{j=1}^{J}\sum_{m=1}^{T}(S_j * Y_{jm}) + \sum_{j=1}^{J}\sum_{k=1}^{K}\sum_{m=1}^{T}\sum_{n=m}^{T}(X_{jkmn} * b_j * P_k) \quad (1)$$

Subject to

$$\sum_{j=1}^{J}\sum_{k=1}^{K}\sum_{n=m}^{T}(b_j * X_{jkmn}) + \sum_{j=1}^{J}(t_j * Y_{jm}) \le \sum_{k=1}^{K} C_{km} \qquad m=1, 2, \ldots,T \qquad (2)$$

$$\sum_{k=1}^{K}\sum_{m=1}^{n} X_{jkmn} = D_{jn} \qquad j = 1,2,\ldots,J; \quad n = 1,2,\ldots,T \qquad (3)$$

$$Y_{jm} = \begin{cases} 1 & \text{if } \sum_{k=1}^{K}\sum_{n=m}^{T} X_{jkmn} > 0 \\ 0 & \text{otherwise} \end{cases} \qquad j = 1,2,\ldots,J; \quad m = 1,2,\ldots,T \qquad (4)$$

$$X_{jkmn} \ge 0 \qquad j = 1,2,\ldots,J; \quad k = 1,2,\ldots,K; \quad m = 1,2,\ldots,T; \quad n = 1,2,\ldots,T \qquad (5)$$

The objective function (1) requires the minimization of the total cost, containing the sum of inventory holding costs, setup costs and capacity costs. Constraint (2) states that the capacity in each period cannot be exceeded. Notice that setup times are significant. The satisfaction of demand of each product in each period is ensured by the constraint (3). In constraint (4), a setup is required if production of product j takes place in period m. Constraint (5) ensures non-negativity of the quantity of products produced.

P1 is an interesting variation of the traditional fixed-charge transportation problem. It differs from most fixed-charge transportation problems in that each fixed charge is

associated with a group of cells, rather than a single cell, a single row, or a single column.

Figure 1 is a tableau showing the transportation problem for *P1*. The supplies in the transportation tableau are denoted by C_{km}, the quantity of source type k production capacity available in the period m. The demands are denoted by D_{jn}, the quantity of product j demanded in period n. The rows are ordered as *(1,1), ..., (K,1), ..., (1,T), ..., (K,T)*. The columns are ordered as *(1,1), ..., (J,1), ..., (1,T), ..., (J,T)*. The cell in row *(k,m)*, $1 \le k \le K$, $1 \le m \le T$ and column *(j,n)*, $1 \le j \le J$, $1 \le n \le T$, will be denoted by *((k,m),(j,n))*. The flow assigned to this cell corresponds to the variable X_{jkmn}, and the value of the setup variable Y_{jm} is implicitly determined by constraint (4). As backordering is not allowed in our model, a tableau solution without flows in backorder cells is a feasible solution to *P1*.

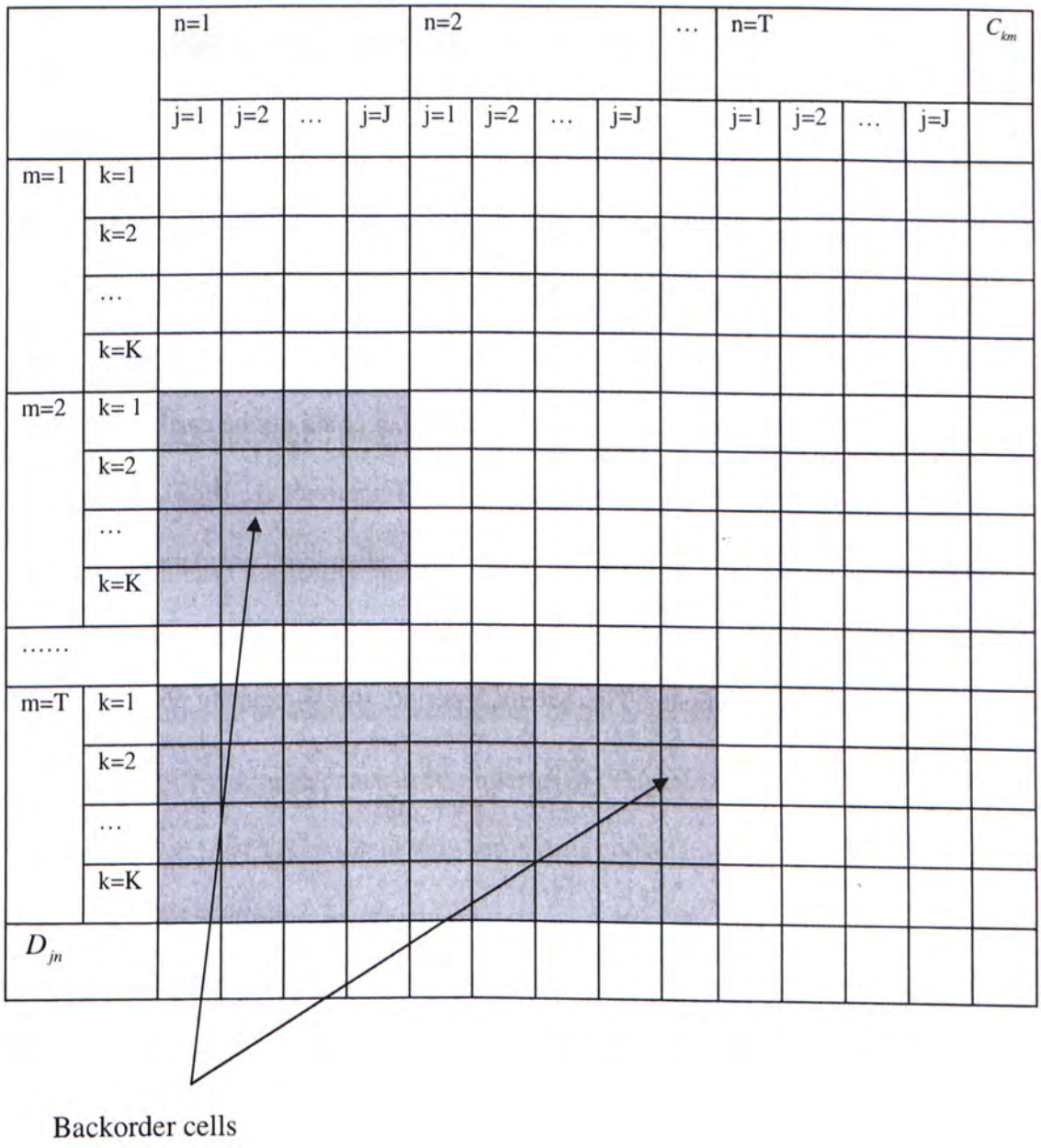

Figure 2: An example of transportation tableau

3.2 Comparison with the traditional formulation

Although our research follows Gilbert and Madan(1991) and Cheng et al. (2001), our formulation considers the significance of setup times. We recognize that there is a trend in reducing setup times in production. In many industries, however, the amount of setup times cannot be easily reduced to zero. Also, it could not be easily regarded as an extension of the problem without setup times (Trigerio et al., 1989). The CLS problem is much more difficult when setup times are taken into consideration.

Trigerio et al. (1989) used an illustrative example to show the reason why setup time consideration is significant. The 2-item, 3-period, single-capacity problem is shown in Figure 3 below:

Item	Setup time	Unit production time	Demand by period		
			1	2	3
1	10	1	10	0	11
2	4	1	0	6	0
Lot-for-lot usage			20	10	21
Capacity available			20	20	20

Figure 3: An example of a CLS problem with setup times

It seems easy to obtain the feasible solution since there only exists 1 unit of overtime in period 3 which should be eliminated. To eliminate the overtime, it must shift some production quantities to previous periods. However, the first period cannot accept additional production, obviously. Period 2 cannot accept item 1 either, because the setup time required is too high. Hence, even if cumulative lot-for-lot capacity usage (including setup time) does not exceed cumulative capacity available, and the average utilization is only 85% (= 51/60), a feasible solution may not be found to this problem.

Lack of an easy feasibility check presents a serious difficulty for an algorithm to such kind of problem. And since all problems of meaningful size are too large to solve optimally, no systematic investigation of the algorithm's ability to correctly solve these difficult problems can be undertaken (Trigerio et al., 1989). Therefore, we realize that the CLS problem with setup time consideration is still an interesting research topic for further investigation.

The formulations by Gilbert and Madan (1991) and Cheng et al. (2001) are different from the traditional formulations used in the CLS research. We adopt the Gilbert and Madan (1991)-like formulation in this research work, because their formulation provides additional benefits. In this section, we compare our formulation with a traditional one to show the advantages.

A typical CLS formulation (Trigerio et al., 1989) is shown as follows:

Minimize:

$$\sum_t \sum_i H_{it} I_{it} + \sum_t \sum_i C_{it} X_{it} + \sum_t \sum_i S_{it} Y_{it} \quad (6)$$

Subject to

$$I_{i,t-1} + X_{it} - I_{it} = d_{it} \quad \text{for all } i, t \quad (7)$$

$$\sum_i b_i X_{it} + \sum_i s_i Y_{it} \le CAP_t \quad \text{for all } t \quad (8)$$

$$X_{it} - MY_{it} \le 0 \quad \text{for all } i, t \quad (9)$$

$$Y_{it} = 0 \text{ or } 1 \quad \text{for all } i, t \quad (10)$$

$$X_{it} \text{ and } I_{it} \ge 0 \quad \text{for all } i, t \quad (11)$$

$$I_{i,0} = 0 \quad \text{for all } i \quad (12)$$

Where

H_{it} is the unit holding cost of item i in period t

I_{it} is the end-of-period inventory of item i in period t

C_{it} is the unit production cost of item i in period t

X_{it} is the production quantity of item i in period t

S_{it} is the setup cost of item i in period t

D_{it} is the demand of item i in period t

b_i is the capacity consumption rate in producing an unit of item i

s_i is the setup time of item i

CAP_t is the quantity of capacity available in period t

M is a very big number

Table 1 below summarizes the differences in the two formulations. In a 3-product, 12-period and single-type-capacity problem, there are 61, 468 and 36 input variables, output variables and binary variables respectively in our problem formulation. While in the traditional problem formulation, there are 96, 108 and 36 input variables, output variables and binary variables, respectively. Note that in any equivalent formulation, we assume $H_j = H_{it}$ and $S_j = S_{it}$ for all t.

	Our problem formulation	Traditional problem formulation
Input variables	$H_j, S_j, b_j, P_k, C_{km}, D_{jn}, t_j$	$H_{it}, S_{it}, b_i, C_{it}, CAP_t, d_{it}, s_i$
Output variables	X_{jkmn}, Y_{jm}	X_{it}, Y_{it}, I_{it}
Binary variables	Y_{jm}	Y_{it}
Total no. of constraints	4	6
No. of constraints containing the binary variables	2	3
No. of input variables	$4J + K + K*T + J*T$	$4J + T + 2J*T$
No. of output variables	$J*K*T^2 + J*T$	$3J*T$
No. of binary variable	$J*T$	$J*T$

where J, T and K represent the number of products, time periods and types of capacity available in the problem.

Table 1: Comparison between our formulation and traditional formulation

In comparison, our problem formulation provides more output variables. For example, our formulation is able to show which period's demand will be satisfied by a production run. Although our formulation is structurally more complicated, it uses the same number of binary variables and makes use of many continuous variables. The structural complication should not significantly increase the computational effort.

3.3 Conclusion

In this chapter, we formulate the CLS problem with setup times without backordering consideration as a type of the fixed-charge transportation problem. A comparison of our formulation and the traditional one is also presented to address the advantages of our formulation.

Chapter 4

Description of the Heuristic

Our heuristic is based mainly on Genetic Algorithms (GA), which contains the following procedures: initialization, selection, crossover, mutation (if any), evaluation and termination. The setup time in the constraint (2) is represented by a setup string, and used to transform the original problem into a modified one that is similar to the traditional fixed-charge transportation problem, from which the production quantities could be determined. A consistency test is used to judge whether the solution generated by the two steps discussed above is feasible or not to the original problem. The feasible solutions according to different setup strings would be kept in the feasible solution pool, from which the population of candidate solutions for GA is formed. The algorithm is terminated when the pre-determined iteration limit is reached.

The flow chart of the whole heuristic algorithm is shown in Figure 4, and the flow chart of the procedure to get the initial feasible solution pool is shown in Figure 5. Each step in the charts would be discussed in detail in this chapter.

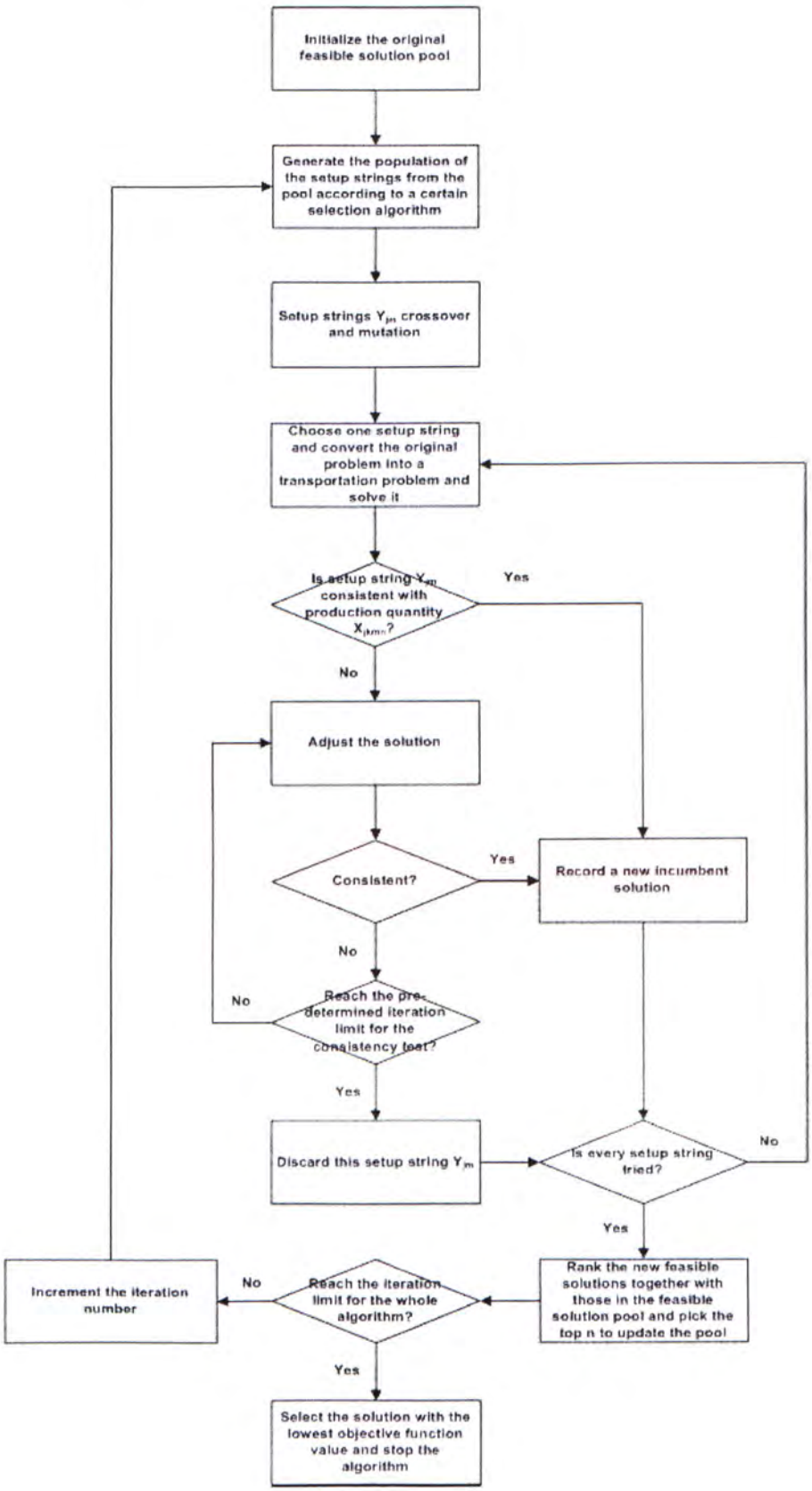

Figure 4: A flow chart of the heuristic algorithm

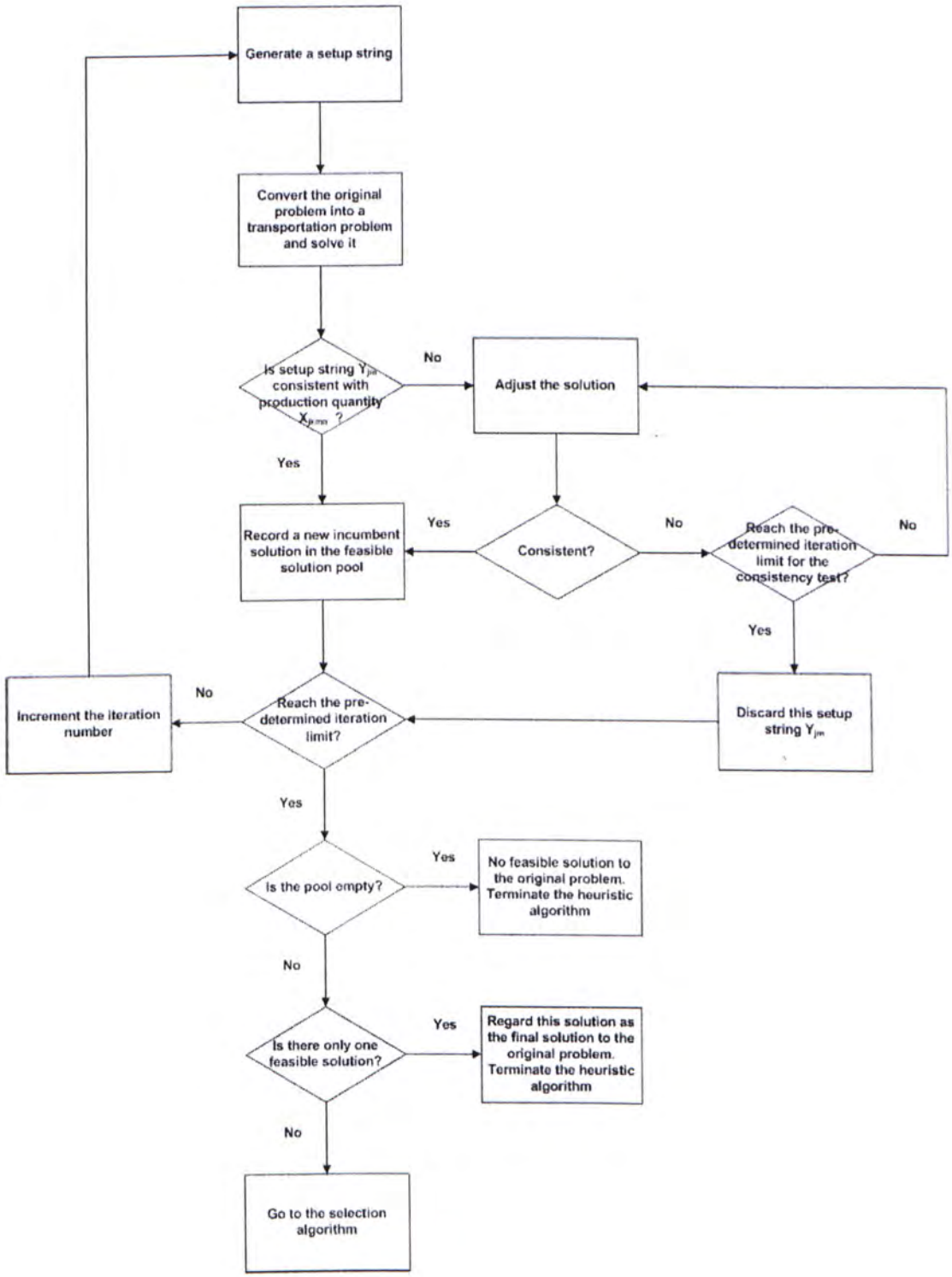

Figure 5: A flow chart of the procedure to generate the original feasible solution pool

4.1 Initialization

As previously mentioned, GA mainly comprises six steps: initialization, selection, crossover, mutation (if any), evaluation and termination.

This initialization procedure is to build up the initial feasible solution pool, from which the population of candidate solutions of GA is generated. It mainly contains three phases: setup string generation, the transportation problem, and the consistency test. Figure 5 above illustrated the main procedure.

4.1.1 Setup string generation

We use three main procedures to generate setup strings here: Wagner-Whitin Algorithm, a variation of Wagner-Whitin-Algorithm-generated solutions, and random generation of setup strings (i.e., Y_{jm}). Half of the solutions in the initial feasible solution pool are with setup strings that are randomly generated. Another half are with those generated from the first two ways.

Firstly we separate the original problem into J uncapacitated, single-item and independent problems, each with the setup cost, inventory holding cost and production cost consideration in a T-period planning horizon. To achieve this separation, we ignore the setup time and capacity restraints temporally. We use the well-known dynamic programming method, Wagner-Whitin Algorithm (Wagner and

Whitin 1958, and Sven Axsäter, 2006), to solve these sub-problems of each product and obtain a setup string by simply integrated the three sub-strings.

Three costs are considered in Wagner-Whitin Algorithm, which are holding cost, unit capacity cost and setup cost. In our problem, k types of capacity resources are considered, such as regular time, overtime, and subcontracting. When the original problem is decomposed into uncapacitated and single-item problems, only type 1 of the capacity is chosen for consideration for its lowest capacity cost. According to the assumption that we make the cost of capacity k always be less than that of capacity k+1 (i.e., $P_k<P_{k+1}$ where k=1, 2,…, K.), the cost of type 1 of capacity is the lowest.

For each product j, we introduce the following notations:

f_v minimum costs over periods 1,2,…, v, i.e., when we disregard periods v+1, l+2,…, T,

$f_{v,t}$ minimum costs over periods 1,2 ,…, v, given that the last setup is in period t, ($1\le t\le v$).

We can get that

$$f_v = Min_{1\le t\le v} f_{v,t} \tag{13}$$

since the last setup must occur in some period in the optimal solution.

Assume that the demands occur at the beginning of a period. It is obvious that:

$$f_0 = 0 \tag{14}$$

$$f_1 = f_{1,1} = S_j + P_1 * D_{j1} \tag{15}$$

Assume now that we know f_{t-1} for some t>0. It is then easy to obtain $f_{v,t}$ for v≥t as following form:

$$f_{v,t} = f_{t-1} + S_j + P_1 * (D_{jt} + D_{jt+1} + ... + D_{jv}) + H_j * [D_{jt+1} + 2D_{jt+2} + ... + (v-t)D_{jv}] \tag{16}$$

where S_j, P_1, H_j and D_{jn} are the setup cost of product j, capacity cost per unit of capacity type 1, unit holding cost per period of product j, and demand of product j in period n, respectively, according to the notation shown in Chapter 3.

After assigning the costs and demands for each product in each period, these independent uncapacitated and single-item problems can be solved by Wagner-Whitin Algorithm. Therefore a set of or several sets of setup decisions is obtained. We put the independent setup strings for each product together to get a whole setup sting Y_{jm} to the original problem *P1*. The string would be regarded as an input to transform *P1* to a variation of a transportation problem. We can get the production quantities by solving the transportation problem, which would be discussed later in this chapter. Note that there is a possibility that this setup string coming from three feasible setup strings for each product respectively may not be feasible to the whole problem, or may not be consistent with the production quantities decided by solving the transportation problem, a consistency test would be introduced to check the

feasibility of the final solution of decision variables X_{jkmn} and Y_{jm} to *P1*, which may be discussed in detail later.

The second way to generate setup strings is to create minor changes to the strings generated by Wagner-Whitin Algorithm. We let some positions of the original string change from 0 to 1 or from 1 to 0. The number of positions that need to be changed, and the specific position to change are both decided stochastically. When one position of a setup string is chosen for variation, its value would be changed from its current value, 0, to 1, and vice versa.

The third way is to generate the setup strings randomly. This will widen the search area covering a bigger solution space. In addition, it is also useful in avoiding the situation that the final solution may be trapped in a local optima.

4.1.2 Transportation problem

Given the set of setup decisions generated in the first subsection, the original problem *P1* can be transformed into a transportation problem *TP1* (Thizy and Van Wassenhove, 1985) as shown below:

TP1: Minimize

$$\sum_{j=1}^{J}\sum_{k=1}^{K}\sum_{m=1}^{T}\sum_{n=1}^{T}\beta_{jkmn} * X_{jkmn} \tag{17}$$

Subject to

$$\sum_{j=1}^{J}\sum_{n=1}^{T} X_{jkmn} \le C_{km}^{*} \qquad \begin{array}{l} k = 1,2,\ldots, K \\ m = 1,2,\ldots T \end{array} \tag{18}$$

$$\sum_{k=1}^{K}\sum_{m=1}^{n} X_{jkmn} = D_{jn}^{*} \qquad \begin{array}{l} j = 1,2,\ldots,J \\ n=1,2,\ldots,T \end{array} \tag{19}$$

$$X_{jkmn} \ge 0 \qquad \begin{array}{l} j = 1,2,\ldots,J \\ k = 1,2,\ldots,K \\ m = 1,2,\ldots T \\ n = 1,2,\ldots,T \end{array} \tag{20}$$

where D_{jn}^{*} denotes the modified demand of product j in period n; C_{km}^{*} denotes the modified amount of capacity of type k available in period m; X_{jkmn} is the production quantity of product j using capacity of type k in period m to meet the demand in period n; and β_{jkmn} is the cost assigned to the cell ((k,m), (j,n)) in the transportation tableau.

The objective function of *TP1* (17) is to minimize the sum of inventory holding cost, capacity cost and setup cost. The constraints (18)-(20) require that all period demands are satisfied with capacity limitations, and the production quantities must be positive.

TP1 can be represented by a transportation tableau shown in Figure 6, in which rows are denoted by the modified supplies of production capacities and columns are denoted by the modified demands. The rows are ordered as (1,1), ..., (K,1), ..., (1,T), ..., (K,T). The columns are ordered as (1,1), ..., (J,1), ..., (1,T), ..., (J,T). The cell in row (k,m), $1 \le k \le K$, $1 \le m \le T$ and column (j,n), $1 \le j \le J$, $1 \le n \le T$, will be denoted by ((k,m),(j,n)). The flow assigned to this cell is the decision variable X_{jkmn}.

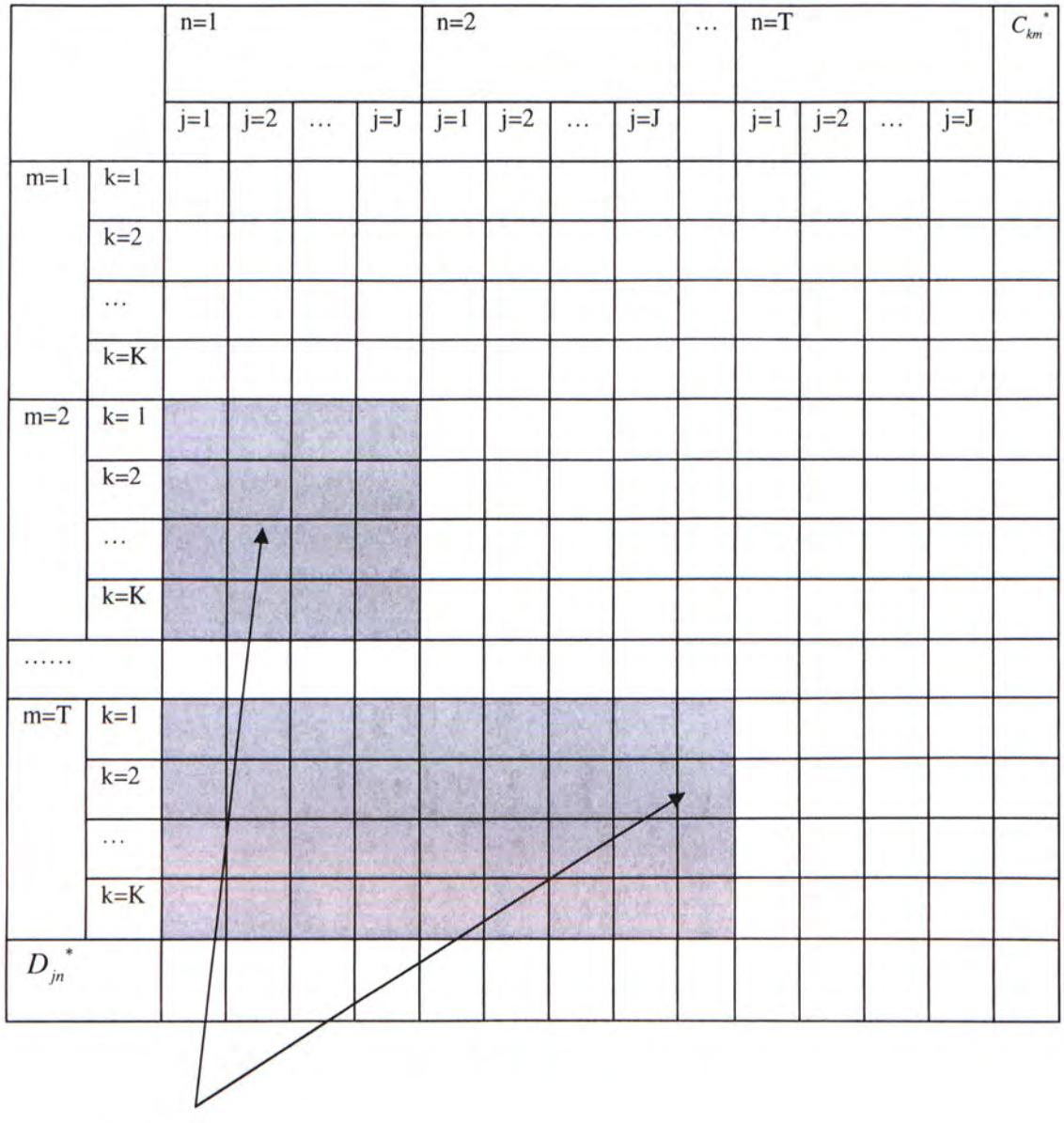

Figure 6: A transportation tableau for the modified problem

The modified supplies C_{km}^{*} and modified demands D_{jn}^{*} are given by the following scheme:

$$C_{1m}^{*} = C_{1m} - \sum_{j=1}^{J}(t_j * Y_{jm}^{*}) \qquad \text{for } k=1 \qquad (21)$$

$$C_{km}^{*} = C_{km} \qquad \text{for } k=2,\ldots,K \qquad (22)$$

$$D_{jn}^{*} = b_j * D_{jn} \qquad \text{for } j=1,2,\ldots,J \text{ and } n=1,2,\ldots,T \qquad (23)$$

where

$Y^{*} = \{Y_{jm}^{*}\}$ Denote the set of setup decisions obtained from the three procedures discussed in section 4.1.1

Since capacity 1 (i.e., regular time production capacity) has the lowest setup cost, it will encourage all setup operations to be performed using capacity 1. Note that we assume the original capacity type 1 in each period is enough for a setup, otherwise it may be unreasonable in the real industry. Therefore, setup time is consumed by the setup operations in capacity type 1 and the capacities of other types remain unchanged.

We apply the concept of fixed charges to estimate the nominal cost of the cells in the transportation tableau. A fixed charge is shared by a group of cells in the tableau, instead of a single cell, a single row or a single column. In the objective function of

TP1 (17), the cost coefficient is denoted by β_{jkmn}. Different cells in the transportation tableau have different values of β_{jkmn}, which are computed in the following costing scheme. To ensure that flows in backordering cells will not occur, a positive infinite value M is assigned to all backorder cells (i.e., $m > n$) in the transportation tableau.

For non-backorder cells with setups planned, according to the setup-string generation procedures discussed in the last section (i.e., $m \leq n$ and $\sum_{k=1}^{K}\sum_{n=1}^{T} X_{jkmn} > 0$),

$$\beta_{jkmn} = [(n-m) * H_j + P_k * b_j] / b_j \tag{24}$$

For non-backorder cells without setups planned, according to the setup-string generation procedures discussed in the last section (i.e, $m \leq n$ and $\sum_{k=1}^{K}\sum_{n=1}^{T} X_{jkmn} = 0$),

$$\beta_{jkmn} = [(n-m) * H_j + P_k * b_j] / b_j + S_j / Q \tag{25}$$

where

$$Q = \min\left\{ \sum_{t=m}^{\min\{m+w,T\}} \overline{D_{jt}^{*}} \;,\; \sum_{k=1}^{K} \overline{C_{km}^{*}} \right\} \tag{26}$$

w is the window size (w=0,1,…,T-1)

$\overline{D_{jt}^{*}}$ is the unmet demand of product j in period t

$\overline{C_{km}^{*}}$ is the unused capacity of type k in period m

To calculate the portion of fixed charge shared by a group of cells, we introduce a window size concept where we consider production quantity enough to satisfy demand in current period, in next period, or in *T-1* periods. The window size concept

provides a more accurate and better estimate of the nominal costs in the transportation tableau. The window size is denoted by the variable *w*, *where w = 0, 1,...,T-1*. Products produced in period *m* will be held for *w* period(s) to satisfy the demand from period *m* to *m+w*. So, there will be *K*(w+1)* cells sharing the same setup cost. Total demand associated with these cells will be equal to $\sum_{t=m}^{\min\{(m+w),T\}} \overline{D_{jt}}^{*}$. Hence, the flow assigned to the cell (i.e., *Q*) will be given by

$$Q = \min\left\{ \sum_{t=m}^{\min\{m+w,T\}} \overline{D_{jt}}^{*} \;,\; \sum_{k=1}^{K} \overline{C_{km}}^{*} \right\}$$

The above costing scheme strongly discourages the existence of flow in backorder cells. For non-backorder cells, the costing scheme encourages production in the cells with setups planned according to the pre-determined setup string, by assigning lower costs to these cells. For non-backorder cells without setups planned, an additional component is added to the costs of these cells additionally. The term S_j / Q represents the fixed charge shared by the cell *((k,m), (j,n))* if an additional setup is required for production. With this addition, the costing scheme encourages production in the cells with setups which is pre-determined.

A feasible tableau solution corresponds to a feasible solution to *TP1*. The procedure of solving the transportation problem *TP1* involves two phases.

Phase 1 is the generation of an initial feasible solution by applying a procedure similar to Vogel's approximation method (Reinfeld and Vogel, 1958). The difference is that the cost coefficients consider the fixed charge associated with each cell in the tableau, as mentioned in the above costing scheme.

Selection of cells for initial feasible solution in the Vogel's approximation method is based on the difference between the two lowest-cost cells leaving an origin or entering a destination. This difference is called the penalty that indicates the highest increase in cost by departure from the lowest cost allocations. The maximum possible value is assigned to the variable associated with the cell having the largest penalty. The cost coefficients of all cells in the tableau are "greedy" costs, and represent the average rate of increase in the objective function of *TP1* if the corresponding cells enter the basis.

Instead of fixing the holding period window size to a specific value between *0* to *T-1*, procedure of Phase 1 is repeated by T times to obtain the best initial feasible tableau solution among all window sizes ($w = 0,1,\ldots,T-1$). Among all window sizes, the best initial solution that gives the smallest value of total cost is selected to enter phase 2.

Phase 2 attempts to improve the solution obtained in phase 1 by replacing variables in the basis. It is similar to a primal network simplex algorithm, and the major difference is that it considers both fixed cost and variable cost in pricing the pivots. We call it the procedure of pivoting. At each iteration, the non-basic cell variable selected to enter the basis is the one that will result in a reduction of the total cost. Note that the non-basic cell selected must not be a backorder cell, since backordering is not allowed in our model. Given an initial feasible solution, the corresponding change in the objective function will be given by $\Delta TC = \Delta S + \Delta V$, where ΔS and ΔV are the changes in setup cost and variable cost respectively if a new non-basic cell enters the basis. At the end of phase 2, a set of production quantities is generated and denoted by $X_{jkmn}^{*} = \{X_{jkmn}^{*}\}$.

Note that for the transportation problem, usually the total demand is equal to the total supply. It is obvious that the problem is infeasible if the total demand exceeds the total supply. In addition, in cases without backordering, it implicitly tells us that the problem is infeasible if the total cumulative demand for a given period is greater than the total cumulative capacity available of the given period. In the cases that the total demand is less than the total supply, then an additional column (dummy demand column) is introduced, with the demand quantity equaling the excess supply.

If the total demand for products over the planning periods is less than the total available capacity, we define $D_{J+1\,T+1}$ as follows:

$$D_{J+1T+1} = \sum_{k=1}^{K}\sum_{m=1}^{T} C_{km} - \sum_{j=1}^{J}\sum_{n=1}^{T} D_{jn} \tag{27}$$

where D_{J+1T+1} is demand for the dummy column (J+1, T+1). We then add the following demand constraints for dummy demand.

$$\sum_{k=1}^{K}\sum_{m=1}^{T} X_{J+1kmT+1} = D_{J+1T+1} \tag{28}$$

Constraint (18) will be changed to equality constraints as shown below.

$$\sum_{j=1}^{J}\sum_{n=1}^{T} X_{jkmn} + X_{J+1kmT+1} = C_{km}^{*} \qquad \begin{array}{l} k = 1,2,\ldots,K \\ m = 1,2,\ldots T \end{array} \tag{29}$$

Thus, a standard traditional transportation model is formulated.

The procedures of two phases mentioned before are shown as follows:

Procedure of Phase 1

Step 1: Initialize the window size to zero (i.e, $w = 0$).

Step 2: Assign costs to all the cells in the tableau according to the costing scheme.

Step 3: For each unmarked row, compute row penalty E_{km}, which is the difference between the lowest nominal cost and the second lowest nominal cost in the row (k,m).

Step 4: For each unmarked column, compute column penalty G_{jn}, which is the difference between the lowest nominal cost and the second lowest nominal cost in the column (j,n).

Step 5: Compute $E_{k'm'} = \max\{ E_{km} \}$ for all $k = 1,2,\ldots K$ and $m = 1,2,\ldots,T$, where $k' \in \{1,2,\ldots,K\}$; $m' \in \{1,2,\ldots,T\}$. $G_{j'n'} = \max\{ G_{jn} \}$ for all $j = 1,2,\ldots J$ and $n = 1,2,\ldots,T$, where $j' \in \{1,2,\ldots,J\}$; $n' \in \{1,2,\ldots,T\}$.

If $E_{k'm'} > G_{j'n'}$, set $\beta_{jkmn} = \min\{ \beta_{jk'm'n} \}$ for all $j = 1,2,\ldots J$ and $n = 1,2,\ldots,T$.

Else set $\beta_{jkmn} = \min\{ \beta_{j'kmn'} \}$ for all $k = 1,2,\ldots K$ and $m = 1,2,\ldots,T$.

The cell $((k',m'), (j',n'))$ is the one selected to enter the basis.

Step 6: Let

$$X_{j'k'm'n'} = \min\left\{\overline{D_{j'n'}}^{*} \ , \ \overline{C_{k'm'}}^{*}\right\} / b_j$$

$X_{j'k'm'n'}$ is the flow assigned to the cell *((k',m'), (j',n'))*.

If $X_{j'k'm'n'} = 0$, mark the column *(j',n')* and return to step 3.

Else continue to step 7.

Step 7: Update the capacity in the row *(k',m')* and the demand in the column *(j',n')*.

If $\overline{D_{j'n'}}^{*} \le \overline{C_{k'm'}}^{*}$,

set $\overline{C_{k'm'}}^{*} = \overline{C_{k'm'}}^{*} - \overline{D_{j'n'}}^{*}$ and $\overline{D_{j'n'}}^{*} = 0$,

mark the column (*j',n'*) to indicate that the corresponding demand is met.

Else

set $\overline{D_{j'n'}}^{*} = \overline{D_{j'n'}}^{*} - \overline{C_{k'm'}}^{*}$ and $\overline{C_{k'm'}}^{*} = 0$,

mark the row (*k',m'*) to indicate the corresponding supply is used up.

If any row or column is unmarked, return to step 3.

Else, a feasible solution is formed and go to step 8.

Step 8: For each feasible tableau solution using window size (*w*), compute

$$TC(w) = \sum_{j=1}^{J}\sum_{k=1}^{K}\sum_{m=1}^{T}\sum_{n=m}^{T}(n-m) * H_j * X_{jkmn} + \sum_{j=1}^{J}\sum_{m=1}^{T} S_j * Y_{jm} + \sum_{j=1}^{J}\sum_{k=1}^{K}\sum_{m=1}^{T}\sum_{n=m}^{T} P_k * b_j * X_{jkmn}$$

If w ≤ *T-1*,

increment the value of window size (i.e, $w = w + 1$),

unmark all rows and columns in the tableau and return to step 2.

Else

Select the best feasible solution with the lowest cost TC_{min}

= min{ $TC(w)$ } for $w = 0,1,\ldots,T-1$.

Then, go to Phase 2.

Procedure of Phase 2

Step 1: Initialize the starting position at the cell *((1,1),(1,1))* in the tableau.

Step 2: Find the next non-basic cell along the same row.

Step 3: Compute S_0, S_1, V_0, V_1,

where S_0 is the fixed cost (i.e., setup cost) of the current tableau,

V_0 is the variable cost (i.e., holding and capacity costs) of the current tableau,

S_1 is the fixed cost if the non-basic cell found enters the basis,

V_1 is the variable cost if the non-basic cell found enters the basis,

Compute $\Delta TC = \Delta S + \Delta V$, where $\Delta S = S_1 - S_0$ and $\Delta V = V_1 - V_0$

If $\Delta TC < 0$ (i.e., the objective function value will be reduced), update the tableau by entering the cell into the basis. Go to step 4.

Else go to step 4.

Step 4: If no non-basic cell in the whole tableau can reduce the objective function value (search the cells in the same row first, and then go to the next row), go to step 5.

Else go back to step 2.

Step 5: Compute the total cost of the new tableau solution

$$TC = \sum_{j=1}^{J}\sum_{k=1}^{K}\sum_{m=1}^{T}\sum_{n=m}^{T}(n-m)*H_j*X_{jkmn} + \sum_{j=1}^{J}\sum_{m=1}^{T}S_j*Y_{jm} + \sum_{j=1}^{J}\sum_{k=1}^{K}\sum_{m=1}^{T}\sum_{n=m}^{T}P_k*b_j*X_{jkmn}$$

4.1.3 Consistency test

The solution obtained from the transportation problem is already feasible to *TP1*, since it satisfies all period demands with capacity limitation and it involves no

backordering. Now we must check whether the solutions from the two steps discussed above are feasible to the original problem *P1*.

We define:

S_1 = the set of cells with setup $Y_{jm}^{*} = 0$ and production quantity $X_{jkmn}^{*} > 0$

S_2 = the set of cells with setup $Y_{jm}^{*} = 1$ and production quantity $X_{jkmn}^{*} = 0$

The following 2 cases are our concern:

Case 1: The set S_1 is not null.

Case 2: The set S_2 is not null.

When S_1 and S_2 are null, a feasible solution is formed according to the consistency of pre-determined setup decisions and the resulted set of production quantities. So when a production takes place, a corresponding setup is assigned. Besides, there is no extra useless setup in the production plan.

When either case 1 or case 2 prevails, it implies that the set of production quantities does not match with the pre-determined setup decisions. Therefore, we must give up the desirable solution of the production quantity from the transportation problem, and try to find another one which is consistent with the pre-determined setup string (i.e., Y_{jm}). Or we should generate a new setup string, to see whether a consistent solution could be found. A total of four integrated procedures are used to resolve the inconsistency of the cells where S_1 is not null (case 1) or S_2 is not null (case 2). In case 1, forcing production out or forcing setup in will be applied

depending on which one can induce a lower total cost. In contrast, forcing production in or forcing setup out will be used in case 2 depending on which one can give a lower total cost.

When we choose the strategy to change the setup string (i.e., Y_{jm}) to eliminate the inconsistency, the setup string is changed. The transportation problem is solved again, according to a new modified C_{km}^{*} from the new setup string. This process of resolving the transportation problem is repeated until a consistent solution results or a fixed iteration limit is reached.

When we choose the strategy to change the production quantities (i.e., X_{jkmn}) to eliminate the inconsistency, a procedure which is similar to the pivoting is run, to find whether some certain production quantities X_{jkmn} could be forced to be consistent with the pre-determined setup string Y_{jm}. If a feasible solution is found, update the original objective function value. Otherwise, we discard the inconsistent solution.

For each case, we will try both these two strategies and put all the consistent solutions into the feasible solution pool, for the use of the selection procedure later. Meanwhile, we will select the setup string Y_{jm} from the pool which is generated by Wigner-Whitin Algorithm with the lowest total cost, change it in different positions to generate other setup strings.

We use this consistency test to filter the inconsistent decision variables X_{jkmn} and Y_{jm}, and regard the consistent pairs as the feasible solution to the original problem *P1*. Then we rank the feasible solutions and pick the first n solutions to form an initial population pool with the size of n, for the selection procedure discussed later.

4.2 Selection

Selection is the process of choosing structures for the next generation from the structures in the current generation. In our problem here, we aim to select the binary variable Y_{jm} from the current feasible solution pool.

While there are many different types of selection, we will cover the most common type here, the roulette wheel selection, also known as Fitness proportionate selection. In roulette wheel selection, individuals are given a probability of being selected that is directly proportionate to their fitness. Proper individuals are then chosen randomly based on these probabilities and produce offspring. The offspring will then bring new solutions to the whole problem, which we rank with the solutions in the current feasible solution pool, and update the pool to a new generation composed by those solutions with the top objective function values.

4.3 Crossover

After we have selected the individuals, we are supposed to somehow produce offsprings with them. The most common solution is something called crossover. If, after crossover, the offspring are different from the parents, then the offspring replace the parents, and are marked for evaluation.

Crossover exchanges alleles among adjacent pairs of the first structures in the new population. (Recall that the population is randomly shuffled in the selection procedure.) It can be implemented in a variety of ways, but there are theoretical advantages to treating the structures as rings, choosing two crossover points, and exchanging the sections between these points (Grefenstette, 1990). The segments between the crossover points are exchanged, provided that the parents differ somewhere outside of the crossed segment. While there are many different kinds of crossover, the most common type is single point crossover.

As you can see from Figure 7, the children take one section of the chromosome from each parent. The point at which the chromosome is broken depends on the randomly selected crossover point. This particular method is called single point crossover because only one crossover point exists.

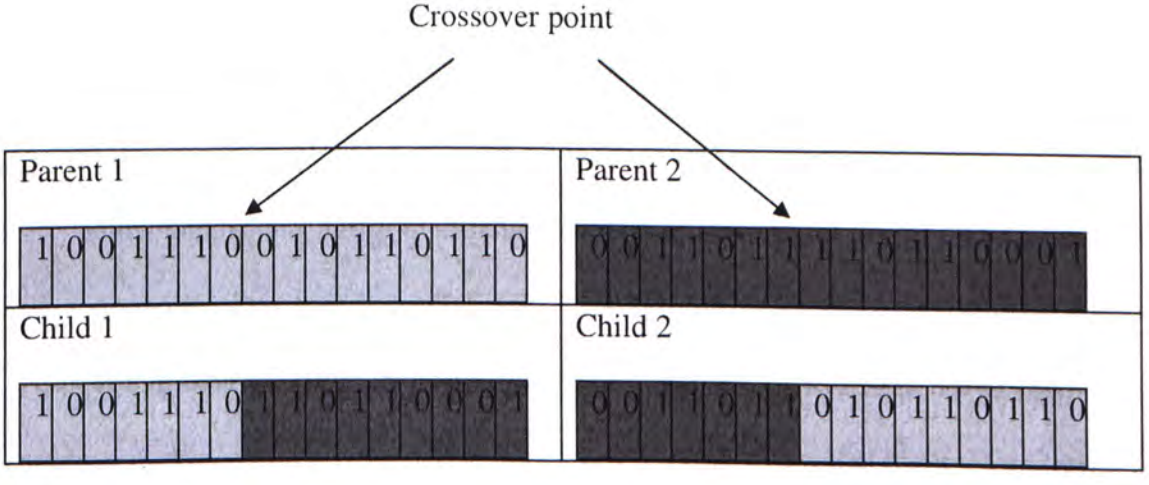

Figure 7: An illustration of single point crossover

In single point crossover, we choose a locus at which you swap the remaining alleles from one parent to the other. This is complex and is best understood visually. Sometimes only child 1 or child 2 is created, but oftentimes both offspring are created and put into the new population. Crossover does not always occur, however. Based on a set probability, no crossover occurs and the parents are copied directly to the new population.

4.4 Mutation

After selection and crossover, we now have a new population full of individuals. Some are directly copied, and others are produced by crossover. In order to ensure that the individuals are not all exactly the same, mutation may be applied to each of these child structures. Each position is given a chance of undergoing mutation. If mutation does occur, the binary value of the mutation position is changed, which means we replace 1 for the position if it is 0 originally, and vice versa. We loop through all the alleles of all the individuals in this way. A visual for mutation is shown in Figure 8 below.

Before mutation:

1	1	0	0	1	0	1	0	1	1	1	1	0	0	0	0

After mutation: Mutation point

1	1	0	0	1	0	1	0	1	1	1	1	1	0	0	0

Figure 8: An illustration of mutation

Mutation is vital to ensuring genetic diversity within the population. In our problem it means the probability of adding or reducing a setup.

4.5 Evaluation

After the crossover and mutation procedure, we take one structure (i.e., one setup string, i.e., Y_{jm}) in the new population as an input and operate the whole procedures of the transportation problem and consistency test discussed above in the first section, to return a new feasible solution, if there exists one, and record it. After every setup string, i.e., Y_{jm} in this new population is evaluated, we rank the new feasible solutions with the solutions in the current feasible solution pool, and update the pool

to a new generation composed by the solutions with the top objective function values, for the next round of selection.

4.6 Termination

So far, one entire iteration of the whole procedure of our algorithm is described. An iteration limit number is set for the iteration procedure. The algorithm will stop when the pre-determined iteration limit is reached. We compare all the solutions in the pool then, and pick the one with the lowest total cost as the best solution among the pool for the whole problem.

4.7 Conclusion

In this chapter, we discuss the whole heuristic algorithm to the CLS problem with setup time consideration described in Chapter 3. The heuristic is based on Genetic Algorithm (GA), which is an iterative procedure that maintains a population of candidate solutions to the objective function, usually operated in the following procedures: initialization, selection, crossover, mutation, and termination. An algorithm to initialize the candidate solution pool is introduced in this chapter. And

the procedures to transform the original problem to a fixed-charge transportation problem are also addressed. The consistency test is presented to test whether a solution is feasible or not to be allowed into the candidate solution pool.

Chapter 5

Design of Experiments and Computational Results

In this chapter, we will design the experiments to test the performance of our heuristic algorithm. The methods to generate the information data and lower bounds are introduced, and analysis of the computational results would also be presented.

A 3-product, 2-capacity, 12-period-planning-horizon problem is designed for a variety of test problems. And all test problems considered three parameters: seasonality of demand (no, medium, and extreme seasonality), the tightness of capacity (80%, 100% and 120%), and the setup cost and setup time level (low, medium and high level).

5.1 Design of experiments

We develop test problems following Graves (1982), Trigerio et al. (1989) and Gilbert and Madan (1991). A 3-product, 2-capacity (regular working time and overtime), 12-period-planning-horizon problem is designed here. The details are given in this section.

- Product demands

In each problem set, the demand of product j in period n is given by:

$$D_{jn}^{\ i} = \sum_{p=1}^{P_j} r_p * z_{jn}^{\ i} \tag{30}$$

where

$z_{jn}^{\ i}$ is a multiplicative seasonality factor of type i for product j in period n,

r_p is the p^{th} random draw from a uniform distribution over the range

$[u_{dp}, l_{dp}]$,

u_{dp}, l_{dp} are the upper and lower limits of the value of r_p respectively,

P_j is equal to 5, 5 and 10 for product j=1, 2 and 3 respectively.

The values of multiplicative seasonality factors are stated in Table 2.

The ranges of demands for each product are summarized in Table 3.

Seasonality	Product	Period											
		1	2	3	4	5	6	7	8	9	10	11	12
None	1	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
(i=1)	2	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
	3	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Moderate	1	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
(i=2)	2	0.8	0.8	0.7	0.5	0.7	1.0	1.0	1.2	1.3	1.5	1.2	1.0
	3	1.0	1.0	1.0	1.2	1.3	1.5	1.3	1.0	0.9	0.7	0.6	0.8
Extreme	1	1.0	0.8	0.6	0.4	0.7	1.0	1.2	0.8	1.6	2.0	2.5	3.0
(i=3)	2	0.8	0.6	0.4	0.5	0.8	0.6	1.2	1.5	2.0	2.0	1.5	1.2
	3	0.7	0.5	0.6	1.0	1.2	1.5	1.8	2.2	1.8	1.0	0.5	0.8

Table 2: Values of multiplicative seasonality factors

Product 1

u_{dp}	20	40	15	25	80
l_{dp}	40	60	25	65	120
u_{sp}	50	50	50	50	50
l_{sp}	150	150	150	150	150

Product 2

u_{dp}	80	15	40	40	55
l_{dp}	120	25	60	60	85
u_{sp}	100	100	100	100	100
l_{sp}	200	200	200	200	200

Product 3

u_{dp}	20	20	30	30	20	30	30	50	60	40
l_{dp}	40	40	50	50	40	70	50	100	80	80
u_{sp}	150	150	150	150	150	150	150	150	150	150
l_{sp}	250	250	250	250	250	250	250	250	250	250

Table 3: Ranges of demands and setup costs for products

- Quantity of capacity available for production

The quantity of regular time capacity available in period m for a time horizon T with seasonality of type i is given by:

$$C_{1m}{}^{i} = \left[\sum_{j=1}^{J} \sum_{n=1}^{T} (D_{jn}{}^{i} + N_j * t_j) \right] / T \tag{31}$$

where

N_j is the number of setups required which is determined by the Economic Order Quantity (EOQ) of product j, i.e.,

$$N_j = \left\lceil \frac{D_{jn}}{\sqrt{\frac{2 * D_{jn} * S_j}{H_j}}} \right\rceil \tag{32}$$

Then, this quantity is multiplied by 0.8, 1.0 and 1.2 for problems with 80%, 100% and 120%, which represents the tightness of capacity, respectively.

The quantity of overtime capacity available in period m with seasonality of type i is given by $C_{2m}^{i} = 0.5 * C_{1m}^{i}$.

- Capacity costs

The unit costs for using regular time and overtime capacity are respectively 40.0 and 60.0 for all periods in all problems.

- Inventory holding costs

The inventory costs for carrying a unit of product 1,2,3 for one period are respectively 4.0, 7.0 and 6.0.

- Setup costs

Setup cost for product j is given by:

$$S_j = \sum_{p=1}^{P_j} t_p \tag{33}$$

where

t_p is the p^{th} random draw from a uniform distribution over the range $[u_{sp}, l_{sp}]$,

u_{sp}, l_{sp} are upper and lower limits of t_p respectively,

P_j is equal to 5, 5 and 10 for product j=1, 2 and 3 respectively.

The ranges of setup costs for products are shown in Table 3. High, medium and low setup cost for product j are computed by $2*S_j$, $1*S_j$ and $0.5*S_j$ respectively.

- Setup times

Low, medium and high level of setup time of a product is an integer selected from the range [10,20], [20,35] and [35,50] respectively. The level of setup time varies with the level of setup cost.

- Absorption rate

The capacity absorption rates of all products are simply set to 1 (i.e., $b_j = 1$ for $j = 1,\ldots,J$).

- Generation of larger problems

The same problem generating algorithm is applied when there exists more than 3 products or 12 periods. For example, a 6-product problem is treated as two 3-product problems, while a 9-product problem is treated as three 3-product problems. Similarly, an (n*12)-period problem is treated as n 12-period problems.

5.2 Discussion of lower bound procedures

Two procedures for computing a lower bound for the problem *P1* is described here, which are presented by Gilbert and Madan (1991), to evaluate the effectiveness of our algorithm.

- Procedure I for computing a lower bound

The objective function (1) of problem *P1* is to minimize the sum of holding, setup and production costs. A lower bound on total cost for *P1* is computed by solving two separate relaxations of the problem, one problem with setup and holding costs only and another one with production costs only. Therefore, the value of the lower bound, LB1, can be obtained by solving one relaxation of *P1* with only setup and holding costs in the objective function, and another one with only production costs, and summing the two objective function values.

The first relaxation problem could be regarded as several single-product uncapacitated lot-sizing problems, and solved by the Wagner-Whitin Algorithm. We sum the individual objective values for each product and get the first part of the lower bound.

The second relaxation problem is a standard transportation problem. Hence, the objective function value is easy to be obtained.

The summation of these two parts above gives a lower bound on the objective of *P1*. We call it Lower Bound 1 (LB1) in the following analysis.

- Procedure II for computing a lower bound

Another approach is also based on solving two separated relaxations of the original problem *P1*. The difference is that, the first part is a problem with the fixed cost only, i.e., the setup cost, and the second part is a problem with the variable cost only, i.e., the holding cost and the production cost.

For the first part, the lower bound on the total setup cost is found by determining a minimum number of setups required for feasibility. For each product, the number of setups required is the smallest integer that is greater than the quotient of the total demand for this particular product and the capacity available per period. Thus, the lower bound of setup costs equals:

$$\sum_{j=1}^{J}((\min_{1\le m\le T}\left\lceil\frac{\sum_{n=1}^{T}D_{jn}}{\sum_{k=1}^{K}C_{km}}\right\rceil)*S_j) \tag{34}$$

For the second part, we force the setup cost to be zero in *P1* and consider the problem as a traditional transportation problem.

The summation of these two parts above gives a lower bound on the objective of *P1*. We call it Lower Bound 2 (LB2) in the following analysis.

5.3 Computational results

We consider two cases here: (1) the CLS problems with setup time consideration; (2) the CLS problems without setup time consideration. The problems with setup times are exactly our model as described previously, and we just take the problems without setup times as a special case in which setup times are all zero. The results of test problems for the first case will be compared with the lower bounds of each problem respectively. And the results of test problems for the second case will be compared with those in So's research thesis (So, 1997).

5.3.1 CLS problems with setup times

We classified the test problems into 27 (3^3) combinations according to the given three parameters. And for each combination, we run 10 different problems with different information data generated by the scheme discussed above and compute the average solution gap with both Lower Bound 1 and Lower Bound 2. The quality of the solution is evaluated by computing a percentage difference as following:

$$\frac{OFV - LB}{LB} * 100\% \tag{35}$$

where OFV is the Objective Function Value of our heuristic algorithm, and LB is the Lower Bound value.

The results of the 270 problems are summarized in Table 4-1 and Table 4-2. Tables 5-1, 5-2, Tables 6-1, 6-2, and Tables 7-1, 7-2 illustrate the effect of the three parameters on the value of the Average Percentage Difference (A. P. D.) with two Lower Bounds respectively. Table 5-1 and Table 5-2 show the A.P.D. between the heuristic solution value and the Lower Bounds for different levels of setup for all problems. Table 6-1 and Table 6-2 are for different setup levels for all problems across different levels of seasonality. And Table 7-1 and Table 7-2 are for different setup levels for all problems across different tightness of capacity.

The average A.P.D. across all 270 problems is 1.66%, ranging from 0.00% to 7.16%, when compared with LB1. In about 70% of the cases, the A. P. D. is less than 1.79%. When compared with LB2, the average A.P.D. across all 270 problems is 7.02%, ranging from 3.01% to 12.21%. And in about 70% of the cases, the A. P. D. is less than 9.54%.

We discuss the performance of our algorithm with respect to three parameters: (1) level of setup; (2) seasonality of demand; and (3) tightness of capacity. When compared with LB1, we find the most critical factors are the setup level and tightness of capacity parameter, while change in seasonality of demand has less apparent change in the average solution gap, except for the case with extreme seasonality (see Table 6-1). When compared with LB2, similar results are found. The setup level

affects most on A. P. D., while change in seasonality of demand and tightness of capacity bring minor change in the average solution gap, except for the case with tight capacity (i.e., 80% capacity) (see Table 7-2).

We also analyse the simultaneous effect of the combinations of two parameters of the three. The results are shown in Tables 8-1, 8-2, Tables 9-1, 9-2, and Tables 10-1, 10-2.

Table 8-1 and Table 8-2 show the simultaneous effect of seasonality and setup levels. When compared with both lower bounds, for low and medium setup cases, the heuristics performs better than high cases. Among all levels of setups, the A.P.D. for none seasonality cases is the lowest, with the value of 0.31% when compared with LB1 (see Table 8-1). While the lowest A. P. D. falls into extreme seasonality case among all setup levels, with the value of 3.05%, when compared with LB2 (see Table 8-2).

Table 9-1 and Table 9-2 present the simultaneous effect of seasonality and capacity tightness. For 100% capacity and 120% capacity cases, the heuristics performs better than tight capacity case (i.e., 80% capacity), when compared with LB1. While compared with LB2, no apparent difference exists among all capacities. The lowest A. P. D. appears in moderate seasonality across all tightness of capacity, with the value of 0.22%, when compared with LB1 (see Table 9-1). Refer to the comparison with LB2, the A. P. D. for extreme seasonality case among all tightness of capacity is the lowest, with the value of 5.96%, when compared with LB2 (see Table 9-2).

Table 10-1 and Table 10-2 summarize the simultaneous effect of setup levels and capacity tightness. Our heuristic performs better in the case of 100% and 120%

capacity than 80% capacity, and the case of low and medium setup level than high level. The lowest figures of A. P. D are both in the case with low setup level among different tightness of capacity, valuing 0.12% and 3.12%, respectively, when compared with two Bounds.

We also generate larger problems (i.e., we double the number of products, the number of capacities, or the planning periods) to further test the performance of our algorithm. The computational time increases by about 2 or 3 times on average, though still reasonable for realistically sized problems. The similar trend appears that, in general, our heuristic performs better in looser capacity scenario and lower setup level one.

Seasonality	Setup	Capacity	1	2	3	4	5	6	7	8	9	10	Average solution gap(%)
No	Low	80%	0.0180	0.0192	0.0164	0.0185	0.0162	0.0152	0.0213	0.0178	0.0175	0.0155	1.75
No	Low	100%	0.0000	0.0051	0.0000	0.0024	0.0000	0.0000	0.0000	0.0000	0.0064	0.0054	0.19
No	Low	120%	0.0000	0.0040	0.0042	0.0000	0.0000	0.0000	0.0040	0.0000	0.0043	0.0000	0.16
No	Medium	80%	0.0049	0.0134	0.0099	0.0085	0.0006	0.0108	0.0190	0.0046	0.0144	0.0040	0.90
No	Medium	100%	0.0000	0.0000	0.0027	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.03
No	Medium	120%	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.00
No	High	80%	0.0267	0.0169	0.0367	0.0203	0.0141	0.0369	0.0153	0.0258	0.0195	0.0260	2.38
No	High	100%	0.0131	0.0104	0.0090	0.0176	0.0121	0.0148	0.0117	0.0133	0.0125	0.0126	1.27
No	High	120%	0.0075	0.0086	0.0055	0.0051	0.0058	0.0055	0.0099	0.0114	0.0049	0.0126	0.77
Moderate	Low	80%	0.0239	0.0240	0.0199	0.0155	0.0190	0.0128	0.0159	0.0162	0.0136	0.0183	1.79
Moderate	Low	100%	0.0006	0.0000	0.0000	0.0003	0.0004	0.0000	0.0003	0.0000	0.0000	0.0001	0.02
Moderate	Low	120%	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.00
Moderate	Medium	80%	0.0263	0.0103	0.0144	0.0106	0.0174	0.0263	0.0301	0.0268	0.0240	0.0235	2.10
Moderate	Medium	100%	0.0008	0.0010	0.0000	0.0006	0.0004	0.0012	0.0013	0.0001	0.0014	0.0007	0.07
Moderate	Medium	120%	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.00
Moderate	High	80%	0.0359	0.0398	0.0336	0.0253	0.0454	0.0281	0.0259	0.0339	0.0351	0.0352	3.38
Moderate	High	100%	0.0143	0.0155	0.0137	0.0129	0.0100	0.0133	0.0127	0.0103	0.0164	0.0144	1.34
Moderate	High	120%	0.0088	0.0050	0.0066	0.0063	0.0085	0.0066	0.0070	0.0077	0.0090	0.0000	0.65
Extreme	Low	80%	0.0756	0.0631	0.0688	0.0762	0.0732	0.0714	0.0612	0.0824	0.0658	0.0634	7.01
Extreme	Low	100%	0.0340	0.0252	0.0270	0.0256	0.0280	0.0195	0.0244	0.0193	0.0232	0.0253	2.52
Extreme	Low	120%	0.0019	0.0000	0.0017	0.0025	0.0037	0.0036	0.0019	0.0022	0.0017	0.0017	0.21
Extreme	Medium	80%	0.0681	0.0731	0.0646	0.0735	0.0744	0.0688	0.0708	0.0786	0.0803	0.0639	7.16
Extreme	Medium	100%	0.0111	0.0143	0.0197	0.0108	0.0159	0.0123	0.0135	0.0183	0.0144	0.0081	1.38
Extreme	Medium	120%	0.0009	0.0011	0.0021	0.0033	0.0015	0.0010	0.0007	0.0025	0.0124	0.0024	0.28
Extreme	High	80%	0.0592	0.0618	0.0665	0.0686	0.0654	0.0823	0.0795	0.0700	0.0638	0.0662	6.79
Extreme	High	100%	0.0208	0.0214	0.0267	0.0197	0.0216	0.0178	0.0176	0.0300	0.0167	0.0242	2.17
Extreme	High	120%	0.0052	0.0018	0.0073	0.0074	0.0097	0.0031	0.0062	0.0033	0.0025	0.0011	0.48

The average solution gap is the average value of solution gaps in 10 sets of problems for each case.

Table 4-1: Performance of the heuristic algorithm in 270 problem cases compared with LB1

Seasonality	Setup	Capacity	1	2	3	4	5	6	7	8	9	10	Average solution gap(%)
No	Low	80%	0.0471	0.0474	0.0458	0.0473	0.0458	0.0407	0.0508	0.0470	0.0442	0.0423	4.58
No	Low	100%	0.0268	0.0391	0.0279	0.0337	0.0321	0.0302	0.0296	0.0329	0.0382	0.0362	3.27
No	Low	120%	0.3291	0.0381	0.0359	0.0306	0.0265	0.0320	0.0349	0.0293	0.0375	0.0322	3.30
No	Medium	80%	0.0639	0.0752	0.0674	0.0636	0.0480	0.0687	0.0754	0.0573	0.0733	0.0616	6.54
No	Medium	100%	0.0652	0.0673	0.0656	0.0607	0.0526	0.0634	0.0612	0.0582	0.0655	0.0639	6.24
No	Medium	120%	0.0652	0.0673	0.0627	0.0607	0.0526	0.0634	0.0612	0.0582	0.0688	0.0593	6.19
No	High	80%	0.1242	0.1117	0.1376	0.1180	0.1116	0.1446	0.1052	0.1230	0.1138	0.1261	12.16
No	High	100%	0.1220	0.1188	0.1176	0.1292	0.1201	0.1252	0.1196	0.1245	0.1174	0.1262	12.21
No	High	120%	0.1194	0.1162	0.1166	0.1183	0.1135	0.1154	0.1214	0.1208	0.1165	0.1246	11.83
Moderate	Low	80%	0.0401	0.0380	0.0401	0.0414	0.0395	0.0361	0.0385	0.0404	0.0393	0.0428	3.96
Moderate	Low	100%	0.0284	0.0282	0.0303	0.0315	0.0307	0.0293	0.0288	0.0317	0.0314	0.0282	2.98
Moderate	Low	120%	0.0333	0.0326	0.0327	0.0308	0.0304	0.0335	0.0313	0.0299	0.0337	0.0309	3.19
Moderate	Medium	80%	0.0761	0.0608	0.0671	0.0609	0.0710	0.0693	0.0785	0.0751	0.0684	0.0727	7.00
Moderate	Medium	100%	0.0657	0.0606	0.0624	0.0624	0.0615	0.0654	0.0608	0.0601	0.0639	0.0649	6.28
Moderate	Medium	120%	0.0621	0.0607	0.0605	0.0606	0.0618	0.0624	0.0623	0.0640	0.0603	0.0631	6.18
Moderate	High	80%	0.1342	0.1363	0.1328	0.1195	0.1498	0.1321	0.1215	0.1269	0.1336	0.1333	13.20
Moderate	High	100%	0.1211	0.1254	0.1245	0.1208	0.1129	0.1187	0.1239	0.1118	0.1236	0.1180	12.01
Moderate	High	120%	0.1221	0.1102	0.1115	0.1154	0.1127	0.1114	0.1134	0.1095	0.1178	0.1045	11.28
Extreme	Low	80%	0.0296	0.0281	0.0310	0.0290	0.0316	0.0314	0.0291	0.0306	0.0303	0.0301	3.01
Extreme	Low	100%	0.0304	0.0310	0.0288	0.0324	0.0305	0.0284	0.0472	0.0306	0.0324	0.0342	3.26
Extreme	Low	120%	0.0277	0.0290	0.0277	0.0279	0.0288	0.0292	0.0286	0.0296	0.0295	0.0285	2.87
Extreme	Medium	80%	0.0579	0.0544	0.0593	0.0610	0.0585	0.0605	0.0572	0.0574	0.0633	0.0591	5.89
Extreme	Medium	100%	0.0554	0.0542	0.0637	0.0577	0.0603	0.0561	0.0616	0.0586	0.0553	0.0559	5.79
Extreme	Medium	120%	0.0511	0.0546	0.0544	0.0546	0.0535	0.0528	0.0535	0.0531	0.0669	0.0537	5.48
Extreme	High	80%	0.1050	0.0922	0.0999	0.1081	0.1043	0.1081	0.1074	0.1043	0.1051	0.1122	10.47
Extreme	High	100%	0.1045	0.1100	0.1006	0.0993	0.1128	0.1039	0.1017	0.1153	0.1066	0.1182	10.83
Extreme	High	120%	0.0959	0.0906	0.1016	0.0993	0.0994	0.0946	0.0962	0.0951	0.0923	0.0891	9.54

Table 4-2: Performance of the heuristic algorithm in 270 problem cases compared with LB2

	Setup Costs and Setup Times		
	Low	*Medium*	*High*
A.D.P.	1.52	1.32	2.14

Table 5-1

The effect of setup levels on A.P.D. compared with LB1

	Setup Costs and Setup Times		
	Low	*Medium*	*High*
A.D.P.	3.38	6.18	11.50

Table 5-2

The effect of setup levels on A.P.D. compared with LB2

	Seasonality		
	None	*Moderate*	*Extreme*
A.D.P.	0.83	1.04	3.11

Table 6-1

The effect of seasonality on A.P.D. compared LB1

	Seasonality		
	Low	*Moderate*	*Extreme*
A.D.P.	7.37	7.34	6.35

Table 6-2

The effect of seasonality on A.P.D. compared with LB2

	Capacity		
	80%	*100%*	*120%*
A.D.P.	3.70	1.00	0.28

Table 7-1

The effect of capacity tightness on A.P.D. compared with LB1

	Capacity		
	80%	*100%*	*120%*
A.D.P.	7.42	6.99	6.65

Table 7-2

The effect of capacity tightness on A.P.D. compared with LB2

Setup	Seasonality		
	None	*Moderate*	*Extreme*
Low	0.70	0.60	3.25
Medium	0.31	0.72	2.94
High	1.47	1.79	3.15

Table 8-1

The effect of seasonality and setup levels on A.P.D. compared with LB1

Setup	Seasonality		
	None	*Moderate*	*Extreme*
Low	3.72	3.38	3.05
Medium	6.32	6.49	5.72
High	12.07	12.16	10.28

Table 8-2

The effect of seasonality and setup levels on A.P.D. compared with LB2

Capacity	Seasonality		
	None	*Moderate*	*Extreme*
80%	1.68	2.42	6.99
100%	0.50	0.48	2.02
120%	0.31	0.22	0.32

Table 9-1

The effect of seasonality and capacity tightness on A.P.D. compared with LB1

Capacity	Seasonality		
	None	*Moderate*	*Extreme*
80%	7.76	8.05	6.46
100%	7.24	7.09	6.63
120%	7.11	6.88	5.96

Table 9-2

The effect of seasonality and capacity tightness on A.P.D. compared with LB2

Capacity	Setup		
	Low	*Medium*	*High*
80%	3.52	3.39	4.18
100%	0.91	0.49	1.59
120%	0.12	0.09	0.63

Table 10-1

The effect of setup levels and capacity tightness on A.P.D. compared with LB1

Capacity	Setup		
	Low	*Medium*	*High*
80%	3.85	6.48	11.94
100%	3.17	6.10	11.68
120%	3.12	5.95	10.88

Table 10-2

The effect of setup levels and capacity tightness on A.P.D. compared with LB2

5.3.2 CLS problems without setup times

We force all setup times to be zero and relax the problem to a CLS problem without setup time consideration. To analyse the performance of our heuristic, the solution value of our algorithm is compared with that of So's (So, (1997)), who discussed a CLS problem without setup times.

The computation of Difference Percentage is as follows:

$$\frac{OFV_1 - OFV_2}{OFV_2} * 100\% \tag{36}$$

where OFV_1 is the Objective Function Value of So's heuristic algorithm, and OFV_2 is the Objective Function Value of our heuristic algorithm.

The same analysis method is used to evaluate the results for the case without setup times. We totally generated 27 different combinations of the cases according to the three parameters, and for each case, 10 problems were generated randomly to obtain the average percentage difference.

The performance of our heuristic is evaluated by comparing the objective function value of both our algorithm and that of So's. The results of the comparison are concluded in Table 11. On average, we improved the objective function value by 0.29% better than So's algorithm. As we can see from the table, our heuristic performs better when seasonality of demand is extreme or setup level is high.

Further, we compare the performance of our heuristic with So's algorithm under different problem characteristics. The results are shown in Table 12 to 14. In each comparison, Average Percentage Difference (A. P. D.) is calculated.

Table 12 presents the difference between two algorithms on the effect of setup level on A. P. D.. We can find that our algorithm takes more advantage as setup level turns higher from the low case.

Table 13 summaries the difference between two algorithms on the effect of seasonality on A. P. D.. When the seasonality becomes extreme, our algorithm performs better than in the case with none seasonality.

Table 14 is the difference between two algorithms on the effect of capacity tightness on A. P. D.. Our heuristic is better on average, although the advantage is minor. No apparent changes appear when the tightness varies from 80% to 120%.

In addition, the interactions among three parameters are analyzed in the following tables from Table 15 to 17. Our algorithm performs better than So's, especially in the case with high setup level and loose capacity (i.e., 120% capacity), and the case with extreme seasonality and loose capacity as well.

Seasonality	Setup	Capacity	Average Percentage Difference (%)
No	Low	80%	0.00
No	Low	100%	0.00
No	Low	120%	0.00
No	Medium	80%	0.00
No	Medium	100%	0.00
No	Medium	120%	0.00
No	High	80%	0.00
No	High	100%	0.31
No	High	120%	0.95
Moderate	Low	80%	0.01
Moderate	Low	100%	0.00
Moderate	Low	120%	0.00
Moderate	Medium	80%	0.01
Moderate	Medium	100%	0.06
Moderate	Medium	120%	0.06
Moderate	High	80%	0.06
Moderate	High	100%	0.38
Moderate	High	120%	1.07
Extreme	Low	80%	0.01
Extreme	Low	100%	0.01
Extreme	Low	120%	0.00
Extreme	Medium	80%	1.98
Extreme	Medium	100%	0.11
Extreme	Medium	120%	0.24
Extreme	High	80%	0.57
Extreme	High	100%	0.83
Extreme	High	120%	1.22

Table 11: Average difference percentage of the 27 problem cases between our heuristic and So's heuristic

	Setup Costs and Setup Times		
	Low	*Medium*	*High*
A.D.P.	0.00	0.27	0.60

Table 12: The effect of setup levels on A.P.D.

	Seasonality		
	None	*Moderate*	*Extreme*
A.D.P.	0.14	0.18	0.55

Table 13: The effect of seasonality on A.P.D.

	Capacity		
	80%	*100%*	*120%*
A.D.P.	0.29	0.19	0.39

Table 14: The effect of capacity tightness on A.P.D.

Setup	Seasonality		
	None	*Moderate*	*Extreme*
Low	0.00	0.00	0.01
Medium	0.00	0.04	0.78
High	0.42	0.50	0.87

Table 15: The effect of seasonality and setup levels on A.P.D.

Capacity	Seasonality		
	None	*Moderate*	*Extreme*
80%	0.00	0.03	0.85
100%	0.10	0.15	0.32
120%	0.32	0.38	0.49

Table 16: The effect of seasonality and capacity tightness on A.P.D.

Capacity	Setup		
	Low	*Medium*	*High*
80%	0.01	0.66	0.21
100%	0.00	0.06	0.51
120%	0.00	0.10	1.08

Table 17: The effect of setup levels and capacity tightness on A.P.D.

5.4 Conclusion

In this chapter, we designed a 3-product, 2-capacity, 12-period-planning-horizon problem for a variety of test problems. All test problems considered three parameters: seasonality of demand, the tightness of capacity, and the setup cost and setup time level. Two Lower Bounds are computed for comparison. Our heuristic algorithm performed quite well when compared with Lower Bound 1. Also, we forced all setup times to zero to relax the problem to a CLS problem without setup time. The computational results for this case are also good and reliable, compared with similar problems discussed by So in his research thesis (So, 1997).

Chapter 6

Conclusion

The Capacitated Lot Sizing (CLS) Problem is concerned with the planning of production that determines the setup decisions and production quantities for multiple products over a finite number of periods without violating capacity constraints. In this research, we develop a model for the CLS problem with the setup time consideration. The model is extended to consider major CLS decisions such as setups, production levels, inventory levels, and overtimes. Its objective is to satisfy the known demands for various products in each period with capacitated resources and to minimize the sum of production, setup and inventory costs without incurring backorders. This CLS problem can be found in many repetitive manufacturing settings. Processes like assembly and stamping are some practical examples.

The CLS problems attract a lot of interest in the literature. But a few researchers explored the area of CLS problem with setup times, especially for CLS problems with the formulation structure that we use here. Many of them believed that CLS problem with setup times was only a simple extension of the one without setup times. We showed the difficulty of solving CLS problem with setup times using Trigerio et al.'s (1989) example. Hence, it is worth further investigating CLS problem with setup times especially in manufacturing industries with significant setup times.

We use a new model that considers setup times and allows for different types of production capacities such as regular time, overtime and subcontracting. So, our model is more realistic and comprehensive. We apply a heuristic approach based on Genetic Algorithms to solve the CLS problem with setup times. We use a 3-product, 2-capacity, 12-planning-period problem to test the performance of our heuristic. Larger-sized problems are generated as well for further investigation. The computational results show that our algorithm gives reliable results within the average percentage difference of 4.34%, ranging from 0.00% to 12.21%, by comparing with two lower bounds generated. Except for those problems having very high setup level or problems with very tight capacities, our heuristic produces solutions quite efficiently. Moreover, our algorithm improved the results of So's (1997) algorithm by 0.29% on average, when the original CLS problem is relaxed to one without setup time consideration.

Due to the nature of the problem, it is very difficult to solve the CLS problem with setup times optimally. Heuristic methods are applied to find reasonable solutions. However, these methods cannot perform well especially in cases of high setup cost and time, and very tight capacity. Besides, in our specifications of test problems, the values of setup cost and setup time are directly proportional to the level of setup. If the level of setup is high, the amount of setup time will be large and it will consume a significant portion of the production capacity available. Hence, capacity allocated for production will be reduced after setup, and it affects the efficiency of production.

One extension of our model is to consider the case when backorder is permitted. A new information data, the backorder cost, may be introduced, and the whole model would be modified. One suggestion to solve this kind of problem is to follow Cheng

et al's (2001) work, which is a CLS problem with backorder consideration and without setup times. A proper integration of the main idea to solve their model and that in our model might be a feasible way to deal with the CLS problems with both setup time and backorder consideration.

We could also have some further discussion about the procedure to generate the setup strings. In this thesis, we let 50% of the strings generated by Wagner-Whitin Algorithm or by making minor changes to the strings using Wagner-Whitin Algorithm. Other algorithms could also be tried to get a setup string, like Silver-Meal Heuristic (1969), for instance.

Another development of this paper work is to find a better approach to generate Lower Bounds. According to the computational results, it is obviously that the results compared with Lower Bound 1 are much better than those with Lower Bound 2. The improvement of the procedure to generate a better Lower Bound is also a big challenge for validating a heuristic algorithm.

Bibliography

[1] Axsater, S., *I*nventory control, Boston/Dordrecht/London: Kluwer Academic Publishers, 2000.

[2] Bahl, H.C., Ritzman, L.P. and Gupta, J.N.D., "Determining Lot Sizes and Resource Requirements: A Review", Operations Research, 35, 329-345, 1987.

[3] Balinski, M.L., "Fixed Cost Transportation Problem", Naval Research Logistics Quarterly, 8(1), 41-54, 1961.

[4] Barany, I., Van Roy, T.J. and Wosley, L.A., "Strong Formulations for Multi-item Capacitated Lot Sizing", Management Science, 30(10), 1255-1261, October 1984.

[5] Cheng C.H., Madan M.S., Gupta Y.P., and So S., "Solving the capacitated lot-sizing problem with backorder consideration", Journal of Operational Research Society, Vol. 52, No. 8, pp. 952-959, 2001.

[6] Chuda Basnet and Janny M. Y. Leung, "Inventory lot-sizing with supplier selection", Computers & Operations Reseach, 32, 1-14, 2005.

[7] Cooper, L. and Drebes, C., "An Approximate Solution for the Fixed Charge Problem", Naval Research Logistics Quarterly, 14(1), 101-113, 1967.

[8] Daniel Quadt and Heinrich Kuhn, "Capacitated lot-sizing with extensions: a review", 4OR: Q J Oper Res., 6, 61-83. 2008.

[9] Dantzig, G.B. and Wolfe, P., "A Decomposition Algorithm for Linear Programs", Econometrica, 29, 767-778, 1961.

[10] Denzler, D.R., "An Approximate Algorithm for the Fixed Charge Problem", Naval Research Logistics Quarterly, 16(3), 411-416, 1969.

[11] 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.

[12] Diaby, M., Bahl, H.C., Karwan, M.H. and Zionts, S., "Capacitated Lot-sizing and Scheduling by Lagrangean Relaxation", European Journal of Operational Research, 59, 444-458, 1992.

[13] Dixon, P. and Silver, E., "A Heuristic Solution Procedure for the Multi-item, Single-levels, Limited Capacity, Lot-sizing Problem", Journal of Operations Management, 2, 23-29, 1981.

[14] Drexl, A. and Kimms, A., "Invited Review: Lot Sizing and Scheduling – Survey and Extensions", European Journal of Operational Research, 99, 221-235, 1997.

[15] Dzielinski, B.P., C. T. Baker AND A. S. Manne. "Simulation Tests of Lot Size Programming". Management Science, 9, 229-258, 1963.

[16] Dzielinski, B.P. and Gomory, R.E., "Optimum Programming of Lot Sizes Inventories and Labor Allocations", Management Science, 11, 9, 874-890, July 1965.

[17] Eisenhut, P.S., "A Dynamic Lot Sizing Algorithm with Capacity Constraints", AIIE Transactions, 7, 2, 170-176, 1975.

[18] Emmeche C., Garden in the Machine. The Emerging Science of Artificial Life, Princeton University Press, pp.,114 ss,1994.

[19] Eppen, G. and Martin, R., "Solving Multi-item Capacitated Lot Sizing using Variable Redefinition", Operations Research, 35, 832-848, 1987.

[20] Gilbert, K.C., and Madan, M.S., "A Heuristic for a Class of Production Planning and Scheduling Problems", AIIE Transactions, 23(3), 282-289, September 1991.

[21] Goldberg D., Genetic Algorithms, Addison Wesley, 1988.

[22] Gopalakrishnan, M., Ding, K., Bourjolly, J.M., and Mohan, S., "A Tabu-Search Heuristic for the Capacitated Lot-Sizing Problem with Set-up Carryover", Management Science, 47(6), 851–863, 2001.

[23] Graves, S.C., "Using Lagrangean Techniques to Solve Hierarchical Production Planning Problems", Management Science, 28(3), 260-275, March 1982.

[24] Grefenstette J. J., "A User's Guide to GENESIS Version 5.0", Technical Report, Navy Centre for Applied Research in Artificial Intelligence, Washington D.C., USA, 1990.

[25] Hernandez W. and Suer G. A., "Genetic Algorithms in Lot Sizing Decisions", Proceedings of the Congress on Evolutionary Computation CEC99, pp. 2280-2286, 1999.

[26] Hindi, K.S., "Computationally Efficient Solution of the Multi-item, Capacitated Lot-sizing Problem", Computers & Industrial Engineering, 28, 709-719, 1995.

[27] Hirsch, W.M. and Dantzig, G.B., "The Fixed Charge Problem", Naval Research Logistics Quarterly, 15(3), 413-424, 1968.

[28] Holland J.H., "Adaptation in natural and artificial system", Ann Arbor, The University of Michigan Press, 1975.

[29] Hyun C J, Kim Y & Kim Y K, "A genetic algorithm for multiple objective sequencing problems in mixed model assembly lines", Computers and Operations Research, vol. 25, No. 7/8, pp. 675-690, 1998.

[30] Ip W H, Li Y, Man K F & Tang K S, "Multi-product planning and scheduling using genetic algorithm approach", Computers & Industrial Engineering, 38, pp. 283-296, 2000.

[31] Janny M. Y. Leung, Thomas L. Magnanti, Rita Vachani, "Faces and Algorithms for Capacitated Lot Sizing", Mathematical Programming, 45, 331-359, 1989.

[32] Karimi, B., Fatemi Ghomi, S.M.T., Wilson, J.M., "The Capacitated Lot Sizing Problem: a Review of Models and Algorithms", Omega, 31(5), 365-378, 2003.

[33] Kennington, J., "The Fixed-charge Transportation Problem: A Computational Study with a Branch-and-Bound Code", AIIE Transactions, 8(2), 241-247, 1976.

[34] Kennington, J. and Unger, E., "A New Branch-and-Bound Algorithm for the Fixed-charge Transportation Problem, Management Science, 22(10), 1116-1126, 1976.

[35] Khouja M, Michalewicz Z & Wilmot M, "The use of genetic algorithms to solve the economic lot size scheduling problem", European Journal of Operational Research, 110, pp. 509-524, 1998.

[36] Kim J & Kim Y, "Simulated Annealing and Genetic Algorithms for Scheduling Products with Multi-level Product Structure", Computers Operations Research, Vol. 23, No. 9, pp 857-868, 1996.

[37] Kimms A, "A genetic algorithm for multi-level, multi-machine lot sizing and scheduling", Computers & Operations Research, 26, pp. 829-848, 1999.

[38] Lambrecht, M.R. and Vanderveken, H., " Heuristic Procedures for the Single Operation, Multi-item Loading Problem", AIIE Transactions, 11, 4, 319-326, December 1979.

[39] Lasdon, L.S. and Terjung, R.C., "An Efficient Algorithm for Multi-item Scheduling", Operations Research, 19(4), 946-969, 1971.

[40] Lozano, S., Larraneta, J. and Onieva L., "Primal-dual Approach to the Single Level Capacitated Lot-sizing Problem", European Journal of Operational Research, 51, 354-366, 1991.

[41] Madan, M.S., "A Heuristic and an Exact Solution Algorithm for a Class of Production Planning and Scheduling Problem", Unpublished Ph.D. Dissertation, University of Tennessee, 1988.

[42] Madan, M.S., and Gilbert, K.C., "An Exact Solution Algorithm for a Class of Production Planning and Scheduling Problems", Journal of the Operational Research Society, 43, 10, 961-970, 1992.

[43] Maes, J., and Van Wassenhove, L., "Multi-Item Single-level Dynamic Lot-sizing Heuristics", Journal of Operational Research Society, 39, 991-1004, 1988.

[44] Manne, A.S., "Programming of Economic Lot Sizes", Management Science, 4(2), 115-135, 1958.

[45] Melanie Mitchell, "An introduction to genetic algorithms", Cambridge, Mass.: MIT Press, 1998

[46] Millar, H.H., and Yang, M., "An Application of Lagrangean Decomposition to the Capacitated Multi-item Lot-sizing Problem", Computers and Operations Research, 20(4), 409-420, 1993.

[47] Millar, H.H., and Yang, M., "Lagrangean Heuristics for the Capacitated Multi-item Lot-sizing Problem with Backordering", International Journal of Production Economics, 34, 1-15, 1994.

[48] Nadjib Brahimi, Stéphane Dauzère-Pérès, Najib M. Najid, and Atle Nordli, "Invited review: Single item lot sizing problems", European Journal of Operational Research, 168, 1-16, 2004.

[49] Nadjib Brahimi, Stéphane Dauzère-Pérès and Najib M. Najid, "Capacitated Multi-Item Lot-Sizing Problems with Time Windows", OPERATIONS RESEARCH, Vol. 54, No. 5, pp. 951–967, September–October 2006.

[50] Ozdanmar L., Bozyel MA, "The capacitated lot-sizing problem with overtime decisions and setup times", IIE Transactions, 32, 1043-1057, 2000.

[51] Pfeiffer T., "Transfer pricing and decentralized dynamic lot-sizing in multi-stage, multi-product production processes", European Journal of Operational Research, 116, 319-330, 1999.

[52] Pochet, Y. and Wolsey, L., "Lot Size Models with Backlogging: Strong Reformulations and Cutting Planes", Mathematical Programming, 40, 317-335, 1988.

[53] Pochet, Y. and Wolsey, L., "Solving Multi-item Lot-sizing Problems Using Strong Cutting Planes", Management Science, 37(1), 53-67, January 1991.

[54] Potts C N & Van Wassenhove L N, "Integrating Scheduling with Batching and Lot-Sizing: A Review of Algorithms and Complexity", Operational Research Society, Vol. 43, no. 5, pp, 395-406, 1992.

[55] Raf Jans, Zeger Degraeve, Meta-heuristics for dynamic lot sizing: A review and comparison of solution approaches, European Journal of Operational Research, 177, 1855–1875, 2007.

[56] Raf Jans and Zeger Degraeve, "Modeling industrial lot sizing problems: A review," International Journal of Production Research, v46, pp. 1619-1643, 2008.

[57] Reinfeld, N.V. and Vogel, W.R., Mathematical Programming, Prentice-Hall, Englewood Cliffs, N.J., 1958.

[58] Salomon, M., Solomon, M.M., Wassenhove, L.N.V., Dumas, Y. and Dauzère-Pérès, S., "Solving the discrete lotsizing and scheduling problem with sequence dependent set-up costs and set-up times using the Travelling

Salesman Problem with time windows", European Journal of Operational Research, 100, 494-513, 1997.

[59] Sambasivan M, Schmidt CP., "A heuristic procedure for solving multi-plant multi-item multi-period capacitated lot-sizing problems", Asia-Pacific Journal of Operational Research, 19, 87-105, 2002.

[60] Shaw DX, Wagelmans LPM, "An algorithm for single-item capacitated economic lot sizing with piecewise linear production costs and general holding costs", Management Science, 44, 831-838, 1998.

[61] Sikora R, "A genetic algorithm for integrating lot-sizing and sequencing in scheduling a capacitated flow line", Computers Industrial Engineering, Vol. 30, No. 4, pp. 969-981, 1996.

[62] Silver E.A. and Meal H.C., "A simple modification of the EOQ for the case of a varying demand rate", Production and Inventory Management, 10, 52-65, 1969.

[63] Silver E.A. and Meal H.C., "A heuristic for selecting lot size quantities for the case of a deterministic time varying rate and discrete opportunities for replenishment", Production and Inventory Management, 14, 64-74, 1973.

[64] So Wai Kuen, "Optimization-based algorithms for a single level constrained resource problem ", MPhil thesis, The Chinese University of Hong Kong, 1997.

[65] Sox CR and Gao Y., "The capacitated lot-sizing problem with setup carry-over", IIE Transactions, 31, 173-181, 1999.

[66] Staggemeier A.T., Clark A.R., "A survey of lot-sizing and scheduling models", 23rd Annual Symposium of the Brazilian Operational Research Society (SOBRAPO), Campos do Jordao SP, Brazil, pp. 938–947, 2001.

[67] Steinberg, D.I., "The Fixed Charge Problem", Naval Research Logistics Quarterly, 17(2), 217-235, 1970.

[68] Thizy, J.M. and Van Wassenhove, L.N., "A Subgradient Algorithm for the Multi-item Capacitated Lot Sizing Problem", AIIE Transactions, 17(4), 308-313, December 1985.

[69] Trigerio, W.W., "A Simple Heuristic for Lot Sizing with Setup Times", Decision Sciences, 20, 294-303, 1989.

[70] Trigerio, W.W., Thomas, L.J. and McClain, J.O., "Capacitated Lot Sizing with Setup Times", Management Science, 35(3), 353-366, 1989.

[71] Trotter L. E. and Shetty C. M., "An algorithm for the bounded variable integer programming problem", J. Assoc. Comp., 21, 505-513, March 1974.

[72] Wagner, H.M. and Whitin, T.M., "Dynamic Version of the Economic Lot Size Model", Management Science, 5, 1, 89-96, October 1958.

[73] Walker, W.E., "A Heuristic Adjacent Extreme Point Algorithm for the Fixed Charge Problem", Management Science, 22(5), 587-596, January 1976.

[74] Wolsey L., "Solving multi-item lot-sizing problems with an MIP solver using classification and reformulation", CORE Discussion Paper, 2002-2012, Universite Catholique de Louvain, Belgium, 2002.

[75] Zangwill, W., "A Deterministic Multi-period Production Scheduling Model with Backlogging", Management Science, 13, 105-119, 1966.