DOKUZ EYLÜL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

GENETIC ALGORITHM BASED HYBRID APPROACHES TO SOLVE CAPACITATED LOT SIZING PROBLEM WITH SETUP CARRYOVER AND BACKORDERING

by
Hacer GÜNER GÖREN

September, 2011
İZMİR

GENETIC ALGORITHM BASED HYBRID APPROACHES TO SOLVE CAPACITATED LOT SIZING PROBLEM WITH SETUP CARRYOVER AND BACKORDERING

A Thesis Submitted to the

Graduate School of Natural and Applied Sciences of Dokuz Eylül University

In Partial Fulfillment of the Requirements for the Degree of Doctor of

Philosophy in Industrial Engineering, Industrial Engineering Program

by

Hacer GÜNER GÖREN

September, 2011

İZMİR

Ph.D. THESIS EXAMINATION RESULT FORM

We have read the thesis entitled **"GENETIC ALGORITHM BASED HYBRID APPROACHES TO SOLVE CAPACITATED LOT SIZING PROBLEM WITH SETUP CARRYOVER AND BACKORDERING"** completed by **HACER GÜNER GÖREN** under supervision of **PROF. DR. SEMRA TUNALI** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Doctor of Philosophy.

Prof. Dr. Semra TUNALI

Supervisor

Prof. Dr. C.Cengiz ÇELİKOĞLU

Thesis Committee Member

Assoc. Prof. Dr. M. Arslan ÖRNEK

Thesis Committee Member

Doç.Dr. Seda Özmutlu

Examining Committee Member

Yrd.Doç. Dr. Derya Eren Akyol

Examining Committee Member

Prof. Dr. Mustafa ŞABUNCU
Director
Graduate School of Natural and Applied Science

ACKNOWLEDGMENTS

My Ph.D. journey started six years ago after a meeting with my advisor on a hot summer day. On that day, I was as excited as I am today. I owe a debt of gratitude to many people who made this journey wonderful in various ways and helped my dream become true.

First and foremost, I would like to express my deepest gratitude to Prof. Dr. Semra Tunalı, who has been my advisor, my mentor and my good friend during my Ph.D. study. I want to thank her for never stopping challenging me, for encouranging me and for always believing in me. Her valuable advices, continous support, guidance and inspiration shed light on my path for research. Besides being an advisor, she is also a great friend who listens to all my problems and gives me a lot of advices. Lastly, I am sincerely thankful for the opportunity to work with her. It was extremely helpful for my academic career and I can not think of a better advisor than her.

The second person whom I would like to thank to is Assoc. Prof. Dr. Raf Jans for his valuable suggestions and guidance throughout this Ph.D. study. Although he did not know me at first, he believed in me and invited me for one year to work together on lot sizing problems. Working with him was a pleasure and fruitful experience for me. I will always appreciate his friendship and support. Also, I would like to thank to his lovely wife Catherine for her hospitality and friendship when I was in Canada.

I am deeply thankful to my thesis committee members, Prof. Dr. C. Cengiz Çelikoğlu and Assoc. Prof. Dr. M. Arslan Örnek for their motivations and support during these years. I would also like to extend my gratitude to Assoc. Prof. Dr. Seda Özmutlu and Asst. Prof. Dr. Derya Eren Akyol for accepting to serve on my dissertation committee in the midst of all their activities.

I would like to acknowledge The Scientific and Technological Research Council of Turkey (TÜBİTAK) for providing me "2211-National Ph.D. Scholarship" and "2214-Ph.D. Foreign Research Scholarship" during my Ph.D. study.

I also need to thank all professors and collegues at the Department of Industrial Engineering of Dokuz Eylül University for their guidance. Special thanks go to my dear friends, Dr. Leyla Demir and Simge Yelkenci Köse for being always with me at good and tough times and providing endless support.

I want to emphasize my thankfulness to everyone at BuildDirect Technologies Inc. who showed great hospitality when my husband and I were in Canada. I would like to thank especially to Jeff Booth for giving us the opportunity of being together after three seperate years in our marriage. I want to extend my thanks to his lovely wife Kelly and his family for their friendships.

Words are not enough to express my gratitude and thanks to my family. I would like to thank to my dear mother Ayşe Güner and my dear father Arda Güner for their quiet sacrifice, unwavering love and support over the years which enabled me pursue my dreams. I feel extremely lucky to have such wonderful parents and I can only hope to be as great parent one day as they were. I am also deeply thankful to my dear and only sister Hale Güner for providing continous support, listening to all my complaints and trying to cheer me up when I am desperate.

Last, but the most, I would like to thank to one very special person in my life, my beloved husband Şükrü Gören for his unconditional love, support, encouragement and confidence in me. He is the one who always gives me the strength to carry on. Although I realized that I did not make our marriage easy for him through the years, he had never complained about it. So, I am also really grateful for his patience with me. Thank you for being where you are- right next to me, always. I wouldn't be the same person without you.

Hacer Güner Gören

GENETIC ALGORITHM BASED HYBRID APPROACHES TO SOLVE CAPACITATED LOT SIZING PROBLEM WITH SETUP CARRYOVER AND BACKORDERING

ABSTRACT

Lot sizing studies aim at determining the periods where production takes place and the quantities to be produced in order to satisfy the customer demand while minimizing the total cost. Having an important impact on the efficiency of production and inventory systems, lot sizing problem is one of the most challenging production planning problems. Due to their applications in production planning, lot sizing problems have been studied for many years with different features. Among these problems, The Capacitated Lot Sizing Problem (CLSP) has received a lot of attention from researchers. The primary aim of this Ph.D. study is to propose novel Genetic Algorithm (GA) based hybrid approaches for solving the CLSP with three extensions, i.e. setup times, setup carryover and backordering. In this thesis, the capacitated lot sizing problem with setup carryover and backordering is solved in two stages. In the first stage, two novel hybrid approaches are proposed for solving the capacitated lot sizing problem with setup times and setup carryover (CLSPC). These two hybrid approaches combine a meta-heuristic, i.e. GA, with a Mixed Integer Programming (MIP) based heuristic, i.e. the Fix-and-Optimize heuristic, in two different ways. In the first methodology, i.e. sequential hybridization, the Fix-and-Optimize heuristic is performed after the GA. The second methodology involves a different hybridization scheme where the Fix-and-Optimize heuristic is embedded into the GA. As an alternative to a random initial population, a novel initialization scheme which consists of problem specific information and randomness is proposed. Moreover, in order to sustain the feasibility during the search of GA, several repair operators are proposed. Lastly, the performances of proposed hybrid approaches are evaluated on various sets of problems from published literature. In the second stage, the CLSPC is extended to include the backorder option, called capacitated lot sizing problem with setup carryover and backordering. For solving the capacitated lot sizing problem with setup carryover and backordering, eight hybrid approaches are proposed. These approaches are modified versions of the hybrid approaches developed for solving the CLSPC. Unlike the hybrid approaches proposed in the first stage, in these hybrid approaches, the Fix-and-Optimize heuristic is implemented with different decomposition schemes. An extensive experimental analysis is carried out to compare the performances of the proposed hybrid approaches to the pure GAs using various problem instances. Moreover, the robustness of the performances of the proposed approaches under different parameter values is examined.

Keywords: Capacitated lot sizing problem, backordering, setup carryover, genetic algorithm, fix-and-optimize heuristic.

HAZIRLIK TAŞIMALI, BİRİKMİŞ SİPARİŞLİ KAPASİTE KISITLI PARTİ BÜYÜKLÜĞÜ PROBLEMİ İÇİN GENETİK ALGORİTMA TABANLI MELEZ ÇÖZÜM YAKLAŞIMLARI

ÖZ

Müşteri taleplerini zamanında karşılayacak ve toplam maliyeti en küçükleyecek şekilde üretimin olacağı dönemleri ve bu dönemlerde üretilecek ürün miktarlarını belirleme amacını taşıyan parti büyüklüğü problemleri, zor üretim planlama problemlerinden birisidir ve üretim ve stok sistemlerinin etkinliği üzerinde önemli bir etkiye sahiptir. Üretim planlamadaki uygulamalarından dolayı, farklı özellikleri taşıyan parti büyüklüğü problemleri uzun yıllardır çalışılmaktadır. Bu problemler arasında, Kapasite Kısıtlı Parti Büyüklüğü Problemi (KKPBP) araştırmacıların ilgisini en çok çeken problemlerden biridir. Bu doktora tezinin başlıca amacı hazırlık zamanları, hazırlık taşıma ve birikmiş sipariş özellikleri eklenen KKPBP'ni çözmek üzere özgün Genetik Algoritma (GA) tabanlı yaklaşımlar sunmaktır. Hazırlık taşımalı ve birikmiş siparişli KKPBP bu tez çalışmasında iki aşamada çözülmüştür. İlk aşamada, sadece hazırlık süreleri ve hazırlık taşımasının olduğu KKPBP (KKPBPC) için iki tane özgün melez yaklaşım önerilmiştir. Bu melez yaklaşımlar bir meta-sezgisel olan GA ve karışık tamsayı programlama (KTP) tabanlı bir sezgisel olan Sabitle-ve-Optimize Et sezgiselini iki farklı şekilde birleştirmektedir. İlk yaklaşımda, ardışık melezleme kullanılmış ve Sabitle-ve-Optimize Et sezgiseli GA'dan sonra uygulanmıştır. İkinci yaklaşım Sabitle-ve-Optimize Et sezgiselini GA'nin içerisine yerleştirerek farklı bir melezleme çeşidi içermektedir. Rastsal başlangıç popülasyonuna alternatif olmak üzere probleme özgü bilgileri ve rastsallık içeren özgün bir başlangıç popülasyonu oluşturma yöntemi önerilmiştir. Bunun yanı sıra, başlangıç popülasyonundaki probleme özgü ve rastsal kısımların oranlarını belirlemek için de bir deneysel çalışma yürütülmüştür. Ayrıca, GA'nın arama süresince olabilirliği sağlayabilmesi için çeşitli tamir operatörleri önerilmiştir. Son olarak da, önerilen yaklaşımların performansları literatürdeki mevcut problemler üzerinde test edilmiştir. İkinci aşamada, KKPBPC'ye birikmiş sipariş özelliği eklenmiş ve hazırlık taşımalı ve birikmiş siparişli KKPBP olarak adlandırılan bu problemi çözmek üzere sekiz farklı melez yaklaşım önerilmiştir. Bu melez yaklaşımlar, KKPBPC için önerilen melez yaklaşımları modifiye ederek geliştirilmiştir. İlk aşamada önerilen melez yaklaşımlardan farklı olarak, bu aşamada önerilen melez yaklaşımlarda Sabitle-ve-Optimize Et sezgiseli farklı şekillerde uygulanmış ve problemin ayrıştırılmasında çeşitli ölçütler kullanılmıştır. Farklı problem örnekleri üzerinde, önerilen yaklaşımların performansı GA ile karşılaştırılmıştır. Ayrıca, önerilen yaklaşımların performanslarının problem parametrelerindeki değişikliklere ne kadar duyarlı olduğu araştırılmıştır.

Anahtar sözcükler: Kapasite kısıtlı parti büyüklüğü problemi, birikmiş sipariş, hazırlık taşıma, genetik algoritma, sabitle-ve-optimize et sezgiseli.

CONTENTS

Page

CHAPTER ONE
INTRODUCTION

1.1 Objectives and Motivations

Considering the increasing interest in operations, service and logistics costs, strategic planning decisions such as allocating scarce resources and operations scheduling have important effect on the success of many industrial firms. The problem of satisfying customer demands on time at the lowest possible cost is complicated and hard to determine which requires complicated solution approaches for decision support.

Production planning is one important area in strategic decisions that considers the best use of production resources such as parts, raw materials, machines and labor, in order to satisfy production goals over a certain period named *planning horizon*. It encompasses three time ranges for decision making: *long-term*, *medium-term* and *short-term*. The *long-term* planning focuses on strategic decisions such as product, equipment and process choices, facility location and design and resource planning. The *short-term* planning usually involves decisions related to the day-to-day scheduling of operations such as sequencing or control in a workshop. The focus of the *medium-term* planning is making decisions on material requirements planning (MRP) and establishing production quantities or *lot sizing* over the planning horizon (Karimi et al., 2003).

Lot sizing problems determine whether and how many items (i.e. lot size) to produce for a particular product for a given horizon. *Lot* is a batch of the items of the same type. The production lots are determined by the trade-offs among machine setup costs, production costs, and inventory holding costs (Gao, 1998). The overall objective is to satisfy customer demands at the lowest possible cost under a set of constraints.

Depending on the characteristics of the production process and the planning detail required, different types of lot sizing models are commonly used in practice (Suerie & Stadtler, 2003). Lot sizing models fall into either small bucket or big bucket problems. In big bucket problems, the time period is long enough to produce more than one product on each resource, whereas in small bucket models the time period is short that at most one product can be produced thus allowing at most one setup per period and machine. The capacitated lot sizing problem (CLSP) can be defined as an example for big bucket models. The discrete lot sizing and scheduling problem (DLSP), continuous setup lot sizing problem (CSLP) and proportional lot sizing and scheduling (PLSP) problem are considered to be small bucket models. The small bucket models solve the lot sizing and scheduling problems together; however the big bucket models only deal with the lot sizing decisions. Moreover, in small bucket models, carrying of at most one setup of a product from one period to another is permitted while this property is not valid for big bucket models. Among these problems, a vast amount of literature has been devoted to the solution of CLSP with different extensions. Since the general case of the single item CLSP is shown to be NP-Hard (Florian et al., 1980), almost all studies in this area focus on proposing efficient solution approaches for solving this hard problem.

The primary aim of this PhD study is to introduce efficient solution approaches for an extended version of the CLSP, namely the capacitated lot sizing problem with setup times, setup carryover and backordering which deals with determining the quantity and timing of production lot sizes and also the semi-sequencing of the products to be produced in each period.

The considered problem extends the classical CLSP with respect to three additional aspects:

Firstly, ***setup times*** are considered in this study. The capacity lost due to cleaning, preheating, machine adjustments, calibration, inspection, test runs, change in tooling and starting up a new product is considered as setup time. Considering the setup time issue makes the problem more complicated, but ignoring it makes the implementation of the methods suggested impractical for real-life applications.

Secondly, ***a setup carryover*** from one period to another is permitted. The most common practice in the relevant literature is to model the lot sizing problem as small bucket problem in order to obtain more accurate plans. However, to yield a solution comparable to that of a big-bucket model it is required to divide the planning horizon into many more buckets (Suerie & Stadtler, 2003). This increases the complexity of the model. In recent years, a new model combining the characteristics of small and big bucket models has emerged. This model is called the capacitated lot sizing problem with setup times and setup carryover (CLSPC). The CLSPC is a big bucket model but it allows carrying one setup state of a product from one period to the next. Allowing setup carryover also helps finding feasible solutions in such situations where too much capacity is consumed by setup times that are not necessary in reality.

Thirdly, the model in this study allows ***backordering*** meaning that if customer demand can not be met on time, it can be satisfied later in future periods of the planning horizon. In traditional lot sizing models, this issue is generally ignored and a product is produced prior to its delivery date. As a result, inventory costs occur. However, in real life problems, it might not always be possible to satisfy the customer demand on time and unsatisfied demand is often backordered. Due to the delay of customer needs, backorder costs are incurred for every unit and period of the delay. Allowing backorders has a great importance in practical settings as sometimes some products may have to be backordered since capacity is limited (Quadt & Kuhn, 2009).

In this Ph.D. study, we first focused on the capacitated lot sizing problem with setup times and setup carryover and proposed two novel GA based hybrid approaches to solve this problem. Next, we added backordering issue to this problem and developed eight GA based approaches to solve the extended problem, i.e. the capacitated lot sizing problem with setup times, setup carryover and backordering.

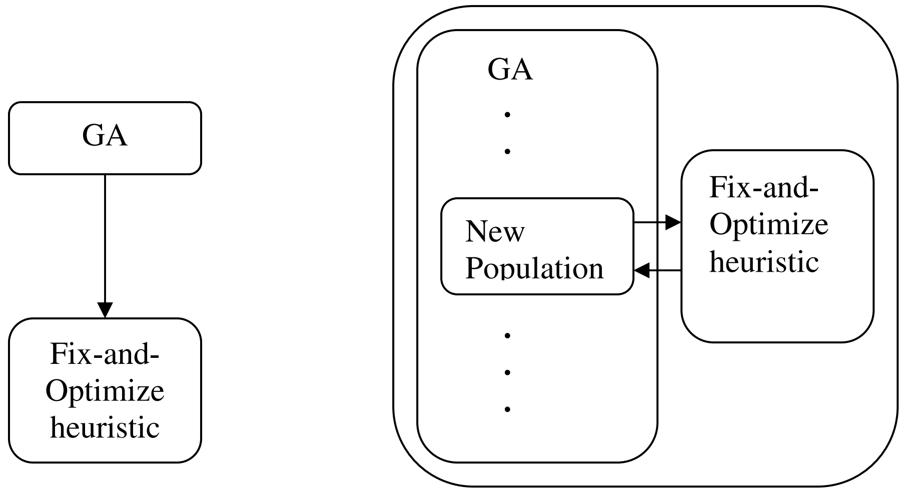
1.2 Research Methodology

The general problem considered in this study is an extension of the classical CLSP. Several optimum seeking methods such as linear programming, integer programming, dynamic programming and branch-and-bound approaches have been used to solve this problem. However, none of these methods have proven to be effective especially for large size problems due to their computational inefficiency. Hence, in recent years, research efforts have been directed to the development of several heuristic approaches.

Among these heuristic approaches, in recent years, evolutionary computation has received increasing attention. The most well known evolutionary computation method is Genetic Algorithms (GAs). GAs are optimization techniques that use the principles of evaluation and heredity to arrive at near optimal solutions to difficult problems (Khouja et al., 1998). GAs have been employed to solve different optimization problems across various disciplines due to their flexibility and simplicity. However, as the problem complexity increases the search space becomes very large and pure GAs may lack the capability of exploring the solution space effectively (Taşan, 2007). To improve the exploration capability of pure GAs for faster and better search in recent years, the attention has focused on the hybridization of GAs. In hybrid GAs, local search methods, problem specific information, other meta-heuristics and exact approaches are used.

In this PhD study, we first solve the CLSPC by employing two novel hybrid approaches. These hybrid approaches include two different hybridization schemes namely, sequential and embedded (see Figure 1.1). The first methodology hybridizes GAs and Fix-and-Optimize heuristic in a sequential way, where the Fix-and-Optimize heuristic is performed after GA. In the second one, the Fix-and-Optimize heuristic is embedded into the loop of GAs to refine the solutions obtained by GAs.

Next, we add the backorder issue into the model and solve this extended problem by employing eight hybrid approaches. While the first four of these approaches are the modified versions of the proposed sequential hybrid approach, the remaining four are the modified versions of the proposed embedded hybrid approach. The Fix-and-Optimize heuristic in these hybrid approaches is applied with different decomposition schemes.



(a) sequential hybridization scheme

(b) embedded hybridization scheme

Figure 1.1 Proposed hybridization schemes

Moreover, to further improve the performance of the proposed hybrid approaches, a novel initialization scheme is proposed for creating the initial population of the GA and its efficiency is tested under various experimental conditions. This initialization scheme utilizes both problem specific information and randomness to create the initial members of the population. A repair procedure employing some novel repair operators is embedded into GAs to fix the infeasible individuals in each generation. Furthermore, to improve the performance of the proposed hybrid approaches, an extensive experimental analysis is carried out to identify efficient GA control parameter settings.

1.3 Outline of the Thesis

The primary focus of this thesis is to develop efficient GA-based solution approaches for solving the capacitated lot sizing problem with setup times, setup carryover and backordering. A brief outline of this thesis is as follows.

In Chapter 2, background information on lot sizing problems are provided with the problem specifications, variants and solution approaches that have been proposed so far.

In Chapter 3, the solution approaches employed in this Ph.D. study, GAs and Fix-and-Optimize heuristic are explained in detail and also hybridization concepts in meta-heuristics are presented.

In Chapter 4, to determine the research gaps in the current literature, a comprehensive literature review on applications of GAs for lot sizing problems is presented. The focus of literature review is twofold. Firstly, the current relevant research is reviewed from the perspective of lot sizing problem specifications; next, the features of GA-based methodologies are discussed to identify possible methodological contributions.

In Chapter 5, two hybrid approaches are proposed to solve the CLSPC. Additionally, a new initialization scheme is proposed and efficient control parameters of GA are determined through pilot experiments to improve the performance of the proposed hybrid approaches. The performances of proposed hybrid approaches are tested on a set of benchmark problems from the literature and the results of comparative experiments are presented.

In Chapter 6, the CLSPC is extended to the capacitated lot sizing problem with setup times, setup carryover and backordering, abbreviated to $CLSP^+$. The proposed hybrid approaches are modified to deal with this more complicated problem. To further improve the performance of these approaches the efficient control parameters of GA are determined based on pilot experiments and also Fix-and-Optimize heuristic is implemented with new decomposition schemes. Various sets of computational experiments are carried out on a set of problem instances ranging from small to large size. Moreover, a statistical analysis is carried out to see whether there are statistically significant differences between the performances of proposed hybrid approaches. Lastly, the sensitivity of the performances of the proposed hybrid approaches to various parameters including backorder cost, capacity utilization, time between orders (TBO), demand variability and setup time are examined in detail.

Finally, in Chapter 7, the summary and contributions of this thesis study are discussed. The future research directions are also presented.

CHAPTER TWO
BACKGROUND INFORMATION FOR LOT SIZING PROBLEMS

2.1 Introduction

The problem considered in this study is the extended version of the capacitated lot sizing problem. This chapter is devoted to the definition of several lot sizing problems, introduction to some of the most important concepts of lot sizing problems and discussion of several solution approaches. The chapter is organized as follows. In Section 2.2, a brief introduction to basic concepts of lot sizing problems is given. In Section 2.3, different variants of lot sizing problems are explained. In Section 2.4, various solution methods proposed for solving the capacitated lot sizing problem are discussed. Finally, in Section 2.5, the context of this chapter is summarized.

2.2 Lot Sizing

Many production processes can only start after the required resources have been set up which involves a setup time and/or setup cost (Buschkühl et al., 2008). Simply, finding the timing and quantity of production to satisfy the customer demand so that production, setup and inventory costs can be minimized, known as lot sizing problem. Since lot sizing problems are critical to the efficiency of production and inventory systems, it is very important to determine the right lot sizes in order to minimize the overall cost (Gören et al., 2010).

Research on lot sizing has started in the early twentieth century and since then a large number of lot sizing problems with different modeling features have been identified (Buschkühl et al., 2008). In order to solve these problems, a lot of solution approaches and algorithms have been developed.

2.2.1 Problem Specifications of Lot Sizing Problems

The complexity of lot sizing problems is dependent on the problem specifications taken into account by the model. The problem specifications can be named as the *planning horizon*, *number of levels*, *number of products*, *capacity constraints*, *type of demand*, *setup time issue* and *inventory shortage*. These will be explained in details in the following.

a. Planning horizon:

The planning horizon can be defined as the time interval on which the master production schedule extends into the future (Karimi et al., 2003). The planning horizon can be *infinite*, *finite* or *rolling*. An *infinite* planning horizon is usually accompanied by stationary demand, whereas a *finite* planning horizon is accompanied by dynamic demand. Under *rolling* horizon, a production planning is made for a fixed number of periods for which the demand is known. The first production decision is implemented and the horizon is rolled forward to the period where the next production decision needs to be made (van den Heuvel & Wagelmans, 2005).

b. Number of levels:

Production systems can be *single level* or *multi level*. In single level systems, raw materials are changed into the final process after a single operation. Product demands come directly from the customer orders or market forecasts. This is the *independent demand*. However, in multi level production systems, there is a relationship among products which create *dependent demands*. The output of one level is input for another operation. Raw materials are converted into final products after several operations.

c. Number of products:

Another important problem specification affecting the modeling and complexity of the problem is the number of end products considered. In terms of number of products, there are two types of production systems. The first one is the *single item* production system, where there is only one end product. The second one is the *multi item* production system, where there are several end products.

d. Capacity constraints:

When there are no limitations on the resources, the lot sizing problem is said to be uncapacitated. The capacitated lot sizing problems are more complicated than the uncapacitated lot sizing problems since the capacity constraints directly affect the problem complexity.

e. Type of demand:

If the value of the demand is known, it is termed as deterministic which can be static or dynamic. If the value of the demand is not known exactly and occurs based on some probabilities, it is termed as probabilistic.

f. Setup time issue:

In capacitated lot sizing problems, adding the setup time issue makes the problem more complex (Degraeve & Jans, 2007). A production changeover between different products incurs setup time and setup cost. The setup structure can be defined into two types, namely, *simple* and *complex* setup structures.

Simple setup structure: If the setup time and cost in a period do not depend on the sequences, the decisions in previous periods or decisions for other products, the setup structure is said to be *simple*.

Complex setup structure: The complex setup structure can be defined in three types. The first one involves the *setup carryover* which allows the continuation of the production run from the previous period into the current period without any additional setup. The second one involves the *family* or *major* setup which is caused by the similarities in manufacturing process and design of group of products (this is related to decision for other products, but in the same period). This type of the setup structure also involves an *item* or *minor setup* which occurs during the change of the production among products within the same family. The last type of complex setup structure occurs when the setup decisions depend on the production sequence which is called *sequence dependent* (Karimi et al., 2003).

g. Inventory shortage:

Inventory shortage is an important specification affecting the problem modeling and complexity. If inventory shortage is allowed, it is possible to satisfy the demand of the current period by production in future periods. This is called *backordering*. There is another option that the demand of the current period may not be satisfied at all. This is called *lost sales*.

2.3 Variants of Lot Sizing Problems

Research on lot sizing starts with the classical economic order quantity (EOQ) model. The EOQ model is developed for a single level production process with a single product and no capacity constraints under stationary demand. Since the assumptions of the EOQ model do not appear realistic, other models have evolved. The classification of lot sizing problems based on the main specifications explained above is given in Figure 2.1 which is adapted from the classification in Bahl et al. (1987).

The first group of lot sizing problems is the *static lot sizing problems* namely the Economic Lot Scheduling Problem (ELSP). The ELSP is a single level multi item problem with capacity constraints under stationary demand and infinite planning horizon. It deals with scheduling the production of a set of products on a single machine to minimize the long run average holding and setup cost under the assumptions of known constant demand and production rates. The objective of the ELSP is to determine a production cycle of N products, $i \in \{1, 2, ..., N\}$ in a repetitive schedule. A repetitive schedule is achieved if there is a time period T_i for each product that represents the time between successive production runs (batches or "lots") of product i (Chatfield, 2007). The repetitive schedule is subject to the following conditions related to the production facility and marketplace, as stated in Bomberger (1966).

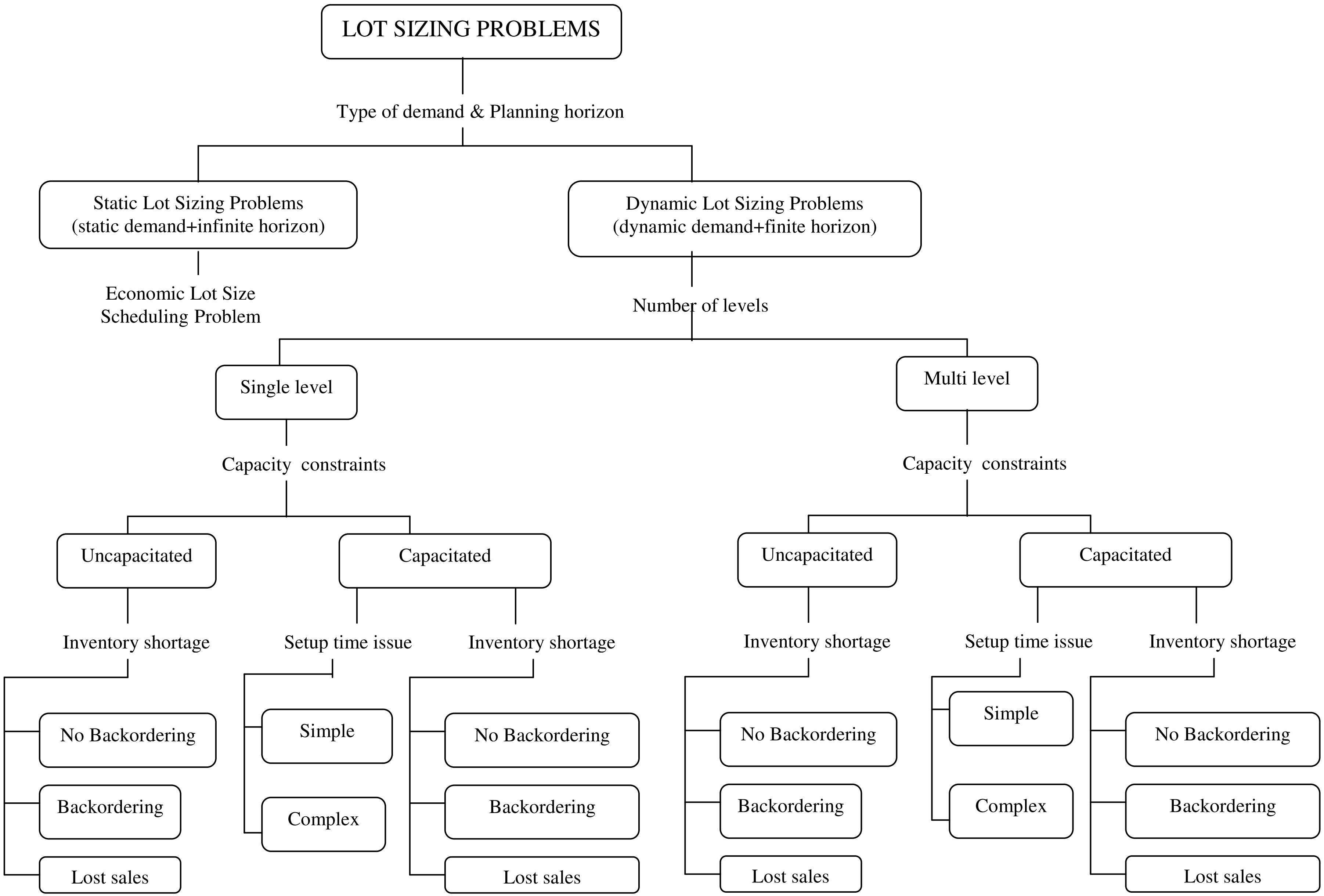


Figure 2.1 A classification of lot sizing problems

1. Only one product *i* can be produced at a time.
2. Setup for a certain product incurs both a specific setup cost (s_i) and setup time (t_i).
3. Setup time and setup cost are determined solely by the product going into the production (sequence independent).
4. Demand rate (r_i) and production rate (p_i) are known and constant for all products.
5. All demand must be met, which means backordering is not allowed.
6. Holding costs (h_i) are determined by the value of products held.
7. Total variable cost for a product equals the average setup cost plus holding cost over a specific period of time.
8. Production time for a lot of product *i*, σ_i, equals the sum of the processing time and the setup time, $\sigma_i = (r_i / p_i) * T_i + t_i$.

A solution set consists of a set $T = \{T_1, T_2, ..., T_N\}$ such that T_i is sufficiently long enough to allow enough production of product *i* at the beginning of the cycle to meet the demand during the entire cycle T_i, plus allow production of other products in the time left between the end of production of product *i* and the start of the next cycle. The cost per unit for a product *i* is defined as in the following.

$$C_i = (average \quad setup \quad \cos t + average\ holding\ \cos t) \tag{1}$$

$$C_i = \frac{s_i}{T_i} + \frac{h_i r_i T_i (p_i - r_i)}{2 p_i} \tag{2}$$

Due to the non-linearity and combinatorial characteristics of the problem, the ELSP falls into the class of NP-Hard problems.

The second group is the *dynamic lot sizing problems* which deal with dynamic demand under a finite planning horizon. The dynamic lot sizing problem can be formulated for a single level with infinite production capacity and a single product over *P* periods of time as follows:

$$Minimise \sum_{t=1}^{T} (S_t Y_t + c_t X_t + h_t I_t) \quad (3)$$

$$s.t. X_t + I_{t-1} - I_t = d_t \quad (\forall t \in P) \quad (4)$$

$$X_t \leq M_t Y_t \quad (\forall t \in P) \quad (5)$$

$$Y_t \in \{0,1\} \quad (\forall t \in P) \quad (6)$$

$$X_t, I_t \geq 0 \; (\forall t \in P) \quad (7)$$

This problem is known as the uncapacitated single item single level lot sizing problem (Wagner & Whitin, 1958), where h_t is the inventory holding cost of the product from one period to the next, d_t represents the product demand at the end of period t, C_t is the variable unit production cost in period t, S_t is the setup cost in period t and Y_t is a binary variable that assumes value 1 if the product is produced in period t and 0 otherwise. M_t is the upper bound on the production. The decision variables X_t represent the production level in each period t and I_t represents the inventory variable of the product at the end of period t. The objective function, Equation 3, includes total holding, setup and production costs. The inventory balance equation for each period is given in Equation 4. The second constraint, Equation 5, forces the setup variable to take the value 1 if there is any production. Equations 6 and 7 are the non-negativity constraints.

The first dynamic lot sizing model was developed in 1958 by Wagner and Whitin. The problem is a single item single level uncapacitated lot sizing problem where there are constant production costs over the planning horizon. In 1960, it was proved by the authors that for the production costs that are not constant there exists an optimal solution that satisfies the following property:

$$I_{t-1} X_t = 0 \quad \forall t \in P \quad (8)$$

This property means that in an optimal solution, one never produces in a period and has inventory coming in from the previous period at the same time. This is called the Wagner-Whitin (WW) property and the problem is said to be WW problem.

2.3.1 The Capacitated Lot Sizing Problem

The Capacitated Lot Sizing Problem (CLSP) can be seen as the extension of the WW problem to capacity constraints. The CLSP can be grouped in large bucket models thus similar to the ELSP; the CLSP is a multi product problem (Drexl & Kimms, 1997).

The linear programming formulation for the CLSP was proposed by Manne in 1958. There are n different products to be produced on a single machine with a production capacity C_t and K is the set of all products. Producing one unit of product i consumes a_j units of capacity, the variable production time. An extra index j is used for defining the product specific variables and parameters. The formulation in the original production and setup variables, X_{jt} and Y_{jt}, is as follows:

$$Min\sum_{j\in K}\sum_{t\in P}(S_{jt}Y_{jt}+C_{jt}X_{jt}+h_{jt}I_{jt}) \quad (9)$$

$$s.t. \quad I_{j,t-1}+X_{jt}-I_{j,t}=d_{j,t} \quad \forall j\in K,\forall t\in P \quad (10)$$

$$X_{jt}\leq \min\{C_t/a_j, sd_{jtm}\}Y_{jt} \quad \forall j\in K,\forall t\in P \quad (11)$$

$$\sum_{j\in K}X_{jt}a_j\leq C_t \quad \forall t\in P \quad (12)$$

$$X_{jt}, I_{jt}\geq 0; Y_{jt}\in\{0,1\} \quad \forall j\in K,\forall t\in P \quad (13)$$

The objective function (9) minimizes the total cost for all products. The demand equations (10) remain same. In the setup constraints (11), production is limited by both capacity and remaining demand which is different from the uncapacitated dynamic lot sizing problem. Total production in each period is now limited by the capacity constraint defined in (12). Equations (13) define the integrality constraints of the decision variables.

The CLSP can be extended to include the setup times. The setup times represent the capacity lost due to cleaning, preheating, machine adjustments, calibration, inspection, test runs, change in tooling etc., when the production for a new product starts. The setup time must be accounted for in the capacity constraint (Jans & Degraeve, 2008). Several studies can be found in the literature considering setup times for the CLSP (Manne, 1958; Trigeiro et al., 1989; Gopalakrishnan et al., 2001; Degraeve & Jans, 2007; Hindi et al., 2003; Jans & Degraeve, 2004).

2.3.2 The Discrete Lot Sizing and Scheduling Problem

Subdividing the (macro-) periods of the CLSP into several (micro-) periods leads to the DLSP (Drexl & Kimms, 1997). The Discrete Lot Sizing and Scheduling Problem (DLSP) is a small bucket problem in which at most one type of product is produced. In DLSP, a discrete production policy is assumed (i.e. *all-or-nothing* assumption), meaning a product must be produced at full capacity (Jans, 2002). The mathematical programming model of the DLSP can be expressed as follows.

$$Min \sum_{j\in K} \sum_{t\in P} (g_{jt} z_{jt} + S_{jt} Y_{jt} + C_{jt} X_{jt} + h_{jt} I_{jt}) \tag{14}$$

$$s.t. \quad I_{j,t-1} + X_{jt} = d_{jt} + I_{jt} \quad \forall j \in K, \forall t \in P \tag{15}$$

$$\sum_{j=1}^{n} y_{jt} \leq 1 \quad \forall t \in P \tag{16}$$

$$a_j X_{jt} = C_t Y_{jt} \quad \forall j \in K, \forall t \in P \tag{17}$$

$$z_{jt} \geq Y_{jt} - Y_{j,t-1} \quad \forall j \in K, \forall t \in P \tag{18}$$

$$X_{jt}, I_{jt} \geq 0; Y_{jt}, z_{jt} \in \{0,1\} \quad \forall j \in K, \forall t \in P \tag{19}$$

The new variable z_{jt} is the start up variable and there is an associated start up cost of g_{jt}. A start up occurs when the machine is set up for an item for which it was not set up in the previous period. The objective function (14) minimizes the total cost of start ups, setups, variable production and inventory. The regular demand constraints are stated in (15). The constraints (16) impose that at most one type of product can be made in each time period. For each product, production can be at full capacity if there is a setup as shown in (17). The start up variables are modeled in constraints

(18). There will be only one start up if the machine is set up for an item for which it was not set up in the previous period. A setup can be carried over to the next period if the production of the same product is continued. The constraints (19) show that the start up and set up variables are binary.

Fleischmann (1990) solves DLSP with sequence-independent setup costs using branch-and-bound where in a further study Fleischmann (1994) adds the sequence dependent costs into the model of DLSP. Cattrysse et al. (1993) propose a heuristic for the DLSP with positive setup times based on dual ascent and column generation techniques. Jordan and Drexl (1998) show the equivalence between DLSP for a single resource and the batch sequencing problem.

2.3.3 The Continuous Setup Lot Sizing Problem

The Continuous Setup Lot Sizing Problem (CSLP) is more realistic than the DLSP. The all-or-nothing assumption does not exist in the CSLP but there is still only one product that can be produced per period (Drexl & Kimms, 1997). The generic model has a similar structure as the DLSP (14)-(19), except that the capacity and set up constraints (17) become an inequality:

$$a_j X_{jt} \leq C_t Y_{jt} \quad \forall j \in K, \forall t \in P \tag{20}$$

Karmarkar and Schrage (1985) consider this problem without setup costs and in a later study, Karmarkar et al. (1987) focus on the single item version of the CSLP with and without capacity constraints. Wolsey (1989) refers this problem as *lot sizing with start up costs*.

2.3.4 The Proportional Lot Sizing and Scheduling Problem

In the CSLP, if the capacity of a period is not used in full, the remaining capacity is left unused. In order to avoid this problem, The Proportional Lot Sizing and Scheduling Problem (PLSP) has emerged (Drexl & Kimms, 1997). The basic idea of the PLSP is to use the remaining capacity for a second product per period. The underlying assumption of the PLSP is that the setup state can be changed at most once per period. Production in a period may take place if the machine is properly setup either at the beginning or at the end of the period. Hence, at most two items may be produced per period (Drexl & Kimms, 1997). If two products are produced in a period, then the first product must be the same as the last product in the previous period. Drexl and Haase (1995, 1996) extend this model with setup times and multi machines. The multi level version of the PLSP can be found in Kimms (1996a, 1996b, 1999).

2.3.5 The Capacitated Lot Sizing Problem with Setup Carryover

In practice, to obtain more accurate plans, smaller bucket sizes are usually preferred. But in small bucket problems, to yield an optimum solution the planning horizon is divided into more buckets than big bucket models and this increases the complexity of the problem since the numbers of constraints and variables increase. Therefore, a new model called "The Capacitated Lot Sizing Problem with Setup Carryover", which combines the big-bucket and small-bucket models, has recently received the attention of researchers in recent years.

The capacitated lot sizing problem with setup carryover (CLSPC) is also called "The Capacitated Lot Sizing Problem with Linked Lot Sizes (CLSPL)" as indicated in Suerie and Stadtler (2003). The main characteristics of the CLSPL, which is a big bucket model (see Figure 2.2), can be summarized as follows (Suerie & Stadtler, 2003):

1. Several products requiring a unique setup state can be produced on each resource in each period (big bucket model).
2. At most one setup state can be carried over from one period to the next. So that two lots of adjacent periods are linked, requiring no new setup activity in the second period.

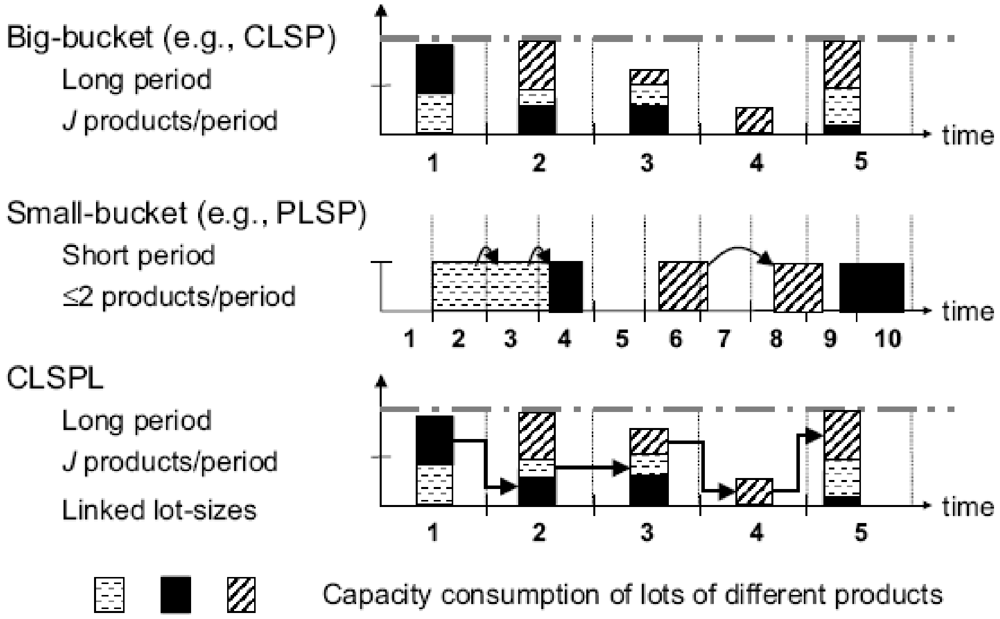


Figure 2.2 Characterization of lot sizing and scheduling models (Suerie & Stadtler, 2003)

Sox and Gao (1999) present a reformulation of the mathematical programming model for the CLSPC. The reformulation is based on a shortest-route representation and a Lagrangian decomposition heuristic is proposed to solve the problem. The reformulation allows multi period setup carryovers for small size problems (i.e. eight products, eight periods, one resource). The authors, however, add the constraint of single period setup carryover to deal with large size problems. The first meta-heuristic application addressing the solution for the CLSPC can be found in Gopalakrishnan et al. (2001). The authors propose a Tabu Search (TS) heuristic which consists of five basic move types, three for the sequencing and two for the lot sizing decisions. In another study, Karimi and Ghomi (2002) propose a four-stage greedy heuristic approach for the capacitated lot sizing problem with setup carryover and backlogging. The feasibility of the production plan is maintained with lot shifting. However; the model does not include setup times. Karimi et al. (2006) extend their previous work (Karimi & Ghomi, 2002) by suggesting a TS approach to solve the same problem. Parallel with the conclusions derived from the study of Gopalakrishnan et al. (2001), Porkka et al. (2003) modify the model proposed by Sox and Gao (1999) by using setup times instead of fixed setup costs and compare its behavior with a benchmark model without the setup carryover. The results show that counting the setup times and setup carryover cuts down the number of setups and also frees a significant amount of capacity.

Suerie and Stadtler (2003) propose an extended formulation and valid inequalities for the CLSPC under the assumption of conservation of one setup state for the same product over two consecutive periods which leads to linking two lots of the adjacent periods together. The authors also propose a time decomposition heuristic for solving the problem for both the single and multi level case.

The sequence-dependent setup costs and times are taken into account in Gupta and Magnusson (2005). However, the exact formulation of the problem consists of a large number of binary variables and also the issue of sequence-dependent setup times and costs make the problem more complicated. To deal with large problem instances the authors propose a heuristic approach coupled with a procedure for obtaining a lower bound on the optimal solution. Motivated by a real world problem in the glass container industry, Almada-Lobo et al. (2007) present two novel Mixed Integer Programming (MIP) formulations for the CLSPC, sequence dependent setup times and costs. A five-step heuristic is proposed in which the first two steps attempt to find an initial feasible solution and the last three are geared towards improving the quality of the solution. The idea of the multi-plants is incorporated in the CLSPC in Nascimento and Toledo (2008). The authors propose a GRASP meta-heuristic to solve the problem with this idea.

The problem is also extended to parallel machines by Quadt and Kuhn (2009) and a period-by-period heuristic is proposed to solve the capacitated lot sizing problem with setup carryover and backordering in parallel machine environment.

2.4 Solution Approaches for the Lot Sizing Problems

Complexity theory and computational experiments indicate that most lot sizing problems are hard to solve (Jans & Degraeve, 2007). To deal with the complexity and find optimal or near-optimal results in reasonable computational time, various solution approaches have been proposed to solve different types of lot sizing problems. Since this study deals with an extension of the CLSP namely the capacitated lot sizing problem with setup times, setup carryover and backordering, the solution approaches that have been proposed for solving other types of lot sizing problems (i.e. ELSP, DLSP, PLSP, CSLP) are out of the scope of this chapter.

In the following sub-sections, we mainly focused on the solution approaches used to solve CLSP with different extensions and investigated the relevant literature according to the classification given in Figure 2.3 which is adapted from the classification given in Buschkühl et al. (2008).

2.4.1 Exact Methods

Apart from using branch-and-bound technique to solve the CLSP, there are other exact approaches such as reformulations and valid inequalities. Since these methods need considerable computational time to find an optimal solution, they can only be used for small size problems. Table 2.1 lists the number of binary and continuous variables along with the number of constraints for the CLSP and some of its extensions.

Table 2.1 Model sizes of the CLSP and some of its extensions (Quadt & Kuhn, 2008)

Model	Binary variables	Continous variables	Constraints
CLSP	*PT*	2*PT*	5*PT*+*P*+*T*
CLSP+ Backorder	PT	3*PT*	6*PT*+3*P*+*T*
CLSP + Setup Carryover (Linked Lot Sizes)	2*PT*	2*PT*	*PPT*+6*PT*+2*P*+2*T*
CLSP + Sequence dependent	*PPT*+*PT*	3*PT*	*PPT*+8*PT*+2*P*+2*T*
CLSP + Parallel Machines	*PTM*	*PTM*+*PT*	3*PTM*+2*PT*+*TM*+*P*
CLSP + Backorder + Parallel Machines	2*PTM*	*PTM*+2*PT*	*PPTM*+4*PTM*+3*PT*+*PM*+*TM*+3*P*+*T*

* *M* number of parallel machines, *P* number of products, *T* number of periods

2.4.1.1 Branch-and-Bound

Branch-and-bound (B&B) method is an exact solution procedure that enumerates feasible solutions implicitly (Buschkühl et al., 2008). The method has two parts namely, "*branching*" and "*bounding*". "Branching" generates new disjoint subsets in the solution space while "bounding" removes the unpromising ones from the solution space. For MIP models with binary variables, branching is based on subsequently fixing the binary variables to 0 and 1 and a relaxed version of each sub-problem is solved to determine a bound. There are different ways to relax the mathematical model such as linear programming (LP) relaxation and lagrangian relaxation. Different B&B applications which are embedded into Lagrangian relaxation scheme can be found in Billington et al. (1986), Chen and Thizy (1990) and Diaby (1992a, 1992b).

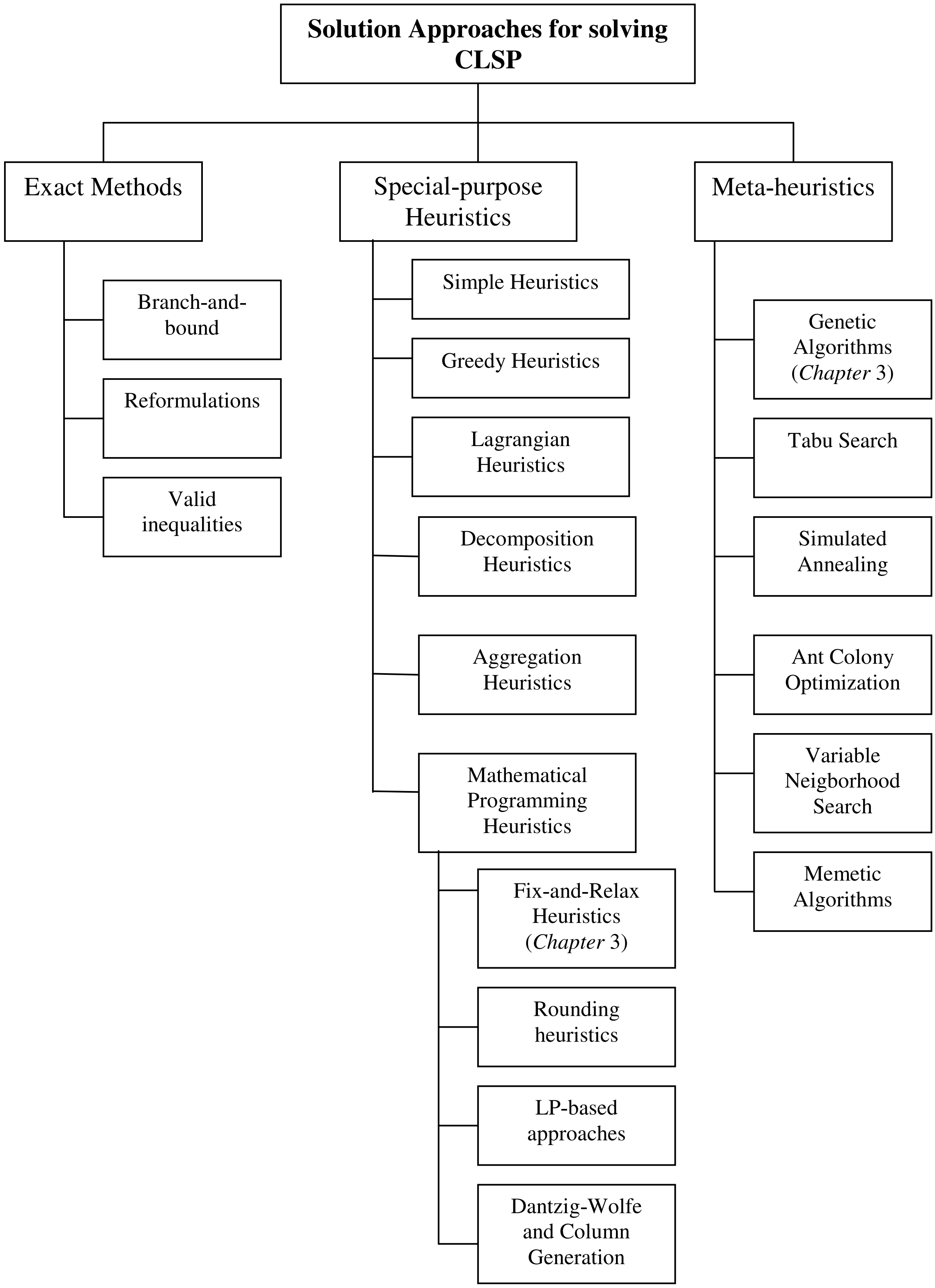


Figure 2.3 Classification of solution approaches for solving CLSP

2.4.1.2 Reformulations

The bounds obtained by relaxing the regular formulation of the CLSP consisting of the inventory and production variables are quite poor. For this reason, one of the research trends in lot sizing area is to reformulate the model and redefine the corresponding decision variables. Two reformulations have been introduced which assign each production quantity to a corresponding demand quantity (Buschkühl et al., 2008). The first one is the simple plant location and the second one is the shortest route reformulations.

Simple Plant Location Reformulation

The idea behind the Simple Plant Location Reformulation (SPL) is that a product/period combination can be regarded as a “plant location” (Rosling, 1986). Only if the “plant location” is set up, it may produce the current period’s demand of the particular product as well as any subsequent demand (Stadtler, 1996).

Rosling (1986) introduces the SPL reformulation for the multi level uncapacitated lot sizing problem as an extension of the work by Krarup and Bilde (1977). Then, capacity constraints are added to the model by Maes et al. (1991). Stadtler (1996) extends these models considering only the assembly-type bill-of material structures to the case of general bill-of-material structures.

In this formulation, the production quantity variables are replaced by variables Z_{st} in the regular formulation according to

$$X_t := \sum_{s=t}^{P} D_s Z_{ts} \qquad (21)$$

D_t denotes the demand for product in period t, and the variable Z_{st} represents the portion of demand of product produced in period s ($s \leq t$) to fulfill the demand in period t. The SPL model for the single product uncapacitated lot sizing problem can be formulated as follows:

$$Min \sum_{s=1}^{P-1} \sum_{t=s}^{P} h_t (t-s) D_t Z_{st} + \sum_{t=1}^{P} S_t Y_t \quad (22)$$

$$s.t. \quad Z_{ts} \le Y_t \qquad \forall s,t \in P, s \le t \quad (23)$$

$$\sum_{s=1}^{t} Z_{st} = 1 \quad \forall s \in P \quad (24)$$

$$Y_t \le 1 \quad \forall t \in P \quad (25)$$

$$Z_{st}, Y_t \ge 0 \quad \forall s,t \in P, s \le t \quad (26)$$

A setup is required in period t for the product whenever a production takes place to cover the demand of period t or any subsequent period s as shown in constraint (23). Constraint (24) imposes that the demand in each period is satisfied from production in that period or from a previous period. The simple upper bound on the setup variables (Y_t) is stated in constraint (25). The non-negativity constraints are given in (26).

Shortest Route Reformulation

The other reformulation is the shortest route (SR) reformulation which is introduced by Eppen and Martin (1987). In this formulation, a new variable zv_{tk}, which represents the fraction of demand from period t to k that will be satisfied by the production in period t, is introduced into the model. Based on the Wagner-Whitin property, the zv_{tk} variable can be imposed to be binary. If zv_{tk} equals one, then in period t we can produce the demand for period t until period k (Jans, 2002). So, the lot sizing problem can be described as a shortest path network problem. Figure 2.4 shows an example of a shortest network problem for a four period problem.

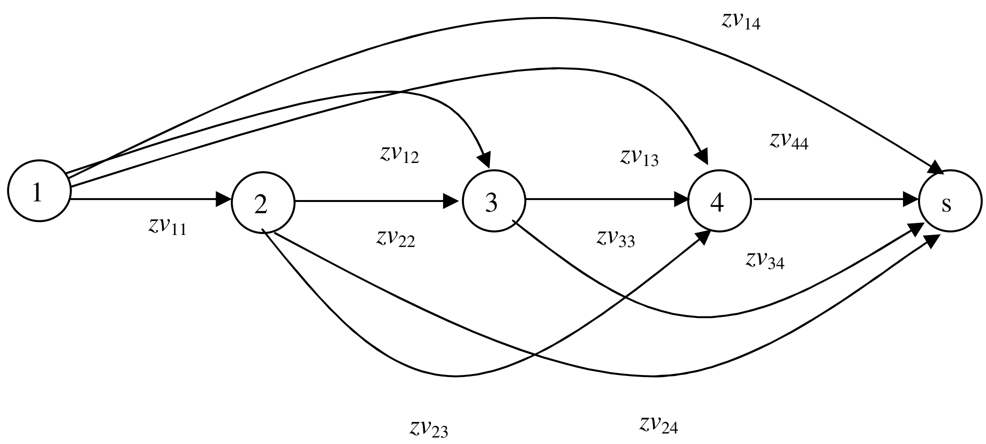


Figure 2.4 Shortest path network for a four period problem (Jans, 2002)

The single product uncapacitated lot sizing problem can be reformulated as follows:

$$Min\sum_{t=1}^{m} S_t Y_t + \sum_{t=1}^{m}\sum_{k=t}^{m} cv_{tk} zv_{tk} \tag{27}$$

$$s.t. \quad \sum_{k=1}^{m} zv_{1k} = 1 \tag{28}$$

$$\sum_{s=1}^{t-1} zv_{s,t-1} = \sum_{k=t}^{m} zv_{tk} \quad \forall t \in P, t \neq 1 \tag{29}$$

$$\sum_{t=1}^{m} zv_{tm} = 1 \tag{30}$$

$$\sum_{k=t}^{m} zv_{tk} \leq y_t \quad \forall t \in P \tag{31}$$

$$zv_{tk} \geq 0; y_t \in \{0,1\} \quad \forall t,k \in P; k \geq t \tag{32}$$

where PC_t is the production cost and cv_{tk} is defined as the total cost for producing the demands for period t until period k in period t (Jans, 2002). The according inventory cost is calculated as

$$cv_{tk} = PC_t sd_{tk} + \sum_{s=t+1}^{k}\sum_{u=t}^{s-1} h_u d_s \tag{33}$$

Constraints (28), (29) and (30) are the conservation of flow equalities for the shortest path network. Constraint (31) is the setup forcing constraint.

The SR reformulation is extended to the multi level case with capacity constraints by Tempelmeier and Helber (1994). Stadtler (1996, 1997) suggests an improved SR formulation which decreases the computational effort.

It should be noted that the LP relaxation of the SR reformulation and the SPL reformulation have identical objective function values (Denizel et al., 2008). While the number of decision variables is same in both reformulations, the number of constraints is more in the SPL reformulation.

2.4.1.3 Valid Inequalities

Another way to strengthen the bounds of the LP relaxation is to generate valid inequalities. Valid inequalities reduce the size of the solution space by cutting off the unpromising areas. There are three types of valid inequalities. The first one is named as "C*utting Plane Method*" which generates the valid inequalities dynamically to cut off current non-integer solutions. The second one is the "*Branch and Cut Method*" which introduces the valid inequalities during the course of a B&B algorithm. The last one is the "*Cut and Branch method*" which incorporates all valid inequalities into the model formulation before the execution of the B&B algorithm (Buschkühl et al., 2008).

The first valid inequalities for lot sizing problems are proposed by Barany et al. (1984). These inequalities describe the convex hull of the single item uncapacitated lot sizing problem and can also be applied to the CLSP. Pochet and Wolsey (1988) present strong valid inequalities for the case with backlogging. Pochet (1988), Leung et al. (1989), Pochet and Wolsey (1993), Miller et al. (2000) and Van Wyne (2003) derive several valid inequalities for the capacitated lot sizing problem and variants. Pochet and Wolsey (1991) review several inequalities for various models such as capacitated models, start ups and multi-level problems. Belvaux and Wolsey (2000) provide a general framework for modeling and solving lot sizing problems. This framework is called *bc-prod* system and includes preprocessing for lot sizing problems and generates lot-sizing specific cutting planes for a variety of lot sizing models. The work is extended in Belvaux and Wolsey (2001) where start-ups, changeovers and switch-offs are introduced into the modeling. Suerie and Stadtler (2003) present valid inequalities for the CLSPC both in single and multi levels.

2.4.2 Special-Purpose Heuristics

The CLSP is known for its computational complexity. Florian et al. (1980) have shown that the general case of the single-item CLSP is NP-Hard. When setup times are introduced into the multi-item CLSP, even the feasibility problem becomes NP-Complete (Trigeiro et al., 1989). Therefore, to deal with the combinatorial nature of the problem the trend in the lot sizing literature is to employ computationally efficient solution techniques such as heuristics. Several heuristics have been proposed for lot sizing problems with different modeling features. We classify them as simple, greedy, lagrangian, decomposition, aggregation and mathematical programming heuristics. The details of these heuristics are given in the following.

2.4.2.1 Simple Heuristics

These heuristics are often used as lot size rules in MRP systems instead of the Wagner-Whitin algorithm (Jans & Degraeve, 2007). These lot size rules can be named as economic order quantity, period order quantity, least period cost, least unit cost, part period balancing, least total cost etc. The definitions of these heuristics can be found in many text books on production planning such as Nahmias (2005). These heuristics are also named as uncapacitated dynamic lot sizing heuristics (Buschkühl et al., 2008).

These simple heuristics can be used in constructing an initial solution for the multi item capacitated lot sizing problem as in Eisenhut (1975) and Lambrecht and

Vanderveken (1979). A recent overview including comparisons of some of these heuristics can be found in Simpson (2001).

2.4.2.2 Greedy Heuristics

The second class in this group is greedy heuristics. These heuristics start from scratch and increase lot sizes successively to achieve cost savings working period-by-period or starting from an initial solution. During a run of the heuristic, the feasibility is checked and a cost criterion is used for minimizing the overall cost. There are two ways used for checking the feasibility of the problems. The first one is *feedback mechanisms* which push infeasible production quantities to earlier periods and the second one is *look-ahead mechanisms* which try to adjust production lots by looking at the future demands (Buschkühl et al., 2008).

Greedy heuristics which start from scratch are called *constructive greedy heuristics*. Most of these heuristics work period by period either forward or backward (Buschkühl et al., 2008). Dixon and Silver (1981), Doğramacı et al. (1981) and Gupta and Magnusson (2005) propose constructive greedy heuristics.

In contrast to starting from scratch, the second type of greedy heuristics starts from an initial solution. This type of heuristics is called *improvement greedy heuristics*. These heuristics try to generate a better feasible solution by shifting production lots forward or backward. Examples of these heuristics can be found in Günther (1987), Trigeiro (1989), Clark and Armentano (1995), França et al. (1997).

2.4.2.3 Lagrangian Heuristics

Lagrangian heuristics are solution approaches based on Lagrangian relaxation. The complicating constraints of an optimization problem are relaxed and put into the objective function with penalty costs (i.e. Lagrangian multipliers) in Lagrangian heuristics. At each step, with the given Lagrangian multipliers a lower bound is computed. A feasible solution is constructed and serves as the new upper bound.

Finally, the Lagrangian multipliers are updated (Buschkühl et al., 2008). Throughout the iterations, the lower and upper bounds start to converge with the updates in the Lagrangian multipliers. The most popular method in updating the Lagrangian multipliers is the subgradient optimization. The subgradient shows the direction of the search where the greatest possible improvement can be achieved.

In solving the CLSP by using Lagrangian heuristics, the most widely used procedure is to relax the capacity constraints. When capacity constraints are relaxed, the remaining problem is decomposed into the single level uncapacitated lot sizing problems. These problems can be solved by using different exact and heuristic approaches such as Wagner and Whitin (1958), Silver and Meal (1969, 1973), Groff (1979) and Wagelmans et al. (1992). One alternative is to relax the demand constraints (Diaby et al., 1992a; Jans & Degraeve, 2004; Süral et al., 2009). The other alternative is to relax the binary constraints for setup variables in the multi level capacitated lot sizing problem as in Chen and Chu (2003).

Trigeiro (1987) solves the CLSP without setup times and proposes a heuristic approach. Then, this work is extended by Trigeiro et al. (1989) for solving the CLSP with setup times. Campbell and Mabert (1991) propose a lagrangian heuristic similar to Trigeiro et al. (1989) for solving the CLSP with cyclic schedules in which the times between production periods of an item are constant. A different lagrangian heuristic is proposed by Sox and Gao (1999) for solving the CLSPC. The authors relax the capacity and setup carryover constraints in this approach. Tempelmeier and Derstoff (1993, 1996) present a solution approach based on Lagrangian relaxation for solving the multi level capacitated lot sizing problem with setup times and lead times. The backlogging for end items is added to the same model by Moorkanat (2000). Sambasivan and Yahya (2005) propose a lagrangian heuristic for solving the CLSP with the idea of multi plants. Several Lagrangian heuristics are proposed in Brahimi et al. (2006) for solving the CLSP with time windows.

Developing hybrid algorithms consisting of Lagrangian relaxation and meta-heuristics is a recent trend for solving the CLSP. A number of different approaches can be found in Özdamar and Barbarosoğlu (1999), Özdamar and Barbarosoğlu (2000) and Hindi et al. (2003). Özdamar and Barbarosoğlu (1999) combine Lagrangian relaxation with simulated annealing to solve a multi level production system with serial product structures for several end items in a parallel machine environment. A similar approach is proposed in Özdamar and Barbarosoğlu (2000) to solve the multi level capacitated lot sizing problem with general product structures. They propose two relaxation schemes. The first one is the hierarchical relaxation in which the capacity constraints are relaxed. The second one is the type of the relaxation scheme where capacity and inventory balance constraints are relaxed. Hindi et al. (2003) propose a solution approach combining Lagrangian relaxation and Variable Neighborhood Search.

2.4.2.4 Decomposition Heuristics

The idea of the decomposition heuristics is to divide the whole problem into smaller sub-problems and coordinate the schedules later (Buschkühl et al., 2008). Newson (1975a, 1975b) presents an item based heuristic approach for solving the CLSP without and with overtime, respectively. The heuristic ignores the capacity constraints first and decomposes the CLSP into single item problems which can be solved using WW algorithm. Another item based decomposition heuristic is proposed by Kırca and Kökten (1994) for solving the CLSP. They decompose the CLSP into single item problems and solve a single item problem with adjusted capacities and extra bounds on production and inventory to guarantee the feasibility of the whole problem.

To deal with the multi level capacitated lot sizing problem Tempelmeier and Helber (1994) decompose the whole problem into a sequences of single level CLSPs. Then, the CLSPs are solved using Dixon-Silver heuristic. The work is extended to cover the problems with setup times by Helber (1995).

2.4.2.5 Aggregation Heuristics

Aggregation heuristics reduce the problem size by omitting details first and breaking the solution down later (Buschkühl et al., 2008).

Özdamar and Bozyel (2000) propose to aggregate the demand of all products in a period. The authors avoid using binary setup variables explicitly. However, they use a setup allowance percentage, which reduces both the capacity and complexity of the MIP model. Then, lots that respect the aggregated lot sizes are determined via a filling procedure.

2.4.2.6 Mathematical Programming Heuristics

These heuristics are all based on exact methods which are truncated in some way to reduce the computational effort. Since these approaches are not transparent to the casual user, they have some distinctive advantages (Maes & Van Wassenhove, 1988). These techniques are based on standard mathematical programming techniques which are quite general and their application is not so much restricted to a specific problem formulation. Most methods can be generalized to problems involving several resources, overtime considerations, time-consuming set ups, time variable capacity absorption, etc. Fix-and-Relax heuristics, Rounding heuristics, LP based heuristics, Dantzig-Wolfe and Column Generation are examples of this class.

Fix-and-Relax heuristics: The logic behind the Fix-and-Relax heuristics is to divide the overall problem into several sub-problems and obtain the overall solution from these sub-problems. Dividing the overall problem helps to reduce the number of binary variables which also decreases the problem complexity. Three kinds of binary variables can be distinguished in these heuristics. The first ones are the binary variables which are to be optimized, the second ones are the relaxed variables and the third ones are the variables which are fixed to the values of a previous iteration.

Dillenberger et al. (1993) develop a Fix-and-Relax heuristic to solve the capacitated lot sizing problem with setup carryover where Dillenberger et al. (1994) apply the same heuristic to an extended model formulation. Stadtler (2003) uses a Fix-and-Relax heuristic with the idea of overlapping windows for solving the multi level capacitated lot sizing problems. The performance of the same heuristic is also tested in Suerie and Stadtler (2003) for solving the CLSPC. A different version of the Fix-and-Relax heuristic is proposed in Mercé and Fontan (2003) for solving the capacitated lot sizing problems with backorder options. Federgruen et al. (2007) present a different Fix-and-Relax heuristic for solving the capacitated lot sizing problem with joint setup costs and call it progressive interval heuristic. The heuristic starts with a small subset of periods (i.e. time window) and the binary variables are solved to optimality within this time window. In each iteration of the heuristic, the time window is extended while the binary variables of the last τ periods are solved to optimality. The binary variables related to the earlier periods are fixed and the heuristic stops when the end of the planning horizon is reached. A number of Fix-and-Relax heuristics can be found in Absi and Sidhoum (2007) to find feasible solutions for the CLSP with setup times and shortage costs. The hybridization of the Relax-and-Fix heuristic with a partitioning and sampling method is used for solving the CLSP in Wu et al. (2010).

Recently, another different version of the Fix-and-Relax heuristic, namely Fix-and-Optimize heuristic, is proposed for solving the multi level capacitated lot sizing problem (Sahling et al., 2009; Helber & Sahling, 2010). Unlike the Fix-and Relax heuristic which deals with three types of binary variables (i.e. relaxed, optimized, fixed) throughout the iterations, the Fix-and-Optimize heuristic deals with only two types of binary variables (i.e. fixed and optimized). Elimination of relaxed binary variables greatly reduces the computational effort of the Fix-and-Optimize heuristic. Further explanation about this heuristic is given in chapter three.

Rounding heuristics: The logic of the rounding heuristics is to solve the LP relaxation of the MIP model and round the fractional binary variables subsequently. However, for the CLSP, these solutions are generally infeasible, as capacity may not be sufficient. To deal with this issue, generally a threshold is used for rounding the fractional binary variables.

Maes et al. (1991) and Kuik et al. (1993) present a number of rounding heuristics for solving the multi level capacitated lot sizing problems without setup times. For the CLSP, Eppen and Martin (1987), Alfieri et al. (2002) and Denizel and Süral (2006) use several rounding heuristics. Recently, Akartunalı and Miller (2008) combine a rounding heuristic with a Fix-and-Relax heuristic for solving the multi level capacitated lot sizing problem. Gören et al. (2011) propose a novel initialization scheme for generating the initial population in GAs for solving the CLSPC. The proposed initialization scheme is a simple rounding heuristic which utilizes the solution of the LP relaxation of the CLSP.

LP based heuristics: In LP based heuristics, the setup variables are omitted and the resulting problem which is easier than the actual problem, is solved to optimality. In these heuristics, the complete setup pattern is either given or only implicitly accounted for, and then the remaining LP with the fixed setup patterns are solved to optimality. Harrison and Lewis (1996) present an iterative coefficient modification heuristic for solving the multi level capacitated lot sizing problem. Since each setup variable is linked to a corresponding continuous production variable, both variables (setup and production variables) must be zero or positive at the same time. Therefore, when the binary setup variables are omitted, this complex problem reduces to a LP. The setup times are accounted for implicitly via modification of the production time coefficients of the related production variables. In each iteration, the reduced LP is solved and production time coefficients are modified. Later, Katok et al. (1998) extend this work by introducing the cost balancing for the modification of the production time coefficients.

Hung and Hu (1998) present another LP based heuristic for solving the CLSP. The authors fix the setup pattern at each iteration, and solve the remaining LP. They use the information of the shadow prices relating to the capacity constraints following the solution of LP. Doing so helps to decide in which periods it is better not to have a setup.

Dantzig-Wolfe Decomposition and Column Generation: Dantzig-Wolfe decomposition is an algorithm for solving LPs with special structure. It is originally developed by George Dantzig and Phil Wolfe (Dantzig & Wolfe, 1960). The basic idea is to divide the problem into smaller sub-problems which are much easier to solve and coordinating mechanism ensuring that a good approximation for the overall problem (Jans & Degraeve, 2007) is obtained.

Dantzig-Wolfe decomposition has been applied to the CLSP which is modeled as a set partitioning problem (Buschkühl et al., 2008). The objective is to find optimum schedules for each item which do not violate the capacity constraint and lead to minimal costs. In order to solve the set partitioning problem, column generation is applied. Column generation is an efficient algorithm for solving large scale LPs. Column generation deals with two problems. The first one is the master problem and the second one is the sub-problem. The process in column generation begins with solving the master problem with a few columns (i.e. schedules for each product). The corresponding sub-problem is to find the single item schedules to feed into the master problem. At each iteration, a separate single item uncapacitated sub-problem for each product is solved. If the new column has a negative reduced cost, it is added to the master problem. In order to generate new columns, the master problem is solved again. Using the new dual prices, new columns are generated. This process continues until no new solution is found or the stopping criterion is reached. At the end of the process, a promising lower bound for the CLSP is obtained. In order to generate a feasible solution, an additional heuristic has to be applied. Manne (1958) proposes to use the column generation for solving the CLSP. Later, this work is extended by Dzielinski and Gomory (1965), who develop a column selection procedure to handle larger problems. However, there is a structural problem with the formulation proposed by Manne (1958). Bitran and Matsuo (1986) present error bounds for the solution obtained by Manne's formulation. Other approaches in this group can be found in Lasdon and Terjung (1971), Bahl et al. (1983), Cattrysse et al. (1990), Salomon et al. (1993), Degraeve and Jans (2007) and Huisman et al. (2003). Recently, Tempelmeier (2010) develops a heuristic solution approach based on column generation for solving the dynamic capacitated lot sizing problem with random demand.

2.4.3 Meta-heuristics

Meta-heuristics are computational methods that solve optimization problems in an iterative manner trying to improve a candidate solution. Meta-heuristics start with a given initial solution. There are two basic principles that determine the behavior of a meta-heuristic, namely intensification and diversification (Buschkühl et al., 2008). Diversification generally refers to the ability to visit many and different regions of the search space whereas intensification refers to the ability to obtain high quality solutions within those regions (Lozano & Martínez, 2010).

In recent years, the usage of meta-heuristics for solving CLSP and its extensions have become popular among researchers in lot sizing field. In this section, we investigated a number of meta-heuristics such as Tabu Search (TS), Simulated Annealing (SA), Genetic Algorithms (GA), Ant Colony Optimization (ACO), Variable Neighborhood Search (VNS) and Memetic Algorithm (MA) proposed for solving different variants of CLSP.

Tabu Search (Glover, 1986) is a mathematical optimization method which is based on local search. The main difference between TS and local search is that TS uses memory structures (i.e. tabu list) during the search. Tabu list is used to keep track of the parts in the search space that are already visited. Salomon et al. (1993) present a TS approach to solve the sub-problems of their column generation heuristic for solving the CLSP without setup times. Another TS approach for the same problem is presented in Hindi (1996). Gopalakrashinan et al. (2001) incorporate setup carryover into CLSP and they present a TS approach for the extended problem. Another extension to the CLSP is to consider backlogging. Hung et al. (2003) propose a TS approach to solve the CLSP with backlogging. The multi level capacitated lot sizing problem is also solved by using TS in Hung and Chien (2000) and Berretta et al. (2005).

Simulated Annealing (Kirkpatrick et al., 1983) is a meta-heuristic proposed for combinatorial optimization problems. It combines iterative improvement and random walk. The inspiration of SA comes from the annealing process in metallurgy, a technique involving heating and controlled cooling of a material to increase the size of its crystals and reduce their defects. SA starts with a non-optimal initial solution. At each step, a random neighbor solution is selected and the difference between the current best solution and random solution is calculated. If there is an improvement, the random neighbor solution is accepted as the current best solution and the search continues from that point. If not, the random neighbor solution may still be accepted according to a probability, which is based on annealing schedule. Salomon et al. (1993) present a SA approach for solving the CLSP without setup times. For solving the CLSP with setup times, Özdamar and Bozyel (2000) propose another SA approach. The CLSP with sequence dependent setup times and costs is solved by Mirabi and Ghomi (2010) in which a hybrid SA is proposed. There are also studies evaluating the performance of SA (Kuik & Salomon, 1990; Tang, 2004) for solving the multi level uncapacitated lot sizing problem. The multi level lot sizing problem can also be extended with capacity restrictions and SA has made important contribution to the solution of this difficult problem. Salomon et al. (1993), Helber (1995), Özdamar and Barbarosoğlu (1999), Hung and Chien (2000), Barbarosoğlu and Özdamar (2000) and Özdamar and Barbarosoğlu (2000) propose different SA approaches to deal with this problem.

Genetic Algorithm (Holland, 1975) is a population based heuristic method proposed for solving combinatorial optimization problems. GAs are modeled on natural evolution and inspired by the natural evolution process. It is based on the genetic process of biological organisms. Various applications of GAs to solve the CLSP with different modeling features can be found in Özdamar and Birbil (1998), Özdamar and Barbarosoğlu (1999), Özdamar and Bozyel (2000), Özdamar et al. (2002) and Xie and Dong (2002). A detailed review of the GA applications on lot sizing problems is presented in chapter four.

Variable Neigborhood Search (Hansen & Mlaydenović, 1999, 2001) is a meta-heuristic approach for solving combinatorial and global optimization problems whose basic idea is a systematic change of neighborhood within local search. At first, a neighboring solution is generated from incumbent solution. If this solution is better than the incumbent solution, the search moves in that direction. If not, another neighboring solution is generated from the incumbent solution. Hindi et al. (2003) propose a solution approach combining Lagrangian Relaxation and VNS to solve the CLSP. The proposed solution approach starts with Lagrangian Relaxation and this solution is set as an initial solution in the smoothing heuristic which is used to eliminate the capacity infeasibilities. The last solution is improved by using the VNS in the last step. Recently, Xiao and Kaku (2010) present an effective approach based on VNS for solving the multi level lot sizing problem. To improve the performance of the proposed approach, the authors use two different neighborhood strategies such as move at first improvement and move at best improvement.

Ant Colony Optimization (Dorigo et al., 1996, 1999) is inspired from the behavior of ant colonies. During food search, ants lay down pheromone trails. The other following ants choose the trail with the highest concentration of pheromone. The similar mechanism is observed in solving optimization problems using ACO. In ACO, a set of software agents called *artificial ants* search for good solutions to a given optimization problem and exchange information on the quality of these solutions. Pitakaso et al. (2006) combine ACO and exact methods and propose a hybrid approach for solving the multi level capacitated lot sizing problem. The idea of the hybrid approach is to decompose the problem into smaller ones and solve these smaller problems by exact methods. The decomposition in the hybrid approach is controlled by ACO. In another study, Pitakaso et al. (2007) present another ACO based hybrid approach for solving the uncapacitated multi level lot sizing problem. This approach starts with finding a good lot sizing sequence which is controlled by ants and then a modified WW algorithm is applied for each product in the sequence separately. Recently, Almeder (2010) combines ACO and an exact method for solving the multi-level capacitated lot sizing problem. The ACO determines the principal production decisions and then an exact method is used to calculate the corresponding production quantities and inventory levels.

Memetic Algorithm is an evolutionary algorithm similar to GA. It combines an evolutionary or population-based approach with separate individual learning or local improvement procedures. The first application of MA in lot sizing literature can be found in Berretta and Rodrigues (2004). In this study, the authors propose a MA for the multi level capacitated lot sizing problem based on the work of França et al. (1997).

2.5 Chapter Summary

In this chapter, basic concepts of lot sizing problem, variants of lot sizing problem and its solution methods were presented. Since lot sizing problem has an important role in production planning, a massive body of academic literature covers the lot sizing problems.

In recent years the application of meta-heuristics has taken the attention of the researchers in lot sizing area, since meta-heuristics provide an alternative to optimum seeking methods for solving NP-Hard combinatorial problems such as CLSP with setup carryover and backordering. Among these meta-heuristics, in this Ph.D. study, GAs are employed to solve the CLSP with setup carryover and backordering. To further improve the performance of GAs, a MIP based heuristic, namely Fix-and-Optimize heuristic is integrated into GAs in different ways. As a consequence, novel GA-based hybrid meta-heuristic approaches are proposed for solving the CLSP with setup carryover and backordering.

The detailed information regarding GAs and Fix-and-Optimize heuristic and also the concept of hybridization are presented in the next chapter.

CHAPTER THREE
BACKGROUND INFORMATION FOR PROPOSED SOLUTION METHODOLOGIES

3.1 Introduction

This study proposes a number of novel hybrid approaches for solving the capacitated lot sizing problem with setup carryover and backordering. These hybrid approaches combine Genetic Algorithms (GAs) with a MIP based heuristic, namely Fix-and-Optimize heuristic. To clarify the contribution of these proposed hybrid methodologies, first some general information on GAs and Fix-and-Optimize heuristic are presented in the following two sections, next the concept of hybrid meta-heuristics is explained and finally the context of this chapter is summarized in Section 3.5.

3.2 Genetic Algorithms

First pioneered by John Holland in 1975, Genetic Algorithms have been widely studied, experimented and applied in many fields. Many of the real world problems involve finding optimal parameters, which might prove difficult for traditional methods but ideal for GAs (De Jong, 1993). GAs have been successfully adapted to solve several combinatorial optimization problems in the literature and have become increasingly popular among meta-heuristic approaches for finding optimal or near optimal solutions in a reasonable time. GAs are modeled on natural evolution and inspired by the natural evolution process. It is based on the genetic process of biological organisms. Over many generations, natural populations evolve according to the principle of “natural selection” and “survival of the fittest”. By mimicking this process and by suitable coding, GAs make the solution evolve and approach the best. Genetic operators manipulate individuals in a population over several generations to improve their fitness.

Unlike simulated annealing and tabu search, GAs use a collection of solutions, from which, using selective breeding and recombination strategies, better and better solutions can be produced.

The steps that should be taken in application of the GAs can be stated as follows (see Figure 3.1):

1. Choice of a representation scheme for a possible solution (coding or chromosome representation.)
2. Decision on how to create the initial population.
3. Definition of the fitness function.
4. Definition of the genetic operators to be used (reproduction, mutation, crossover, elitism).
5. Choice of the parameters of the GAs such as population size, probability of applying genetic operators.
6. Definition of the termination rule.

To start the search GAs are initialized with a population of individuals. The individuals are encoded as chromosomes in the search space. GAs use mainly two operators namely, crossover and mutation to direct the population to the global optimum. Crossover allows exchanging information between different solutions (chromosomes) and mutation increases the variety in the population. After the selection and evaluation of the initial population, chromosomes are selected on which the crossover and mutation operators are applied. Next the new population is formed. This process is continued until a termination criterion is met.

3.2.1 Basic Concepts of GA

In order to understand the philosophy of GAs, the basic concepts should be defined. These concepts include coding of solution, population, fitness function, selection scheme, genetic operators (mutation and crossover), survival scheme and termination criteria.

In GA terminology, a solution is an *individual* or a *chromosome*. Chromosomes consist of discrete units called genes, which control one or more features of a chromosome. In the original implementation of Holland, a binary string is used for representing a chromosome (see Figure 3.2). However, various chromosome representations have been used in many GA applications so far.

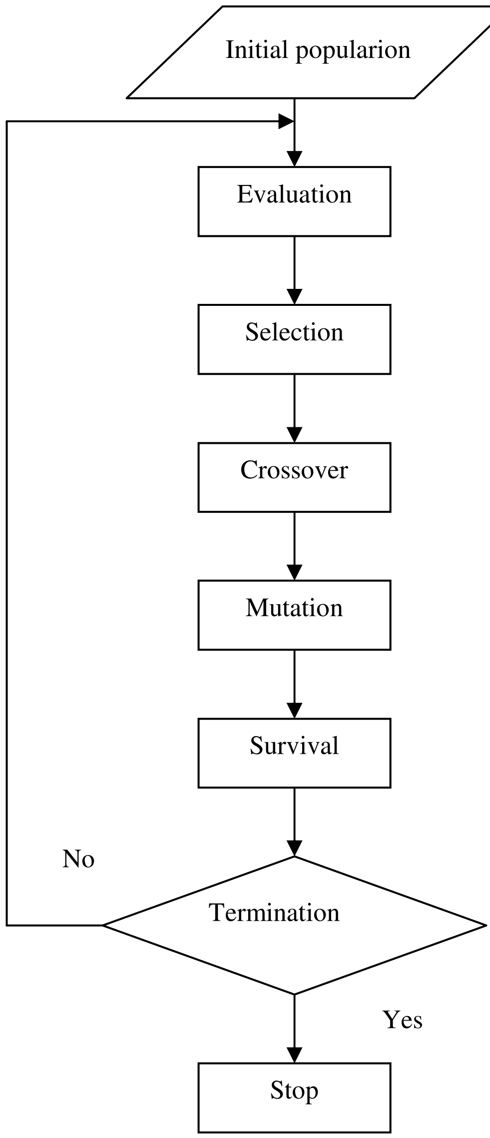


Figure 3.1 Flowchart of a simple GA

1	1	0	1	0	0	1	0	1	1

Figure 3.2 Binary chromosome representation

Hence, a chromosome corresponds to a unique solution in the solution space. This requires a mapping mechanism between the solution space and the chromosomes. This mapping is called an *encoding*. In fact, GA works on the encoding of a problem, not on the problem itself. The use of an inappropriate coding scheme has been the cause of many GA failures (Taşan, 2007).

Unlike single search approaches such as Tabu Search or Simulated Annealing, GAs work with a group of chromosomes called *population*. It is well known in the literature that the performance of GAs is affected by the quality of *initial population.* In most of the GA applications in the literature, the initial population is created randomly in order to improve the diversity of the population. When no priori knowledge exists for assessing the performance of the algorithm, random initial population generation method is usually preferred to others. Using problem specific information is another method to form an initial population. In this case, GA starts with a set of solutions (i.e. chromosomes) satisfying problem specific requirements (Pham & Karaboga, 2000).

After forming the initial population, each chromosome in the population is evaluated using an appropriate *fitness function*. The *fitness function* is used to evaluate and rate the performance of a chromosome. The fitness of a chromosome is a real number which forms the basis of the selection process.

Selection involves selecting the chromosomes which will go under the genetic operators. The aim of the *selection* is to reproduce more copies of chromosomes which have higher fitness values. Therefore, the *selection* process has an important role on driving the search towards a promising area and finding good solutions in a short time. The two most widely used selection methods are the *roulette wheel* and *tournament*.

In *roulette wheel selection*, the fitness values of chromosomes represent the widths of slots on the wheel. The algorithm for the *roulette wheel* can be summarized as follows (Coley, 2003):

- Sum the fitness of all the population members. Call this sum f_{sum}.
- Choose a random number, R_s, between 0 and f_{sum}.
- Add together the fitness of the population members (one at a time) stopping immediately when the sum is greater than R_s. The last individual added is the selected individual and a copy is passed to the next generation.

Basically, *tournament selection* involves running several tournaments among a few chromosomes chosen randomly from the population. The winner of each tournament (the one with the best fitness) is selected for crossover.

To generate new chromosomes from the existing ones, GA uses two kinds of genetic operators: *crossover* and *mutation*. The *crossover* operator which makes the GA different from other algorithms, is used to create two new chromosomes, namely offsprings from two existing chromosomes (i.e. parents). A number of *crossover* operators have been proposed but some common ones are one-point, two-point and uniform crossover. In one point crossover, the simplest version of the crossover operator, all genes to the left of the crossover point are exchanged. Crossover points are usually randomly determined. Figure 3.3 explains one-point crossover operator.

Unlike crossover, *mutation* is a monadic operation which means that it is generally applied at the gene level. In *mutation*, all chromosomes in the population are checked gene by gene and the values of the genes are changed according to a specified rate, known as the *mutation* rate. The *mutation* operator forces the algorithm to search new areas in the solution space. Like crossover operator, many versions of this operator can be found in the literature. Figure 3.4 shows the simplest version of it, single bit flip mutation.

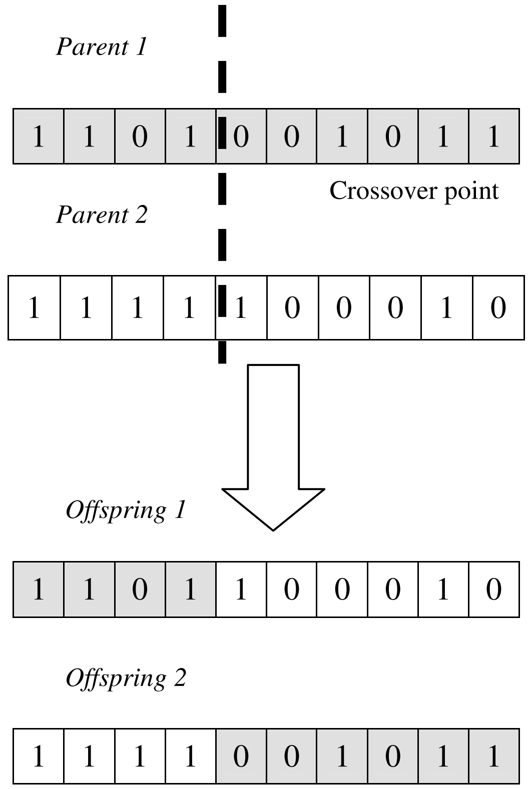


Figure 3.3 One point crossover

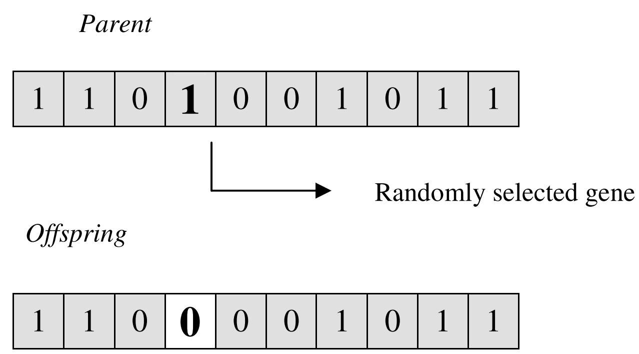


Figure 3.4 Single bit flip mutation

Survival schemes are related to the creation of the next generation. Survival involves selection of chromosomes for the next generation. In the simplest version of the GA, the child replaces the parents. However, in literature there are many variants of this rule. The most commonly used survival scheme is *elitism*, which makes survival of some number of the best individuals at each generation; hence guaranteeing that the final population contains the best solution ever found.

For *termination criteria*, various stopping conditions are proposed. The most widely used ones are to stop the algorithm after a fixed number of generations, or after a given time period. Another option is to stop the search when there is no change in the average or best fitness value in the population for a number of generations.

3.2.2 Identifying Efficient GA Control Parameters

As given in earlier sections, GA has several parameters including crossover, mutation rate and population size, and it is well known in the literature (Boyabatlı & Sabuncuoğlu, 2004) that the performance of GAs is affected by the values of these parameters. Hence, an important decision faced in many GA applications is to identify efficient GA control parameters. While inappropriate choice of GA parameters can lead to inferior performance and slow down the convergence, good values might cause the algorithm converge to the best result in short time. As a result, identification of the efficient GA parameters plays a crucial role on the quality of solution and convergence speed of GA application. However, choosing the right parameter values is a time consuming task.

Eiben et al. (1999) classify the parameter setting efforts in two classes as *parameter tuning* and *parameter control*. *Parameter tuning* is a widely used approach in which the good values for the parameters are determined before the genetic search starts. Then, the search starts with these parameters and these parameters are fixed during the search. As an alternative approach, *parameter control* is quiet different from *parameter tuning* as it starts the search with some initial parameter values and these values are changed during the run.

3.3 The Fix-and-Optimize Heuristic

The Fix-and-Optimize heuristic is a MIP based heuristic in which a sequence of MIPs is solved over all real-valued decision variables and a subset of the binary variables. The Fix-and-Optimize heuristic is an improved version of the relax-and-fix heuristic and is originally named as *Exchange* by Pochet and Wolsey (2000, pg. 113). The numerical effort required to solve the MIP model is mostly affected by the number of binary setup and setup carryover variables rather than the number of real-valued variables. The idea in the Fix-and-Optimize heuristic is to solve a series of smaller problems in a systematic manner. In each iteration of the algorithm, one problem is solved by setting most of the binary setup and carryover variables to fixed values. This reduction leads to a limited number of non-fixed binary variables which are optimized for a given problem. Then the problem is solved using a standard MIP solver. In the next iteration, there is a new problem with a different subset of fixed binary variables and the rest of the binary variables are optimized. However, in each problem the complete set of real-valued decision variables is considered (Sahling et al., 2009).

The Fix-and-Optimize heuristic is originally implemented in Sahling et al. (2009) for solving the multi level CLSP with setup carryover and later in Helber and Sahling (2010) for solving the multi level capacitated lot sizing problem.

3.3.1 The Algorithm

The algorithm needs an initial solution to start and goes through the problems defined by different types of decomposition schemes either once ($\ell = \ell_{\max}$) or until it reaches a local optimum. A problem is defined based on the type of decomposition scheme which yields a temporary solution.

Each temporary solution to a problem yields an objective value of Z which is at least as good as the current best solution (Z^{old}). So, a new solution is only accepted if it yields an objective value better than the current best solution.

There are two issues affecting the quality of the solution obtained by the Fix-and-Optimize heuristic:

- How to generate the initial solution?
- What should be the number of iterations ($\ell_{\max}$)?

The basic outline of the algorithm is explained in the following.

Input: The initial solution.

Set the initial solution as the best solution. The objective value of the best solution is named as Z^{old}.

repeat

$l = 1$ (the iteration number)

for each problem (determined by the type of the decomposition)

determine the set of the binary variables that will be optimized and fixed.

Fix the determined binary values to the values based on the best solution and optimize the rest.

Solve the problem, compute Z^{new} and obtain a temporary solution.

if Z^{new} is better than Z^{old}

then

$Z^{old}=Z^{new}$

best solution= temporary solution

end if

else

Keep the best solution and Z^{old}

end for

$l = l + 1$

until $l = l^{\max} \quad or \quad Z^{new} \geq Z^{old}$

Output: The best solution after the Fix-and-Optimize heuristic.

Figure 3.5 The algorithm of the Fix-and-Optimize heuristic

3.4 Hybrid Meta-heuristics

Meta-heuristics have received a lot attention of the researchers in the field of combinatorial optimization in recent years. However, as the problem size and complexity increases, the probability of finding good solutions by meta-heuristics decreases. Therefore, the trend in recent years is to combine meta-heuristics with other solution techniques in order to find better solutions in shorter time. These combined approaches are commonly referred as *hybrid meta-heuristics*. The motivation behind hybridization of different algorithms is to obtain a better performing system that exploits and unites advantages of the individual pure strategies (Raidl, 2006). In the past 10 years the focus of research on meta-heuristics has moved from an algorithm-oriented point of view to a problem-oriented point of view. In other words, today the main motivation for the researchers is to solve the problem under consideration as best as possible. This leads to designing more powerful hybrid algorithms which combine strong properties of different algorithms. However, the key issue in designing a hybrid algorithm is to decide which components to hybridize in order to create an effective algorithm (Caserta & Voss, 2009). Effective decision making in this area will greatly affect the performance of hybrid algorithms in solving difficult problems.

3.4.1 Classification of Hybridization

As a result of surveying current relevant literature it is noted that hybrid meta-heuristics are classified from different perspectives (Preux & Talbi, 1999; Talbi, 2002; Raidl, 2006). In this study, the classification given by Raidl (2006) is presented to illustrate the various classes and properties used for the categorization of hybrid meta-heuristics.

As seen in Figure 3.6, hybrid meta-heuristics can be formed by integrating meta-heuristics with different meta-heuristic strategies, or with problem-specific algorithms such as special simulations. Exact approaches like branch-and-bound, dynamic programming and various specific LP techniques and soft computation techniques like neural networks and fuzzy logic are also successfully combined with different meta-heuristics to form different types of hybrid meta-heuristics. Among these, *matheuristics,* which exploit mathematical programming techniques in (meta) heuristic frameworks, have attracted the attention of the researchers in recent years (Caserta & Voss, 2009).

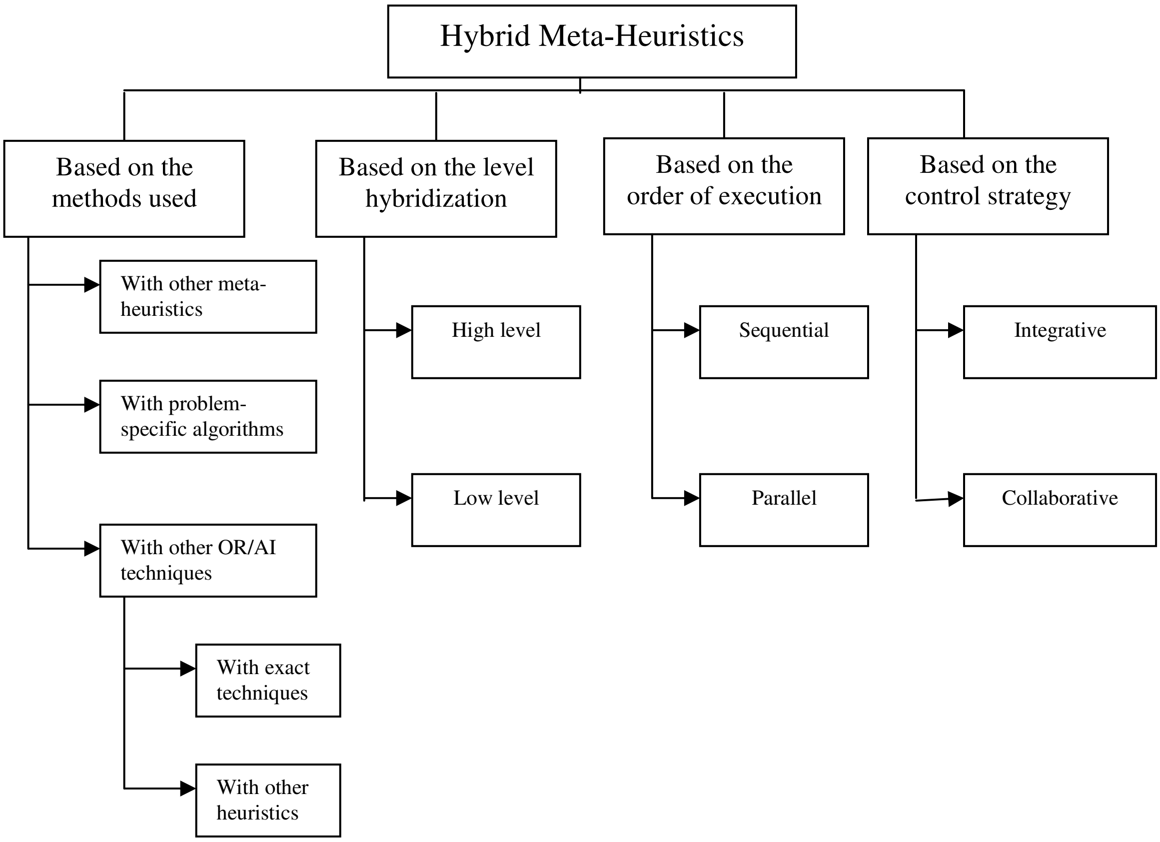


Figure 3.6 Classification of hybrid meta-heuristics (Raidl, 2006)

Another criterion which is used in classifying hybrid metaheuristics is the level at which different algorithms are combined. The identities of the algorithms in high-level combinations remain unchanged. There is no direct and strong relationship between the algorithms. However, in low-level combinations, the algorithms depend on each other and some components or functions of the algorithms are exchanged (Raidl, 2006).

Based on the order of execution, hybrid meta-heuristics are classified as *sequential* and *parallel* (Preux & Talbi, 1999). In *sequential hybridization*, one algorithm is performed after another; hence the output of the first algorithm is the input to the second algorithm. Unlike *sequential hybridization,* in *parallel* hybrid meta-heuristics, algorithms interact in more sophisticated ways.

The last group categorizes hybrid meta-heuristics as *collaborative* and *integrative* based on the control strategy used. In *integrative approaches*, one algorithm is embedded in another algorithm. In *collaborative approaches*, algorithms exchange information but are not part of each other (Raidl, 2006).

3.4.2 Hybridization of GA

The motivation of hybridizations of GAs and other solution approaches is to obtain better search algorithms that unite the advantages of the individual pure strategies (Raidl, 2006). Although GAs can rapidly locate the region in which the global optimum exists, they take a relatively long time to locate the exact local optimum in the region of convergence (Preux & Talbi, 1999). There are many other, more efficient, traditional algorithms for climbing the last few steps to the global optimum. This implies that using a GA to locate the hills and a traditional technique to climb them might be very powerful optimization technique (Coley, 2003). Goldberg (2002) states that hybridization is an important efficieny enhancement technique of GA. Incorporating a search method within a GA can improve the search performance on the condition that their roles cooperate to achieve the optimization goal (El-Mihoub et al., 2006).

The most popular form of hybrid GAs is to incorporate a local optimization method as an add-on extra with the pure GA loop. A combination of a GA and a local search method can speed up the search to locate the exact global optimum. In such a hybrid, applying a local search to the solution or solutions that are guided by a GA to the most promising region can accelerate convergence to the global optimum (El-Mihoub et al., 2006).

3.5 Chapter Summary

In this chapter, basic concepts of GA and Fix-and-Optimize heuristic are presented in detail. Unlike most conventional heuristic methods and some of the meta-heuristics (i.e. Tabu Search and Simulated Annealing) which conduct single directional search, GA performs multiple directional searches using a set of candidate solutions (Gen & Cheng, 1997). Moreover, no domain knowledge is required in GAs and stochastic transition rules are used during the search. It is well-known that GAs are better in finding promising areas in the search space. However, they are not good at locating the minimum or maximum of these points in large complex search spaces and can easily get stuck at local optimas. Therefore to avoid the premature convergence to local optima and hence, to further improve the performance of proposed GAs in solving lot sizing problem with setup carryover and backordering in this study, we concentrated on hybridization of GAs with a MIP based heuristic, namely the Fix-and-Optimize heuristic.

Next chapter is devoted to the review of the relevant literature. Since the focus in this study is on the application of GAs for lot sizing problems, we mainly focus on studies proposing GAs for solving lot sizing problems with different modeling features.

CHAPTER FOUR
LITERATURE REVIEW: APPLICATIONS OF GENETIC ALGORITHMS IN LOT SIZING

4. 1 Introduction

Lot sizing problems have attracted the attention of many researchers. There are a number of survey studies on lot sizing. De Bodt et al. (1984) discuss the state of the art of lot sizing under dynamic demand conditions, the impact of the use of rolling horizon and the influence of demand uncertainty on lot sizing decisions. Bahl et al. (1987) classify the lot sizing problems based on the demand type and resource constraints and concentrate on capacity dimensions of the production planning problem. Kuik et al. (1994) discuss the impacts of lot sizing and production planning at different decision levels in an organization. The basis of this review is a distinction of lot sizing issues related to process design/choice, activity planning and activity control. Wolsey (1995) reviews the history of the single item uncapacitated lot sizing problems by various solution algorithms, extensions and important reformulations. Drexl and Kimms (1997) present a classification on different variants of lot sizing and scheduling problems for both discrete and continuous time models. Belvaux and Wolsey (2001) show how to model the basic lot sizing problems including different extensions such as backordering, start up costs etc. and present some computational results for various sets of problems. Karimi et al. (2003) review the studies employing exact and heuristic approaches to solve the single level capacitated lot sizing problems. Brahimi et al. (2006) review the single item lot sizing problems under uncapacitated or capacitated situations. Jans and Degraeve (2008) give an overview of recent developments in the field of modeling single level dynamic lot sizing problems. Complexity theory and computational experiments indicate that most lot sizing problems are hard to solve (Jans & Degraeve, 2007). To deal with the complexity and find optimal or near-optimal results in reasonable computational time, in recent years, a growing number of researchers have employed heuristic approaches to solve lot sizing problems (Afentakis, 1987; Tempelmeier & Helber, 1994; Gopalakrashinan et al., 2001; Tang, 2004; Karimi et al., 2006; Pitakaso, 2007). Among these heuristic approaches, evolutionary computation has received increasing attention. The most well known evolutionary computation method is Genetic Algorithms (GAs). GAs are optimization techniques that use the principles of evolution and heredity to arrive at near optimum solutions to difficult problems (Khouja et al., 1998). GAs have been employed to solve different optimization problems across various disciplines due to their flexibility and simplicity.

During the literature survey, we noted two review studies (Aytuğ et al., 2003; Jans & Degraeve, 2007) discussing applications of GAs to lot sizing problems. Aytuğ et al. (2003) discuss different applications of GAs in the broad field of production and operations management problems by analyzing over 110 papers, including some lot sizing problems, while Jans and Degraeve (2007) review the recent literature employing a variety of meta-heuristics and other solution approaches (dynamic programming, Dantzig-Wolfe decomposition, Lagrange relaxation) to solve the dynamic lot sizing problem. Unlike these two studies which review the relevant literature from a broader perspective, in this chapter we will particularly focus on lot sizing problem with many different features and discuss how GAs are applied to these various lot sizing problems.

Considering the large number of studies in this area, to better highlight the research gaps the current literature was reviewed from two different perspectives: 1. The specifications of the lot sizing problems, 2. The features of the proposed GAs to deal with these problem specifications (Gören et al., 2010). Figure 4.1 presents the structural framework for reviewing.

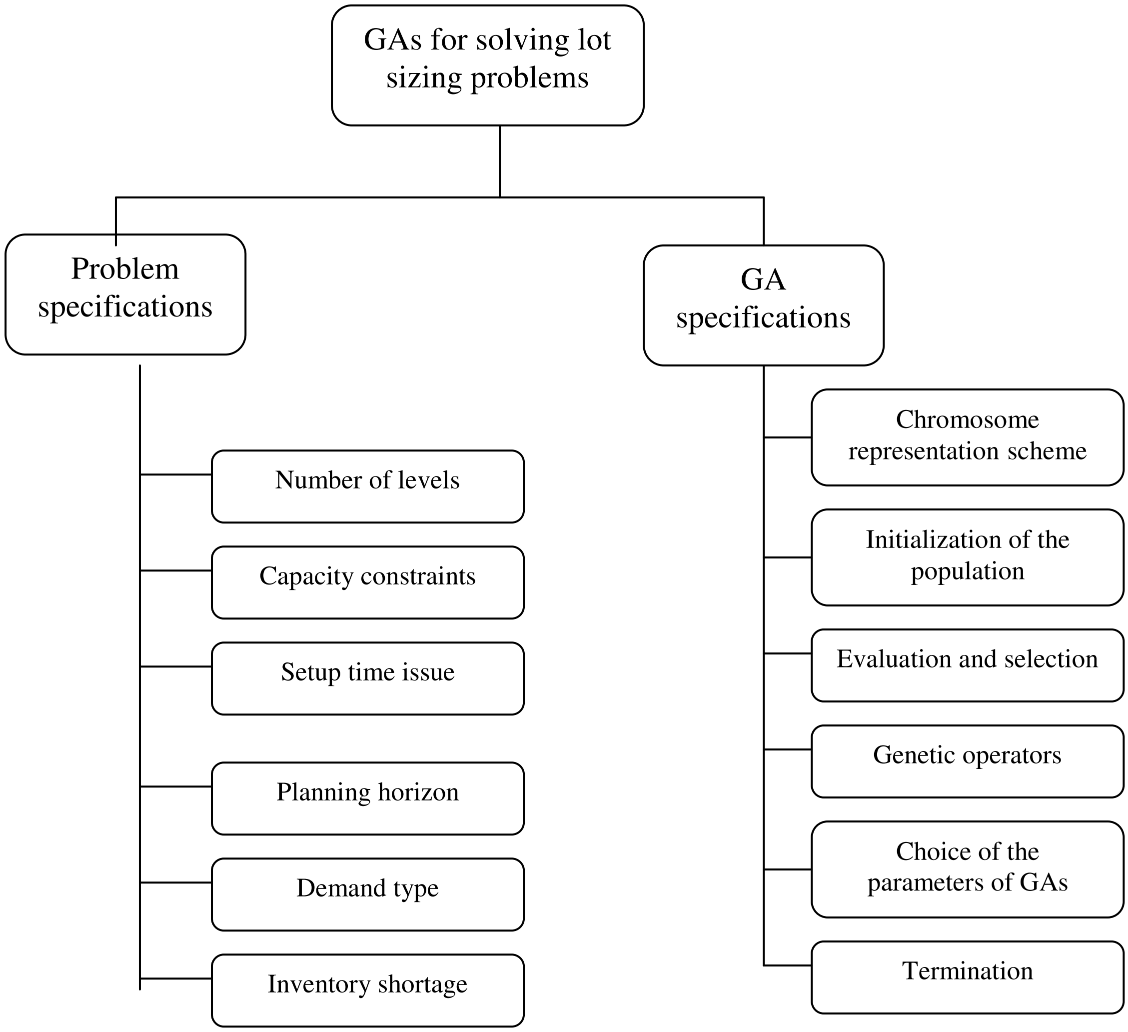


Figure 4.1 Structural framework for reviewing

As seen in Figure 4.1, problem specifications contain the main features of the lot sizing problem mentioned in chapter two. GA specifications summarize information about the chromosome representation, initialization of the population, evaluation, selection, genetic operators, choice of the GA parameters and termination criteria.

This chapter is organized as follows. Using this framework, in section 4.2, we focused on the problem specifications of the published literature in chronological order, in section 4.3; we analyzed the GAs proposed solving lot sizing problems with different modeling features. Finally, in section 4.4, the context of this chapter is summarized along with the concluding remarks.

4.2 Problem Specifications

This study reviews the literature published after 1995 based on the number of levels, capacity constraints, setup time issue, planning horizon, demand type and inventory shortage. Table 4.1 chronologically lists the recent published literature based on the problem specifications.

4.2.1 Research on Single Level Uncapacitated Lot Sizing Problems

The main issue in single level uncapacitated lot sizing problems is to determine production lot sizes for the planning horizon so that the sum of setup, inventory holding and production cost is minimized.

The first application of GAs to a single item, single level lot sizing problem without backordering appears in Hernandez and Süer (1999). The authors employ scaling in the fitness function to give higher reproduction probabilities to those chromosomes that represent better solutions. To evaluate how different aspects such as the population size, reproduction probability and scaling affect the results, they carry out various sets of experimental studies and state that a higher scale factor increases the chance of obtaining better solutions.

Van Hop and Tabucanon (2005) present a new adaptive GA for uncapacitated lot sizing problem in single level. The authors encode the timing of the replenishment as a chromosome. During the evaluation, the rates of GA operators such as mutation, selection and crossover for the next generation are automatically adjusted based on the rate of survivor offsprings. The proposed procedure gives faster and better results than using the static rates for the GA operators.

Gaafar (2006) applies GAs to the deterministic time-varying lot sizing problem with batch ordering and backorders. The author proposes a new coding scheme for the batch ordering policy. A comparative study with the modified Silver-Meal heuristic indicates that the GAs outperform over the modified Silver-Meal heuristic.

4.2.2 Research on Single Level Capacitated Lot Sizing Problems

The presence of capacity constraints increases the complexity of lot sizing problems. The first study in this group belongs to Özdamar and Birbil (1998) in which the authors propose a hybrid approach combining TS, SA and GAs to solve capacitated lot sizing and loading problem in single level with setup and overtimes. The hybrid approach integrates a GA into a local search scheme which incorporates features from TS and SA. The approach starts with GAs and when the population tends to get stuck to the same area of the feasible/infeasible solution space, TS/SA procedure carries out the local search on randomly selected chromosomes in the current population. Computational results show that the proposed hybrid heuristic is efficient and has high potential in solving different complex problems in production planning and control.

Khouja et al. (1998) use GAs in solving ELSP. The authors propose different binary representations, crossover methods and initialization methods in order to identify the best settings and the results of comparative experiments yield good results.

Table 4.1 The published literature based on the problem specification

Literature reviewed	PROBLEM SPECIFICATIONS										
	Number of levels		Capacitated	Setup time issue	Planning horizon			Demand Type			Backordering
								Deterministic Demand		Probabilistic Demand	
	Single Level	Multi level			Infinite	Finite	Rolling	Static Demand	Dynamic Demand		
Xie and Dong (1995)		*				*			*		
Khouja et al. (1998)	*		*	*	*			*			
Özdamar and Birbil (1998)	*		*	*		*			*		
Hernandez and Suer (1999)	*					*			*		
Kimms (1999)		*	*			*			*		
Özdamar and Barbarosoğlu (1999)		*	*	*		*			*		*
Hung et. al. (1999)	*		*	*		*			*		*
Kohlmorgen et al. (1999)	*		*			*			*		
Özdamar and Bozyel (2000)	*		*	*		*			*		
Dellaert and Jeunet (2000)		*				*			*		

Table 4.1 The published literature based on the problem specification (cont.)

Literature reviewed	**PROBLEM SPECIFICATIONS**										
	Number of levels		Capacitated	Setup time issue	Planning horizon			Demand Type			Backordering
								Deterministic Demand		Probabilistic Demand	
	Single Level	Multi level			Infinite	Finite	Rolling	Static Demand	Dynamic Demand		
Dellaert et al. (2000)		*				*			*		
Hung and Chien (2000)		*	*	*		*			*		*
Prasad and Chetty (2001)		*				*	*	*	*		
Sarker and Newton (2002)	*		*		*			*			
Özdamar et al. (2002)	*		*	*		*			*		
Moon et. al. (2002)	*		*	*	*			*			
Xie and Dong (2002)		*	*	*		*			*		
Duda (2005)	*		*			*			*		*
Yao and Huang (2005)	*		*	*	*			*			
Hop and Tabucanon (2005)	*					*			*		
Chang et. al. (2006)	*		*	*	*			*		Fuzzy	

Table 4.1 The published literature based on the problem specification (cont.)

Literature reviewed	PROBLEM SPECIFICATIONS										
	Number of levels		Capacitated	Setup time issue	Planning horizon			Demand Type			Backordering
								Deterministic Demand		Probabilistic Demand	
	Single Level	Multi level			Infinite	Finite	Rolling	Static Demand	Dynamic Demand		
Kämpf and Köchel (2006)	*		*	*		*				*	*
Megala and Jawahar (2006)	*		*			*			*		*
Moon et al. (2006)	*		*	*	*			*			
Gaafar (2006)	*					*			*		*
Li et al. (2007)	*		*			*			*		
Chatfield (2007)	*		*	*	*			*			
Jung et al. (2007)		*	*			*			*		
Duda and Osyczka (2007)	*		*			*			*		*
Fakhrzad and Zare (2009)		*	*	*		*			*		
Rao et al. (2009)	*		*	*	*					Fuzzy	
Sun et al. (2009)	*		*	*	*			*			
Santos et al. (2010)	*										
Mohammadi and Ghomi (2011)	*		*	*			*		*		
Gören et al. (2011)	*		*	*		*			*		
Gonçalves and Sousa (2011)	*		*	*	*			*			*

In another study, Kohlmorgen et al. (1999) deal with the CLSP using island GA. Potential solutions are genetically represented by a string of real values. The empirical study shows that the parallel GA achieves the same solution quality with the heuristic proposed by Kırca and Kökten (1994). Hung et al. (1999) use evolutionary algorithms for production planning with setup decisions. Firstly, to generate new chromosomes traditional GAs are used with the conventional crossover and mutation operators. Secondly, the GA modified with the sibling operator and the conventional reproduction operators are used to produce new chromosomes. Lastly, a sibling evolution algorithm using the sibling operator is employed to reproduce. Experimental studies show that the sibling evolution algorithm performs the best among all the algorithms used.

Özdamar and Bozyel (2000) propose three heuristic approaches including the hierarchical production planning approach, GAs and SA to solve the capacitated family lot sizing problem with setup time and overtime decisions for the single level case. The computational results show that GAs provide good performance only in small sized populations and the SA outperforms the others irregardless of the population size.

Sarker and Newton (2002) present a GA approach to determine a purchasing policy for raw materials of a firm under a limited storage space and transportation fleet of known capacity. In order to deal with the capacity constraints, three different penalty functions namely static, dynamic and adaptive penalties, are proposed. Experimental results show that the performances of all three proposed penalty functions are similar.

Özdamar et al. (2002) reconsider the capacitated lot sizing problem with overtime and setup times given in a previous study (Özdamar & Bozyel, 2000). Unlike the earlier study where transportation type of presentation and three different solution approaches (Hierarchical Production Planning, GAs, SA) are used, in this study the authors prefer using the direct coding and a hybrid approach consisting of GAs, TS and SA. The computational results show that the proposed hybrid approach is capable of finding good solutions in reasonable computational time.

Moon et al. (2002) develop a hybrid GA for the ELSP. The proposed hybrid algorithm integrates GAs with Dobson's heuristic, which has been regarded as the best in its performance for the economic lot size scheduling problem in literature. The hybrid GA outperforms Dobson's heuristic.

Yao and Huang (2005) solve the Economic Lot Size Scheduling Problem with deteriorating items using the basic period approach under power of two heuristic. The study presents a hybrid GA with a feasibility testing procedure and a binary search heuristic to efficiently solve the problem. The computational results show that the hybrid approach can be very helpful to derive the production scheduling and lot sizing strategies for deteriorating items efficiently in the food industry.

Duda (2005) presents a GA approach with repair functions for the classical DLSP originating from a real production environment, in single level where multi items are produced. The author employs three variants of GA each using special crossover, mutation operators and repair functions.

Kämpf and Köchel (2006) combine GAs with simulation for the stochastic capacitated lot sizing problem. The problem involves defining a manufacturing policy consisting of a sequencing rule and a lot size rule, which maximizes the expected profit per time unit. In this study, the sequencing and lot sizing decisions are represented in a chromosome as an individual. Each individual representing a chromosome is then evaluated by using simulation under different sequencing policies such as First Come First Serve, Random and Cyclic. Experimental studies show that the proposed approach can be applied to arbitrary system structures and control policies.

Megala and Jawahar (2006) study the single item dynamic lot sizing problem using GAs and Hopfield neural network under capacity constraints and discount price structure. The authors carry out a comparative study and note that while GAs provide either optimal or near optimal solutions in most of the cases, Hopfield neural network produces satisfactory results for only small sized problems.

Chang et al. (2006) present a fuzzy extension of the economic lot-size scheduling problem for fuzzy demands. The problem is formulated via the power-of-two policy and basic period approach which allows different items to have different cycle lengths restricting each product's cycle time to be an integer multiple k of a time period called basic period (Moon et al., 2002). GAs are employed with triangular fuzzy numbers in order to find the cycle time and start time of the production.

Moon et al. (2006) apply Group Technology (GT) principles to the ELSP. GT is an approach to manufacturing and engineering management that seeks to achieve the efficiency of high speed and mass production by identifying similar parts and classifying them into groups based on their similarities. The GT approach often has many benefits in manufacturing systems such as shortened setup times, reduced work-in-process inventory, less material handling, and better production planning and control. The authors modify the heuristic proposed by Kuo and Inman (1990) by considering the modified cycle length and propose a hybrid GA to solve the ELSP. The computational results show that the proposed hybrid heuristic outperforms the heuristic of Kuo and Inman (1990).

In another study, Li et al. (2007) analyze a version of the capacitated dynamic lot-sizing problem with substitutions and return products using GAs. The authors first identify the periods requiring setups by applying a GA then they develop a dynamic programming approach to determine the number of new products to be manufactured or the number of return products to be remanufactured in each of these periods.

The performance of pure GA is tested in Chatfield (2007) for the ELSP. The author creates a binary encoding scheme for chromosome representation and applies it to a benchmark problem in the literature. The results are impressive such that some of them are the best ones up-to date.

Duda and Osyczka (2007) develop a GA for solving the DLSP with a capacity loading criterion. The authors include the technological constraints into the optimization model directly and test the performance of the GA on real world problems.

Sun et al. (2009) propose a GA approach for solving the ELSP under the extended basic period and power-of-two policy. The designed GA uses an integer encoding scheme which speeds up the search. With keeping both feasible and infeasible solutions in the population, optimal solutions are obtained for almost all the tested sample problems.

Rao et al. (2009) consider the ELSP in fuzzy environment with fuzzy costs and objective goal and propose a fuzzy GA approach for solving this problem. The impreciseness in inventory costs are taken into account and represented by fuzzy linear membership functions. To test the performance of the approach, benchmark problems are used and the results are compared with the results of crisp model. The results show that fuzzy GA gives good results and works better for higher utilization levels of the ELSP.

Santos et al. (2010) present a new GA with new components to deal with the capacitated lot sizing and scheduling problem with sequence dependent setups. To deal with the infeasibilities occurring during the run of GAs, the authors classify individuals according to their level of infeasibility in bands. Within each band, the fitness functions of individuals are different. Throughout iterations, the widths of bands are dynamically adjusted to improve the convergence of the individuals into the feasible domain. The computational results show that the proposed approach is effective in guiding the search toward feasible domain especially for highly capacitated instances.

Recently, Mohammadi and Ghomi (2011) propose a GA based heuristic for the CSLP in flow shops with sequence-dependent setups. The authors combine rolling horizon approach with GA. Experimental results show the outperformance of the proposed heuristic especially in solving large scale problems.

Gonçalves and Sousa (2011) develop a hybrid approach combining GA and linear programming for solving the ELSP. The authors develop a mixed integer non-linear programming formulation which takes explicit account of initial inventories, setup times and backorders. Experimental results validate the quality of the solutions and the effectiveness of the proposed approach.

Another GA based heuristic can be found in Gören et al. (2011) for solving the CLSP with setup carryover. The authors hybridize GA with a MIP based heuristic, namely the Fix-and-Optimize heuristic and also present a new initialization scheme based on the solution of the LP relaxation of the CLSP. In this study, the Fix-and-Optimize heuristic is embedded into the loop of GA. Following the mutation operator, a new population is formed and a randomly selected solution from this new population is set as an initial solution in the Fix-and-Optimize heuristic. After the solution is improved by the Fix-and-Optimize heuristic, it is put back into the population and the execution of the GA continues. The experimental studies show that the solution quality of the proposed hybrid approach is good when compared to the recent results reported in the literature. The details of this approach can be found in chapter five.

4.2.3 Research on Multi Level Uncapacitated Lot Sizing Problems

Multi level lot sizing introduces dependent demands: the lot sizing and timing decisions for items at one level in the product structure depend on the decisions made for their parents (Bahl et al., 1987).

The first GAs in solving uncapacitated multi level lot sizing problems is proposed by Xie and Dong (1995). The setup patterns are coded as binary integers in the chromosome and other decision variables are computed from these patterns.

Following Xie and Dong (1995), Dellaert and Jeunet (2000) develop a hybrid GA to solve uncapacitated multi level lot sizing problem. The authors employ the period order technique, the STIL algorithm and Wagner-Whitin based techniques to create the initial population and state that the proposed hybrid approach provides cost-effective solutions in a moderate execution time when compared with other techniques proposed in the literature.

Dellaert et al. (2000) develop a hybrid GA to solve the multi level lot sizing problem with no capacity and product structure constraints. The authors consider the most general statement of the problem in which the inventory holding and setup costs vary from one period to the next. The simulation results show that the proposed approach considering time varying costs makes it an appealing tool to industrials.

Prasad and Chetty (2001) present a new heuristic called Bit_Mod combined with a GA for multi level lot sizing under both fixed and rolling horizon and evaluate the influence of different parameters such as demand pattern, lot sizing rule, product structure and forecasting model under fixed and rolling horizon through simulation experiments.

4.2.4 Research on Multi Level Capacitated Lot Sizing Problems

Inclusion of capacity constraints and dependent demand between items make the problem much more complicated than the multi level uncapacitated lot sizing problems.

Kimms (1999) presents a MIP formulation and a GA approach for the multi level, multi-machine proportional lot sizing and scheduling problem. The author proposes a procedure in which a two-dimensional matrix is used to encode the solutions. The computational results show that the proposed approach outperforms the TS in terms of both run-time performance and finding the feasible solutions.

Özdamar and Barbarosoğlu (1999) propose two hybrid approaches for the multi level capacitated lot sizing and loading problem. The first one integrates GAs and SA whereas the second one consists of the lagrangean relaxation and SA. Experimental results show that the hybrid approach consisting of lagrangean relaxation and SA yields better results with respect to the solution quality and computation time.

Hung and Chien (2000) examine multiple demand classes in multi level capacitated lot sizing problem. Each demand class corresponds to a MIP model. The authors first generate feasible solutions by sequentially solving the MIP models each corresponding to a demand class, and then they employ TS, GAs and SA to solve the problem. Experimental results show that TS and SA yield better results than GAs.

Xie and Dong (2002) study the capacitated lot sizing problem with setup times and overtimes using GAs where the product structure is general acyclic network. The authors use only setup decision variables as chromosomes and other decision variables (inventory and lot sizes) are derived from these patterns. Since the problem involves capacity constraints, a heuristic approach based on lot shifting is embedded in the loop of GAs in order to eliminate the infeasible chromosomes.

Jung et al. (2007) use GAs in solving the integrated production planning problem in case of manufacturing partners (suppliers). The objective of this study is to provide efficient integrated production plans for manufacturing partners and a local firm under finite production capacity, while minimizing the total production cost. The authors formulate a MIP model by modifying the multi level lot sizing problem. With the unique chromosome structure, chromosome generation method and genetic operators, the proposed heuristic generates quite good solutions when compared with a commercial software optimization package.

Recently, another GA approach is proposed in Fakhrzad and Zare (2009) for solving the lot sizing problem in multi stage production systems. In the first step, the original problem is converted to several individual problems using a heuristic approach which is based on the Lagrangean multipliers. For solving these individual problems, in the second step a new approach combining GA and neighborhood search techniques is employed. In the last step, remaining capacities are calculated and resource leveling is implemented. These steps continue until a stopping criterion is met. The computational results show the effectiveness of the proposed approach in solving lot sizing problems.

4.2.5 Findings Based on the Problem Specifications

In single level systems, the product demands are directly derived from customer orders or market forecasts. However, in multi level systems due to the relationship between items, the researchers need to deal with the dependent demand which makes the multi level lot sizing problem more complicated. Adding other constraints such as capacity, setup times, backordering etc. makes this problem even more complex. As a result of surveying current relevant literature, it has been noted that most of the studies (i.e. 26 out of 36) focused on single level capacitated lot sizing problems.

When the capacity is tight, considering the issue of setup carryover helps in reducing the number of setups and related costs. This issue has been considered in only one study dealing with small bucket lot sizing problem involving proportional lot sizing and scheduling decisions (Kimms, 1999). Dealing with the large bucket lot sizing problem using GAs under setup carryover and capacity constraints can be a promising research area.

It has been noted that the issue of backordering has not received much attention (i.e. 9 out of 36 studies). Another important specification in lot sizing problems is the lead time. Except for Kimms (1999), Dellaert et al. (2000) and Hung and Chien (2000) nearly all studies reviewed, assume that the lead times are neglected. Moreover, in the majority of the studies reviewed, the planning horizon is assumed to be infinite or finite. Only two studies deal with the multi level uncapacitated lot sizing problem employing rolling horizon (Prasad & Chetty, 2001; Mohammadi & Ghomi, 2011). Under rolling horizon, making revisions in the current plan according to the demand forecast, often cause disturbances in production, inventory cost and supply of raw materials and subassemblies. These disturbances are called nervousness. Thus, lot sizing techniques developed for single level under deterministic conditions may not perform well under rolling horizon. The Wagner-Whitin algorithm (1958) which gives an optimum solution with fixed horizon, does not necessarily give good solutions under rolling horizon so other lot sizing heuristics are preferred in multi level settings (Prasad & Chetty, 2001). Hence, considering the fact that most real world problems are solved under rolling horizon, this issue has enough merits to take the attention of the researchers.

4.3 Genetic Algorithm Specifications

In this section, the proposed GAs were investigated based on six criteria; chromosome representation, initialization of the population, evaluation and selection, genetic operators (crossover and mutation), choice of the GA parameters and termination criteria. Table 4.2 presents the chronological order of published literature based on these six criteria.

4.3.1 Chromosome Representation Scheme

The first and the most important step in applying GAs to a particular problem is to convert solutions (individuals) of lot sizing problem into a string type structure called chromosome. This representation must uniquely map the chromosome values onto the decision variable domain. To represent a solution the following two classes have been noted:

Direct Representation: MIP models of dynamic lot sizing problems contain integer variables both for the setups and possibly for the sequencing. Moreover, continuous variables for the production quantities are also present in these models. Direct representation uses the integer variables representing the setup decisions as well as sequencing decisions and continuous variables representing the production quantities in the solution of GAs when dealing with dynamic lot sizing problems. Two options for this representation can be defined as follows:

1. The first option of this group includes variables for both the binary setup decisions and the continuous production quantities (Özdamar & Birbil, 1998; Özdamar & Barbarosoğlu, 1999; Barbarosoğlu & Özdamar, 2000; Özdamar & Barbarosoğlu, 2000; Özdamar & Bozyel, 2000; Özdamar et al., 2002).
2. The second option of this group includes only the integer variables and the production quantities can be found by solving the LP model, which is the original MIP problem with the integer variables fixed as in Hung et al. (1999) and Gören et al. (2011).

In the case of multi level uncapacitated lot sizing problem (Xie and Dong, 1995; Dellaert and Jeunet, 2000; Dellaert et al., 2000) the set up decisions automatically determine the optimal production quantities through the zero-switch rule (Askin & Goldberg, 2001). When dealing with the ELSPs, direct representation includes the variables for cycle times, integer multipliers and other variables related to the solution of the problem. The cycle times and integer multipliers are coded in the solution of GAs in Khouja et al. (1998) in which Basic Period Approach is used. In dealing with the ELSP using Extended Basic Period approach, Yao and Huang (2005) represent the solution as the multipliers of a basic period, Chang et al. (2006) use fundamental cycle time (basic period) in the solution encoding and Chatfield (2007) codes the fundamental cycle time, integer multipliers and production beginning periods in a chromosome. In Moon et al. (2006), each gene in a chromosome represents the frequency of each product. The cycle times and other related variables are calculated based on these frequencies obtained by GAs.

Table 4.2 The published literature based on the GA specifications in a chronological order

Literature reviewed	GENETIC ALGORITHM SPECIFICATIONS				
	Initialization of the population	Chromosome representation	Selection	Genetic operators	
				Crossover	Mutation
Xie and Dong (1995)	Randomly	Direct	Roulette wheel+elitism	One-point	Bit flip
Khouja et al. (1998)	Randomly	Direct	Tournament+ elitism	One point Two point Uniform	Bit flip
Ozdamar and Birbil (1998)	Randomly	Direct	Proportional to the objective value	Two point	Randomly
Hernandez and Suer (1999)	Randomly	Direct	Roulette wheel+scaling	One-point	Bit flip
Kimms (1999)	Randomly	Indirect	Deterministically	Problem specific	Problem specific
Ozdamar and Barbarosoglu (1999)	Randomly	Direct	Proportional to the objective value	Two point	Randomly
Hung et al. (1999)	NA	Direct	Roulette wheel	Problem specific	Bit flip
Kohlmorgen et al. (1999)	NA	Direct	NA	Two point	Randomly
Ozdamar and Bozyel (2000)	Randomly	Direct	NA	Linear Order (two point)	Randomly
Dellaert and Jeunet (2000)	From insertion of the solutions of different lot sizing rules	Direct	Clustering+elitism	One point	Bit flip
Dellaert et al. (2000)	Application of the mutation operator and a replenishment rule	Direct	Scaling+elitism	Problem specific	Bit flip
Hung and Chien (2000)	NA	Direct	Roulette wheel+scaling	Problem specific	Problem specific
Prasad and Chetty (2001)	Randomly	Direct	Ranking	NA	NA
Sarker and Newton (2002)	Randomly	Direct	Ranking	Two point	Bit flip

Table 4.2 The published literature based on the GA specifications in a chronological order (cont.)

Literature reviewed	GENETIC ALGORITHM SPECIFICATIONS				
	Initialization of the population	Chromosome representation	Selection	Genetic operators	
				Crossover	Mutation
Ozdamar et al. (2002)	Randomly	Direct	Proportional to the objective value	Two point	Randomly
Moon et. al. (2002)	From the solution of the nonlinear model of the ELSP	Indirect	Stochastic tournament+scaling+elitism	Partial Mapped	Randomly
Xie and Dong (2002)	Randomly	Direct	Roulette wheel + elitisim	One point	Bit flip
Duda (2005)	Only the predetermined part is randomly generated	Direct	Binary tournament+elitism	Problem specific	Irregular
Yao and Huang (2005)	Randomly	Direct	Roulette wheel+elitism+linear ranking	Two point Uniform	Bit flip
Hop and Tabucanon (2005)	NA	Direct	NA	One point	Bit flip
Chang et. al. (2006)	Randomly	Direct	Roulette wheel	One point	Bit flip
Kämpf and Köchel (2006)	NA	Direct	Tournament+elitism	One point	One gene Randomly
Megala and Jawahar (2006)	Randomly	Direct	Roulette wheel+scaling	Partial Mapped	Randomly
Moon et al. (2006)	From heuristic solutions	Direct	Ranking	Uniform	Randomly
Gaafar (2006)	Randomly	Direct	Roulette wheel with elitism	Simple Uniform	Randomly
Li et al. (2007)	Randomly	Direct	Ranking+Elitism	One point	Bit flip
Chatfield (2007)	Randomly	Direct	Roulette wheel+elitism+scaling	One point	Bit flip
Jung et al. (2007)	From the Proposed procedure	Direct	Minimum generation gap selection	Problem specific	Problem specific

Table 4.2 The published literature based on the GA specifications in a chronological order (cont.)

Literature reviewed	GENETIC ALGORITHM SPECIFICATIONS				
	Initialization of the population	Chromosome representation	Selection	Genetic operators	
Duda and Osyczka (2007)	NA	Direct	Roulette wheel + Tournament	Problem specific	Regular
Fakhrzad and Zare (2009)	From heuristic solutions and randomly	Direct	Restart strategy	NA	Problem specific
Rao et al. (2009)	Randomly	Direct	Tournament	Simulated Binary	Problem specific
Sun et al. (2009)	Randomly	Direct	Elitism + proportional to the fitness value	Two point	Problem specific
Santos et al. (2010)	From heuristic solutions	Indirect	Positional roulette wheel + elitism	One point crossover + problem specific	Problem specific
Mohammadi and Ghomi (2011)	From heuristic solution	Indirect	Tournament	Problem specific	Shift
Gören et al. (2011)	From heuristic solutions and randomly	Direct	Ranking based roulette wheel + elitism	One point	Bit flip
Gonçalves and Sousa (2011)	Randomly	Indirect	Elitism	Parameterized uniform	Problem specific

Indirect Representation: The second class in the representation is indirect representation of the solution. To represent a solution, a two dimensional matrix in which each entry represents a rule for selecting the setup state for each machine at the end of the period is employed in Kimms (1999). Recently, in Santos et al. (2010) and Mohammadi and Ghomi (2011) the chromosome shows the sequence of the products produced in a period.

In the case of ELSPs, Moon et al. (2002), who deal the problem with time varying lot size approach, use two kinds of chromosomes, the first of which represents the item number and the second one shows the absolute locations of the genes. These two chromosomes are useful in determining the frequencies of the items. Gonçalves and Sousa (2011) propose another different representation which shows the maximal production sequence and number of setups used to construct the production sequence.

4.3.2 Initial Population

The search in GA starts from an initial population. In the majority of the papers surveyed, the initial population is created randomly (i.e. 19 out of 36). Inclusion of heuristically generated solutions to the initial population is first reported in Dellaert and Jeunet (2000). As initial population, Moon et al. (2002) employ the nonlinear solution of the economic lot size scheduling problem. In Duda (2005), only the predetermined part (only about 3%) of the initial population is generated randomly. Moon et al. (2006) use the heuristic solutions of Kuo and Inman's (1990) in constructing a diversified initial population. Fakhrzad and Zare (2009) utilize from Wagner-Whitin algorithm for the initial population.

Moreover, Santos et al. (2010) use four different heuristic methods based on lot-for-lot basis for creating the initial population. Mohammadi and Ghomi (2011) construct the initial population through a simple and effective heuristic which considers the setup costs of the products. Recently, Gören et al. (2011) propose a new strategy for creating the initial population. This strategy utilizes the solution obtained from the LP relaxation of the CLSP. The half of the initial population is created based on the solution and the other half is created randomly.

4.3.3 Evaluation and Selection

Following the generation of the initial population, the fitness of each individual in the population is calculated by employing an appropriate objective function. The objective function provides a measure of a chromosome's performance or fitness in the search space. The potential parents are selected to create the offsprings based on their relative fitness after the evaluation of each chromosome. It has been noted in this review study that with some exceptions (see Table 4.2); the roulette wheel selection is used in most of the studies. In roulette wheel selection, a sector of a roulette wheel whose size is proportional to the appropriate fitness measure is assigned to the individuals. Then a random number is generated and the parents are selected according to their random position on the wheel.

Besides the roulette wheel selection operator, a number of other different selection operators are also used in the studies reviewed. Kimms (1999) select the parents deterministically in which the individuals with the highest fitness functions are chosen to become parents. Özdamar and Bozyel (2000) and Özdamar et al. (2002) select the chromosomes according to their fitness values. The difference from the probability calculation in roulette wheel selection lies in the calculation of the reproduction probability which is a second degree polynomial of the inverse of the chromosome's objective function value given in Equation (1).

$$prob_k = \frac{(1/obj^2)}{\sum_{k \in population}(1/obj^2)} \tag{1}$$

The parents are selected based on the stochastic tournament in Moon et al. (2002). In a recent study, Jung et al. (2007) propose a different selection operator based on the minimum generation gap selection method. The proposed operator randomly selects two different chromosomes from the old population and applies genetic operators to produce the offsprings. Two chromosomes with the highest fitness values are selected among two chromosomes from the offsprings and two parents and then enter the next generation. One of the strengths of this method is that it tries to preserve the good parents in the next generation, while other typical selection methods construct a set of candidates consisting of only child chromosomes so that the parents with good genes can not enter the next generation.

It has been noted that in order to control the selection process, scaling is used in a number of studies reviewed. In the proportional selection procedure, the selection probability of a chromosome is proportional to its fitness. However, this selection property exhibits some undesirable properties. For example, in early generations, there is a tendency for a few super chromosomes to dominate the selection process. In later generations, when the population has largely converged, the competition among chromosomes becomes less strong and a random search behavior emerges. Hence, scaling and ranking are proposed to mitigate these problems. Scaling maps the raw objective function to some positive real values in which the survival probability for each chromosome is determined according to these values (Gen & Cheng, 1997). Hernandez and Suer (1999), Hung and Chien (2000), Dellaert et al. (2000), Moon et al. (2002), Yao and Huang (2005), Megala and Jawahar (2006) use the scaling procedure in the selection process. Recently, Gören et al. (2011) determine the selection probabilities based on the ranking of the chromosomes. In this approach, chromosomes in the population are sorted according to their fitness values and a rank is assigned to each chromosome. Based on these rankings, the selection probabilities are determined and then the standard roulette wheel selection operator is applied to select parents which will produce offsprings. The authors name this selection procedure *Ranking based roulette wheel* selection operator.

In a GA, survival is an important process that removes the individuals having low fitness. Survival is related to the population size. All the studies reviewed assume a constant population size. In steady-state GAs, which employ a constant population size a survival scheme is needed to reduce the population size to its predetermined value after generating the offsprings. In nearly half of the studies reviewed (i.e. 16 out of 36), elitism strategy is used as a survival scheme. Elitism strategy preserves the best individuals in one generation and translates them to the next generation without any change.

Another important issue in designing the GA is to decide on whether infeasible individuals should be allowed in the population or not. We noted a number of approaches dealing with the infeasible solutions. These approaches can be listed as follows:

- To discard all infeasible solutions or attach an infinite cost to them (Kimms, 1999; Dellaert et al., 2000; Dellaert & Jeunet, 2000; Duda, 2005; Li et al., 2007).
- To use some penalty costs (Khouja et al., 1998; Özdamar & Birbil, 1998; Özdamar & Barbarosoğlu, 1999; Özdamar & Barbarosoğlu, 2000; Xie & Dong, 2002; Sarker & Newton, 2002; Duda, 2005; Moon et al., 2006; Duda & Osyczka, 2007; Chatfield, 2007; Gören et al., 2011).
- To repair the infeasible solutions by introducing some repair operators (Özdamar & Birbil, 1998; Özdamar & Barbarosoğlu, 1999; Özdamar & Bozyel, 2000; Dellaert et al., 2000; Duda, 2005; Van Hop & Tabucanon, 2005; Jung et al., 2007; Duda & Osyczka, 2007; Sun et al., 2009; Santos et al., 2010; Gören et al., 2011).

4.3.4 Genetic Operators

Genetic operators such as crossover and mutation are used in order to explore the search space. The crossover operator combines the chromosomes selected by the selection operator into a new chromosome. The mutation is used to maintain the diversity in the population.

During the literature survey, it has been noted that the authors develop problem-specific crossover operators to reflect the peculiarities of the lot sizing problems they study. Some of these crossover operators include *Period* (Hung et al., 1999; Dellaert et al., 2000; Jung et al., 2007), *Product* (Hung et al., 1999; Dellaert et al., 2000), *Compare* (Hung et al., 1999), *Random* (Hung et al., 1999; Hung & Chien, 2000), and *Partial Mapped Crossover* (*PMX*) (Moon et al., 2002; Megala & Jawahar, 2006). *Similar block order crossover*, *Similar job two point crossover* and *Similar block two point crossover* (Mohammadi & Ghomi, 2011).

Moon et al. (2002) use *PMX* Crossover which prevents two or more genes having the same value. Under PMX, two strings are aligned, and two crossing points are picked uniformly at random along the strings. These two points define a matching selection that is used to affect a cross through position-by-position exchange of operators. Duda (2005) presents a new type of crossover in which the fitness's of the parents are considered. In his representation, a chromosome shows the lots produced per shift in a day. The crossover operator randomly chooses a string of genes (representing the lots in a single shift) in two parents. If the fitness value of the first parent is better than that of the second parent, the string in the first parent is placed in the second parent. If the second parent has better fitness, the string in the second parent replaces the lots in the first parent. Four different crossover operators are proposed in Santos et al. (2010) for creating offsprings. The first one is the standard *one point* crossover which works by choosing a period randomly and then swapping segments of two parents. The second one is the *intersection* crossover which works by checking the common parts of both parents. If both parents have the same common parts, then these parts are automatically put in the offsprings. The third one is the *union* crossover operator which considers the ranking positions of the parents in the population. The last crossover operator combines the weighted roulette and *T*-point (*T* is the number of periods) crossover in producing offsprings. Gonçalves and Sousa (2011) employ *parameterized uniform* crossover in place of the traditional one-point and two-point crossover. After choosing two parents this crossover operator involves generating a random number between 0 and 1 for each gene. If this random number is less than or equal to the crossover probability, the gene of the offspring is set to the corresponding gene in the first parent; otherwise it is set to the corresponding gene of the second parent.

Moreover, it has been noted that single bit flip mutation operator is used in most of the studies where the setup state is changed for a randomly chosen item and period (Dellaert et al., 2000; Dellaert & Jeunet, 2000; Xie & Dong, 2002; Gören et al., 2011). In a representation in which the lot sizes are also included, the mutation operator can change the lot size by a random amount (Özdamar & Birbil, 1998; Barbarosoğlu & Özdamar, 1999; Özdamar & Bozyel; 2000, Fakhrzad & Zare, 2009). Specialized mutation operators are also developed to reflect the peculiarities of the lot sizing problems in a number of studies. To account for product interdependencies in multi level case, Dellaert et al. (2000) propose a mutation operator called *Cumulative,* in which the setup periods for the predecessors are changed when mutation is performed on a given item. The mutation operator performs the same mutation on the immediate predecessors of the item under consideration and tries to ensure the feasibility. Gaafar (2006) uses different mutation operators during the genetic algorithm application such as *Random*, *Boundary*, *Ordering*, *Change ordering*, *Random swapping* and *Neighbor swapping*. *Random* mutation replaces a randomly chosen chromosome with a new one generated randomly. *Boundary* mutation randomly switches genes from a randomly selected position till the end of the chromosome. *Ordering* mutation maintains the number of orders in a randomly chosen chromosome. *Change ordering* mutation randomly increases or decreases the number of orders in a randomly chosen chromosome. *Random swapping* mutation swaps two randomly chosen genes in a randomly chosen chromosome and *Neighbor swapping* mutation swaps a randomly chosen gene with the neighbor gene (before or next to it) in a randomly chosen chromosome. Jung et al. (2007) propose different problem specific mutation operators matching with the chromosome representation they use. These operators along with the chromosome representation and the crossover operator proved to be efficient in terms of solution quality. A number of different mutation operators can also be found in Santos et al. (2010) such as *Scramble*, *Insertion*, *Displacement*, *Replacement*, *Setup Carryover*, *Minmax feasibility* (*random*), *Minmax feasibility* (*static*), *Minmax feasibility* and *shifting forward*. *Scramble* selects a random period and scrambles the productions in it. *Insertion* chooses a period randomly and a product that is not produced in that period and inserts this product in a random position. *Displacement* picks a period randomly and removes a random product from the chromosome. *Replacement* combines an insertion and a displacement procedure. *Setup carryover* looks for improvement in the solution by changing the initial setup state of the machine in a randomly selected period. *Minmax feasibility* (*random*) re-sequences every chromosome according to the minmax random version detailed in Almada-Lobo (2007) who tries to minimize the setup times. *Shifting forward* tries to reduce the inventory holding costs by shifting forward a fraction or an entire lot of every product from a source period to the following period. Instead of using a traditional mutation operator, Gonçalves and Sousa (2011) randomly generate some new chromosomes to prevent premature convergence of the population, like a mutation operator.

In addition to crossover and mutation operators, some different operators such as the migration and sibling operators are also proposed for lot sizing problems. The migration operator allows for a crossover between chromosomes from different populations (Özdamar & Birbil, 1998; Barbarosoğlu & Özdamar; 1999, Özdamar et al., 2002) helping in generating good quality offsprings and variety in population. Hung et al. (1999) propose a sibling operator which performs like crossover and mutation operators. It stochastically chooses a better sibling from the neighborhood of a chromosome in the current generation and creates new chromosomes from the next generation.

4.3.5 Choice of the Parameters of GAs

The performance of the GAs depends on the rates of the parameters such as the population size, crossover rate and mutation rate. However, the optimization of the parameter set has not attracted the attention of the researchers. Hung et al. (1999) determine the suitable parameters through conducting a series of pilot experiments. The authors use the default parameter file and vary only one control parameter at a time from the default control parameter set. Jung et al. (2007) determine the parameter set through an experimental setup including three levels for each parameter. The details of the experiments are not explained in the study. Chatfield (2007) tests different crossover and mutation rates and uses a performance metric which tracks a genetic algorithm's progress toward finding the best solution. Mohammadi and Ghomi (2011), Gonçalves and Sousa (2011) and Gören et al. (2011) also determine the level of GA parameters through a set of preliminary tests.

A new trend in parameter setting in GAs is adaptive genetic operators in which the rates of genetic operators change during the search. Özdamar and Birbil (1998), Özdamar and Barbarosoğlu (1999), Özdamar and Bozyel (2000), Özdamar et al. (2002) determine the crossover and mutation probability according to the convergence of the population's performance range. The crossover and mutation probabilities increase when the population's performance range tends to get stuck at a local optimum. Dellaert and Jeunet (2000) implement learning algorithm on the probabilities to the operators in GAs. Hung and Chien (2000) and Prasad and Chetty (2001) use the adaptive GA proposed by Srinivas and Patnaik (1994), which dynamically adjusts the crossover and mutation probability. Starting with a high crossover rate and a low mutation rate, Yao and Huang (2005) keep decreasing the crossover rate and increasing the mutation rate until a specified level. The authors hope that in doing so the GAs can still explore the new regions in the search space and raise the diversity of the population, during the evolutionary process. Van Hop and Tabucanon (2005) propose using adaptive genetic operators which are based on the rate of survivor off-springs. Chang et al. (2006) use adaptive probabilities for the crossover and mutation. Namely, when the GA keeps finding the same chromosome with the highest fitness value for some number of generations, both crossover and mutation probabilities are increased gradually to further increase the chromosome diversity. Once another chromosome is found, these probabilities are set to their default values.

4.3.6 Termination

The search in GAs is terminated according to some rules. In surveying the current relevant literature, it has been noted that most of the researchers specified a maximum number of generations as a terminating condition. However, in some studies, the search is stopped if there is no improvement in the last predefined generations or the best individual does not improve more than the predetermined threshold in a predetermined number of generations.

4.3.7 Findings Based on the GA Specifications

Since the search in GAs starts from an initial population, the initial population has an important effect on the performance of the GAs. In this survey study, it has been noted that the initial population is generated randomly in most of the studies. Most of the researchers use the direct representation as a representation scheme.

In order to reflect the peculiarities of the lot sizing problems, a number of different crossover and mutation operators are used (see Section 4.3.4). Despite the great variety in proposed GA operators, it has been noted that the researchers mainly employ one point crossover and single bit flip. Moreover, it has been noted that when the representation of the solution is unique to the problem type (Delleart et al., 2000; Moon et al., 2002; Gaafar, 2006; Jung et al., 2007; Santos et al., 2010; Mohammadi & Ghomi, 2011) usually problem specific genetic operators are used. To help the convergence of the GAs, other operators such as migration (Özdamar & Birbil, 1998; Barbarosoğlu & Özdamar, 1999; Özdamar et al., 2002) and sibling operator (Hung et al., 1999) are also proposed.

Since the performance of the GAs heavily depends on the genetic operators used, the determination of efficient GA parameters is another important issue in designing the GAs. However, any study dealing with this issue has not been noted during the literature review. In recent years, a new trend focusing on adaptive genetic operators has received attention in the literature. To obtain better and faster solutions, Özdamar and Birbil (1998), Özdamar and Barbarosoğlu (1999), Özdamar and Bozyel (2000), Özdamar et al. (2002), Hung and Chien (2000), Prasad and Chetty (2001), Van Hop and Tabucanon (2005), Yao and Huang (2005), Chang et al. (2006) propose adaptive genetic operators in which the rates of the genetic operators are automatically adjusted during the search.

As a result of surveying the current relevant literature, it has been noted that when the representation of the solution is unique to the problem type, the problem specific genetic operators are used to reflect the peculiarities of the problems. This statement implies that for an effective GA design, taking into account the genetic operators together with the chromosome representation is a better idea.

The insight gained as a result of surveying current relative literature with respect to problem specifications and also GA specifications can be given as follows:

- Unlike single level lot sizing problems where the researchers employed standard genetic operators, for multi level lot sizing problems, some problem specific operators considering the interdependency among items in the product structure have been proposed (Dellaert & Jeunet 2000; Dellaert et al. 2000; Jung et al. 2007). This can be attributed to the complexity of multi level lot sizing problems. These specific genetic operators constrain the search to the set of feasible solutions rather than letting the algorithm explore infeasible solutions. This speeds the convergence of the algorithm to the optimum.
- An important issue in solving multi item stochastic capacitated lot sizing problem using GAs is to realistically deal with probabilistic demand. It has been noted that combining specially designed GAs with simulation has quite potential to find sufficiently good solutions (Kämpf & Köchel, 2006).
- In most studies considering ELSPs the penalty functions are used to deal with infeasible solutions. A recent trend to enhance the feasibility testing and generate a feasible production schedule is to employ heuristics (Yao & Huang, 2005). These heuristics check the feasibility of solutions generated and GAs search for the best solution among all feasible ones.
- The island GAs which are executed concurrently on several sub-populations with the added possibility of exchanging regularly good individuals between neighboring islands offer many advantages in obtaining satisfying results in a reasonable computational time. It has been noted these island GAs are employed only for single level capacitated lot sizing problem (Kohlmorgen et

al., 1999). This can be attributed to the complex structure of the multi level lot sizing problem.

- In solving dynamic lot sizing problems which involve many constraints to be satisfied, repair operators are widely used to ensure the feasibility of the capacity, inventory and other constraints (Özdamar & Birbil 1998; Kimms 1999; Özdamar & Barbarosoğlu 1999; Dellaert et al. 2000; Özdamar et al. 2002; Duda, 2005; Jung et al., 2007; Santos et al., 2010; Gören et al., 2011).
- In solving capacitated lot sizing and loading problem involving the issues of setup and overtime, the chromosome is structured to include both lot sizing and loading decisions (Özdamar & Birbil, 1998; Özdamar & Barbarosoğlu; 1999).
- Unlike classical lot sizing problem in the literature, some researchers (Jung et al., 2007) take into account manufacturing partners and they propose case specific GAs approach, i.e. problem specific representation, genetic operators.

4.3.8 Motivation of This Ph.D. Study

GAs are increasingly used to solve many different production and operations management problems. One of the most well-known of these problems is lot sizing problems which deals with the determination of the lot sizes in order to minimize average total cost per time unit. To state the current research issues on solving lot sizing problems using GAs, this chapter summarizes the main specifications of the problems studied, and discusses the features of the proposed GAs to deal with these problem specifications.

From the perspective of GAs, it is noted in most of the published literature that the proposed GAs are found to have better performance with respect to solution quality and speed than exact solution approaches and other heuristic approaches. However, it is known from published literature that it is possible to further improve the efficiency of the proposed GAs by appropriate selection of control parameters and the initial population. Despite this fact, it has been noted that choosing efficient parameters and population initialization did not attract many researchers.

As for the lot sizing perspective, more than half of the studies reviewed focus on the single level lot sizing problems. To get more realistic results regarding the performance of the GAs, the focus might be on real world lot sizing problems or more complex lot sizing problems including features such as setup times, setup carryover, sequence dependent setup costs, parallel machines, backordering, rolling horizon and lead times can be taken into consideration. To the best of our knowledge, we have not noted any study dealing with single level or multi level multi-item lot sizing problem with setup times, setup carryover and backordering using GAs.

Having gained insight on application of GAs to lot sizing problems, and identified the current research gaps, we set our research directions for this Ph.D study as follows:

- *To focus on the single level capacitated lot sizing problem with setup times, setup carryover and backordering.*

  This literature survey reveals that the number of studies addressing the solution of the capacitated lot sizing problem with setup times, setup carryover and backordering is very limited. To fill this perceived gap, in this study the focus will be on the single level capacitated lot sizing problem with setup times, setup carryover and backordering.

- *To propose hybrid GA approaches to solve the capacitated lot sizing problem with setup times, setup carryover and backordering.*

  The CLSP with extensions such as setup times, setup carryover and backorder is more complicated than the regular CLSP. However, as a result of surveying current relevant research, it is noted that the emphasis given to the development and application of hybrid GA approaches for solving this complicated problem is not adequate. Hence, to fill the perceived gap, this Ph.D. study proposes a number of GA-based hybrid approaches to solve the capacitated lot sizing problem with setup times, setup carryover and backordering.

- *To propose a new population initialization scheme utilizing problem specific knowledge.*

  It is noted that most of the published studies in this field prefer random initial population in the GA search. However, it is a well known fact that the initial population has an important effect on the performance of the GA. To further improve the performance of proposed GA-based hybrid approaches, in this Ph.D. study we propose a novel initialization scheme based on problem specific information.

- *To set the values of efficient GA parameters through pilot experiments over different instances.*

  It is known from the literature that performance of GAs is affected by choice of control parameters. However, it is noted that in the most of the published literature, no effort has been made to determine the efficient GA parameters. Unlike previous relevant research, in this study, experimental studies are carried out to select efficient GA parameters.

In summary, this Ph.D. study proposes novel GA-based hybrid approaches which integrate GA with Fix and Optimize Heuristic to solve the capacitated lot sizing problem with setup times, setup carryover and backordering. Moreover, to further improve the performance of proposed GA-based hybrid approaches, it proposes to use a novel initialization scheme in these approaches.

4.4 Chapter Summary

In this chaper, the current studies on applications of GAs for solving lot sizing problems with different features were extensively reviewed to state the current research gaps and motivation for this Ph.D. thesis.

The review includes the studies published after 1995. The current literature was reviewed from two different perspectives. The first one is the specifications of the lot sizing problems such as number of levels, capacity constraints, setup time issue, planning horizon, demand type and inventory shortage. The second one is the features of the proposed GAs to deal with these problem specifications such as chromosome representation scheme, initialization of the population, evaluation, selection, genetic operators, choice of the GA parameters and termination.

To fill the perceived research gap, this Ph.D. study focuses on capacitated lot sizing problem with setup times, setup carryover and backordering which is the least studied lot sizing problem in the literature and proposes novel GA-based hybrid approaches to solve this complex problem. The details about these hybrid approaches are presented in next chapters.

CHAPTER FIVE
HYBRID APPROACHES FOR SOLVING THE CAPACITATED LOT SIZING PROBLEM WITH SETUP CARRYOVER

5.1 Introduction

The main objective of this Ph.D. study is to develop efficient GA-based solution approaches for solving capacitated lot sizing problem with setup carryover and backordering. It is known that pure GAs are good at identifying promising areas in the search space, however they have problems in finding the exact global optimum (Taşan, 2007). In order to improve the performance of the pure GAs, this Ph.D. study presents novel hybrid approaches combining GAs with a MIP based heuristic, namely Fix-and-Optimize heuristic. It should be noted that all these hybrid approaches employ a new population initialization scheme and an efficient GA control parameter setting determined as a result of experimental analysis. The capacitated lot sizing problem with setup carryover and backordering is solved in two stages. First, the proposed hybrid approaches are used to solve the capacitated lot sizing problem with setup carryover (CLSPC). In the second stage, the issue of backordering is considered and the problem is extended to the capacitated lot sizing problem with setup carryover and backordering ($CLSP^{+}$). While this chapter presents the studies done during the first stage, how the proposed hybrid approaches are adapted to solve the capacitated lot sizing problem with setup carryover and backordering are given in chapter six.

The rest of the chapter is organized as follows. In section 5.2, the statement of the problem is given. The details of the proposed approaches are given in section 5.3. The results of experimental studies carried out to test the performance of the proposed hybrid approaches are discussed in section 5.4. Finally, in section 5.5 the context of this chapter is summarized.

5.2 Problem Statement: The Capacitated Lot Sizing Problem with Setup Carryover

The CLSPC determines the timing and sizing of the production along with the semi-sequencing (i.e. first and last product produced in a period) of the products in a period. In this study, the model proposed by Suerie and Stadtler (2003) has been employed under the following assumptions to solve the CLSPC:

- The planning horizon P is fixed and divided into time buckets (1,….., P).
- There is one resource available.
- Several products requiring a unique setup state can be produced on the resource in each period (property of a big bucket model).
- The resource consumption to produce one unit of product j is fixed.
- Setups incur setup costs and consume setup time. Setup costs and setup times are sequence independent.
- At most one setup state can be carried over from one period to the next on the resource.
- Single item production is possible (i.e. the conservation of one setup state for the same product over two consecutive bucket boundaries).
- A setup state is not lost if there is no production on the resource within a bucket.

5.2.1 Sets, Indices, Parameters, and Variables

The sets, indices, parameters and decision variables in this problem are given as follows.

<u>Sets and indices:</u>
j: items $j \in K = \{1, 2, 3, ...K\}$;
t: periods $t \in P = \{1, 2, 3, ...P\}$.
<u>Parameters:</u>
$sc_{j:}$ setup cost for item j;
$h_{jt:}$ unit holding cost for item j in period t;
$C_{t:}$ amount of capacity available in period t;
a_j: time to process one unit of item j;

st_j: setup time of item j;
M: a large number;
d_{jt}: demand for item j in period t.

Decision Variables:

I_{jt}: Inventory level for item j at the end of period t,
X_{jt}: Production amount of item j in period t,
Y_{jt}: Binary setup variable (=1, if a setup for item j is performed in period t, =0 otherwise),
W_{jt}: Binary linkage variable which indicates whether a setup state for item j is carried from period (t-1) to (t) (=1) or not (=0),
Q_t: Single item variable which indicates that the resource is occupied solely by item i in period t (=1) or not (=0).

5.2.2 Mathematical Model of the CLSPC

The mathematical programming model in the form of "Inventory and Lot Size Representation (I&L)" for the CSLPC can be stated as follows (Suerie & Stadtler, 2003):

$$Min \sum_{j=1}^{K}\sum_{t=1}^{P}(sc_j Y_{jt} + h_{jt} I_{jt}) \qquad (1)$$

s.t.

$$I_{j,t-1} + X_{jt} - I_{jt} = d_{jt} \quad \forall j \in K; \forall t \in P \qquad (2)$$

$$\sum_{j \in K}(a_j X_{jt} + st_j Y_{jt}) \le C_t \quad \forall t \in P \qquad (3)$$

$$\sum_{j \in K} W_{jt} \le 1 \quad \forall t \in \{2,....,P\} \qquad (4)$$

$$W_{jt} \le Y_{jt-1} + W_{jt-1} \quad \forall j \in K, \forall t \in \{2,....,P\} \qquad (5)$$

$$W_{jt+1} + W_{jt} \le 1 + Q_t \quad \forall\, t \in \{1,....,P-1\} \qquad (6)$$

$$Y_{jt} + Q_t \le 1 \quad \forall\, t \in \{1,....,P-1\} \qquad (7)$$

$$X_{jt} \le M(Y_{jt} + W_{jt}) \; \forall j \in K; \forall\, t \in \{1,....,P\} \qquad (8)$$

$$Q_t \ge 0 \quad \forall\, t \in \{1,....,P-1\} \qquad (9)$$

$$W_{jt} \in \{0,1\} \quad (W_{j1} = 0), \quad Y_{jt} \in \{0,1\} \quad \forall j \in K; \forall t \in P \qquad (10)$$

$$X_{jt}, I_{jt} \geq 0 \qquad \forall j \in K; \forall t \in P \tag{11}$$

The objective function (1) aims at the minimization of inventory holding and setup costs. Constraints (2) are the inventory balance equations. The capacity constraints are placed in Constraints (3). Constraints (4) ensure that at most one setup state can be preserved from one period to the next on the resource. Constraints (5) guarantee that a setup can be carried over to period *t* only if either item *j* is setup in period *t*-1 or the setup state is already carried over from period *t*-2 to *t*-1. A setup state can only be preserved over two bucket boundaries, if $Q_t = 1$ in constraints (6), which is only possible if there is no setup in this period (7). Constraints (8) are the upper bounds on the production quantities. Finally, variables are restricted to be nonnegative or binary, respectively, (9) to (11). It is assumed that there are no setup carryovers in the first period as stated in constraints (10). More details on this formulation can be found in Suerie and Stadtler (2003).

5.3 Proposed Hybrid Approaches

In this section, two hybrid approaches are proposed to solve the CLSPC. First, the details and logic of the genetic search is given. For hybridization of GA, two different schemes are proposed. The main elements of these two hybrid approaches and the proposed methodologies are given in the following sections.

5.3.1 The Logic of the Search in GAs

The search in a GA is done for the setup (Y_{jt}) and setup carryover variables (W_{jt}), which are binary. Once these variables are fixed to the specific values according to the chromosome values, the result is a LP model which deals with determining the optimal production and inventory decision variables. This LP model is called a sub-problem. Therefore, for each chromosome in the population of the GA, the setup and setup carryover variables are fixed to the values determined throughout the search and the production and inventory variables are determined by the LP model. The advantage of embedding a LP sub-problem in the GA can be explained as follows.

For a given integer solution, there may be infinite combinations of the values for the continuous variables. By solving the LP sub-problem, values that optimally correspond to integer solution can be obtained easily (Defersha & Chen, 2008). Hence, the objective value of the sub-problem is used to determine the fitness of the chromosome during the search.

5.3.2 Elements of the Hybrid Approaches

The following sections present the elements of the proposed hybrid approaches for solving the CLSPC.

5.3.2.1 The Fix-and-Optimize Heuristic

The Fix-and-Optimize heuristic is a MIP based heuristic in which a sequence of MIPs is solved over all real-valued decision variables and a subset of the binary variables. The background information about this MIP based heuristic can be found in chapter three. In this chapter, only the heuristic with the time decomposition is explained.

The Fix-and-Optimize heuristic is originally implemented in Sahling et al. (2009) for solving the multi level CLSPC. Later, in another study, Helber and Sahling (2010) employed a Fix-and-Optimize heuristic to solve the multi level capacitated lot sizing problem. It should be noted that our Fix-and-Optimize heuristic differs from Sahling et al. (2009) and Helber and Sahling (2010) in the following ways:

- The Fix-and-Optimize heuristic is implemented as a stand-alone heuristic by Sahling et al. (2009) and Helber and Sahling (2010), whereas in this study it is integrated with GA and it is used to improve the solutions obtained by the GA in both of the proposed hybrid approaches.
- In Sahling et al. (2009) and Helber and Sahling (2010), it is assumed that initially, there is a setup for each product in each period. The heuristic proceeds based on this initial solution and tries to improve it throughout the

iterations. In both hybrid approaches proposed in this study, the initial solution is determined by the GA.

Integrating the GA with the Fix-and-Optimize heuristic, it is hoped that the Fix-and-Optimize heuristic will act like a diversification tool in the solution space and help the GA to overcome local minima by guiding it to new regions. Particularly, the Fix-and-Optimize heuristic is used to improve the solution quality of the GAs. The definitions of the problems and algorithm are given in detail in the following sections.

5.3.2.2 Definition of Problems Obtained by Time Decomposition in the Fix-and-Optimize Heuristic

By adding the following constraints into the MIP model given, the smaller problems used in the Fix-and-Optimize heuristic are obtained.

$$Y_{jt} = \overline{Y}_{jt} \quad \forall (j,t) \in KP_{Y,s}^{fix} \tag{12}$$

$$W_{jt} = \overline{W}_{jt} \quad \forall (j,t) \in KP_{W,s}^{fix} \tag{13}$$

The explanations of the parameters used above can be found in Table 5.1. The Fix-and-Optimize heuristic starts with an initial solution. This initial solution yields an initial objective value which is indicated as Z^{old}. After initialization, the algorithm iterates through the ordered set of problems according to the time decomposition scheme either once or until it reaches a local optimum.

Table 5.1 Additional notation for the definition of a problem

<u>Sets:</u>	
KP	Set of all product-period (j,t) combinations $\mid j \in K \quad and \quad t \in P$
$KP^{opt}_{Y,s} \subseteq KP$	Set of product-period combinations for which the binary setup variables Y_{jt} are optimized in the current problem s
$KP^{opt}_{W,s} \subseteq KP$	Set of product-period combinations for which the binary setup carryover W_{jt} variables are optimized in the current problem s
$KP^{fix}_{Y,s} \subseteq KP$	Set of product-period combinations for which the binary setup variables Y_{jt} are fixed in the current problem s
$KP^{fix}_{W,s} \subseteq KP$	Set of product-period combinations for which the binary setup carryover variables W_{jt} are fixed in the current problem s
<u>Parameters:</u>	
$\overline{Y}_{jt}$	Exogenous value of the fixed setup variable Y_{jt}
$\overline{W}_{jt}$	Exogenous value of the fixed setup carryover variable W_{jt}

5.3.2.3 The Algorithm with Time Decomposition

The basic structure of the Fix-and-Optimize algorithm with the time decomposition scheme is outlined in Figure 5.1. The algorithm needs an initial solution to start and goes through the problems obtained by the time decomposition scheme either once or until it reaches a local optimum. For example, if time window is set to five for an instance with 15 periods, in one iteration (ℓ) of the Fix-and-Optimize heuristic, three problems are solved. The implementation of the Fix-and-Optimize heuristic to solve such a problem is illustrated in Figure 5.1. Each temporary solution to a problem yields an objective value of Z which is at least as good as the current best solution (Z^{old}). So, a new solution is accepted only if it yields an objective value better than the current best solution.

It should be noted that a capacity infeasible solution is never considered as a candidate for the best solution.

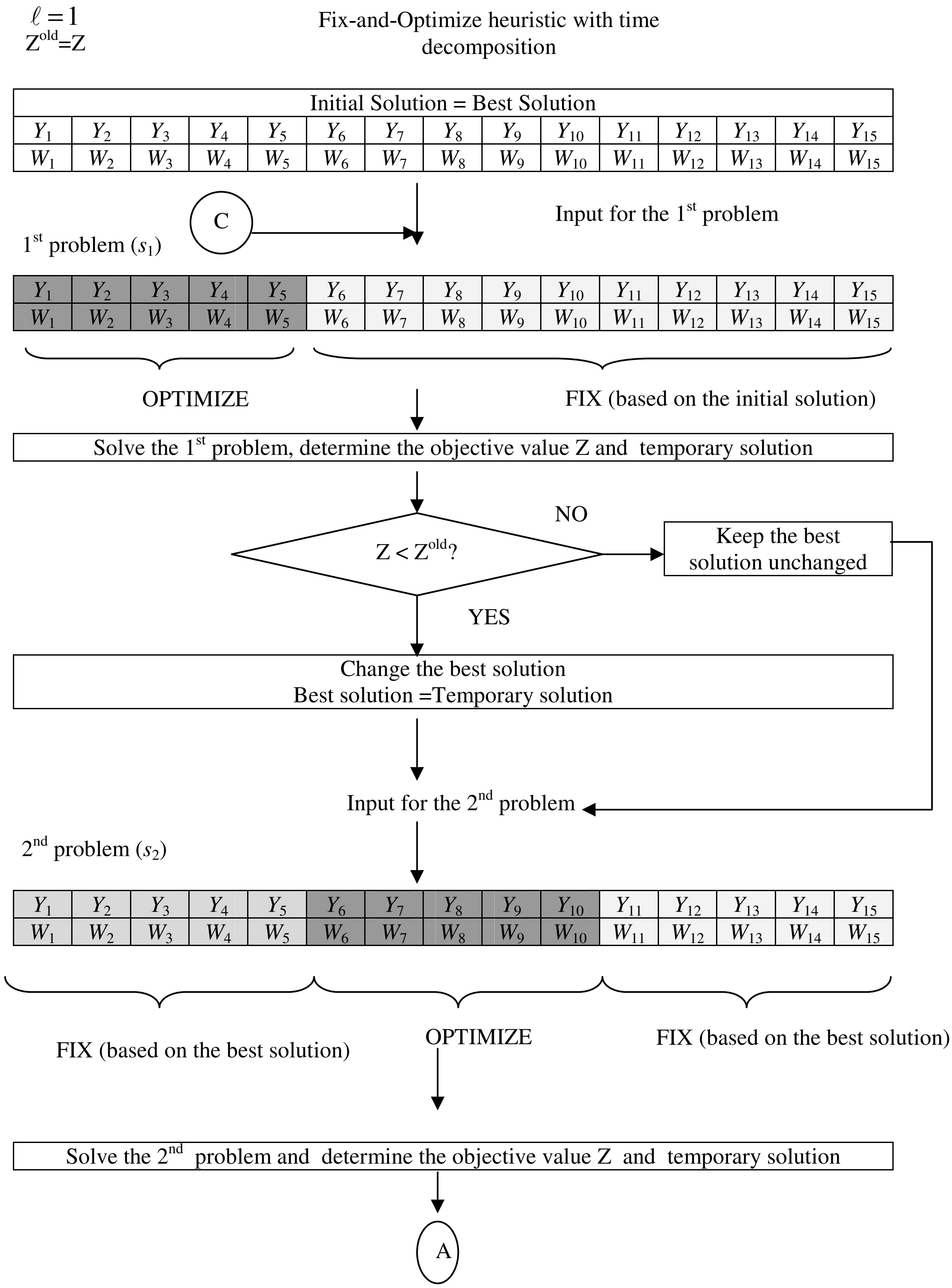


Figure 5.1 The outline of the Fix-and-Optimize heuristic with time decomposition

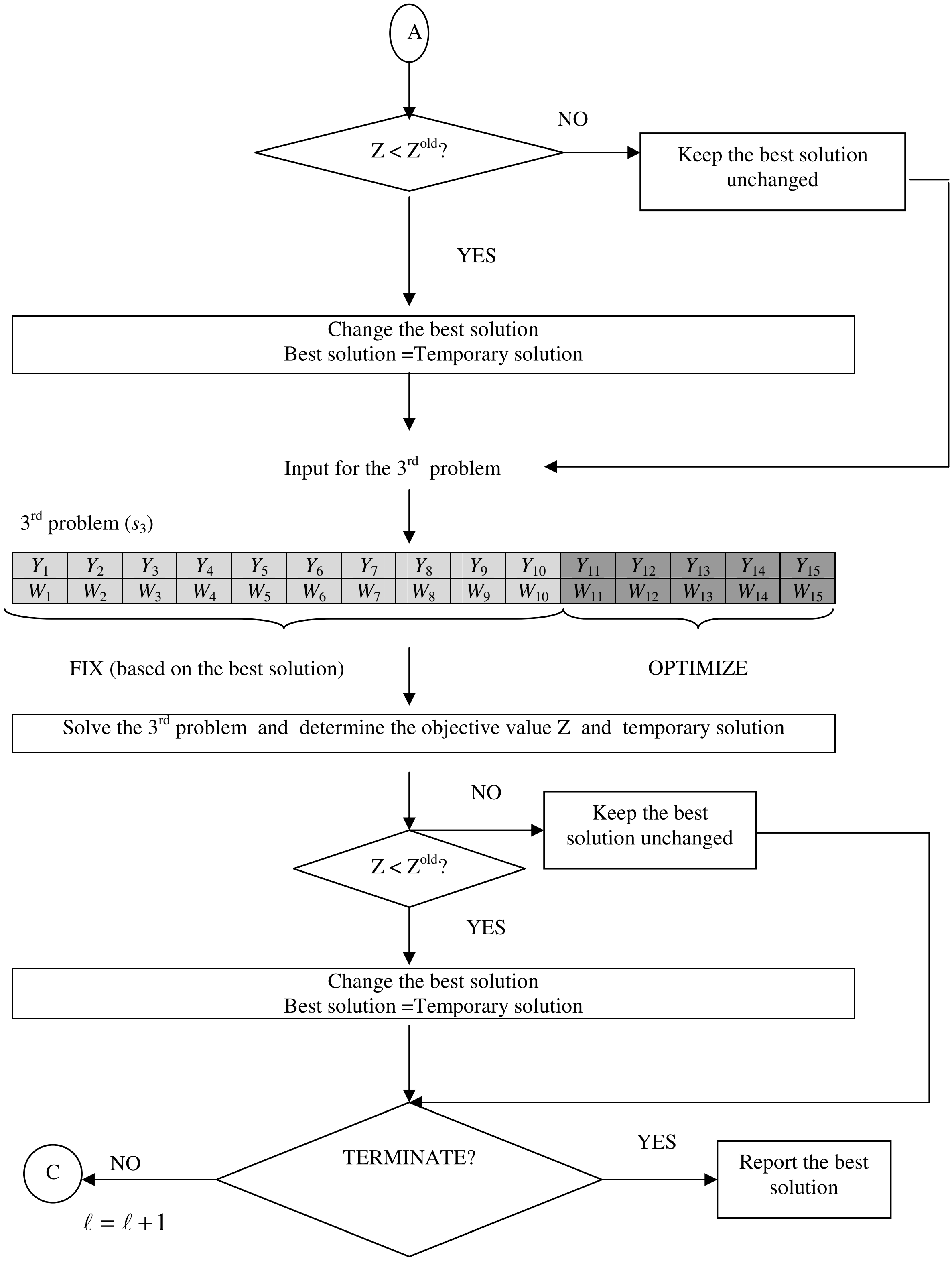


Figure 5.1 The outline of the Fix-and-Optimize heuristic with time decomposition (cont.)

There are two issues affecting the quality of the solution obtained by the Fix-and-Optimize heuristic:

- How to generate the initial solution?
- What should be the number of iterations ($\ell_{\max}$)?

Instead of using a random initial solution like in Sahling et al. (2009) and Helber and Sahling (2010), in this study, the initial solution of the Fix-and-Optimize heuristic comes from GA in both proposed hybrid approaches.

5.3.2.4 Chromosome Representation

The first and most important step in applying GAs to a particular problem is to convert solutions (individuals) of lot sizing problem into a string type called chromosome. In this study, encoding the proposed model involves the binary decision variables Y_{jt} and W_{jt}. Hence, a chromosome is a binary string type. A matrix representation is used for representing a chromosome. The matrix consists of two rows in which the first row shows the setup variables (Y_{jt}) and the second row shows the setup carryover variables (W_{jt}). The columns of the matrix show the products and periods. Thus, the length of the chromosome is the number of periods (P) multiplied by the number of products (K). Figure 5.2 shows a chromosome for *four* products over a planning horizon of *three* periods.

Period 1				Period 2				Period 3			
Y_{11}	Y_{21}	Y_{31}	Y_{41}	Y_{12}	Y_{22}	Y_{32}	Y_{42}	Y_{13}	Y_{23}	Y_{33}	Y_{43}
W_{11}	W_{21}	W_{31}	W_{41}	W_{12}	W_{22}	W_{32}	W_{42}	W_{13}	W_{23}	W_{33}	W_{43}

Figure 5.2 Chromosome representation

5.3.2.5 Initial Population

The search in GA starts from an initial population. It is well known that the performance of the meta-heuristics is affected by the choice of the initial solution/solutions (Gören et al., 2011; Mohammadi & Ghomi, 2011). If the initial solution/solutions is/are good enough, the probability of finding better solutions will increase and the convergence to the near-optimal or optimal solution will be more quickly. The initial population of a GA has a special role on the performance since all the populations in the iterative search process depend, to some extent, on the preceding solution and, eventually on the initial population (Maaranen et al., 2004). However, we noted during the review of the current relevant literature that the initialization of the population has not gained much attention of the researchers in lot sizing area. Hence, to further improve the performance of the proposed hybrid approach a novel initialization scheme is suggested in this study.

The proposed initialization is implemented in two steps to determine the first row (i.e. setup variables) and the second row of the chromosomes (i.e. setup carryover variables), respectively (see Figure 5. 3).

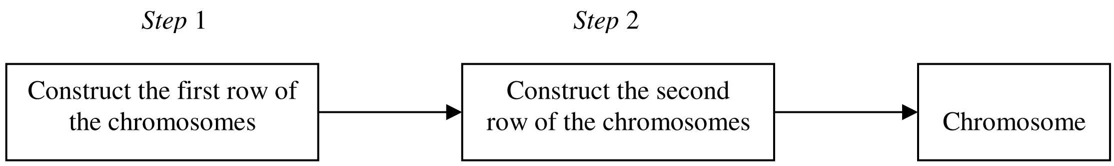


Figure 5.3 The control logic of creating a chromosome

5.3.2.5.1 Generating the Setup Variables. The main idea of the proposed initialization scheme is to use problem-specific information in generating some portion of the initial population. This part of the initial population is called Smart Part. More specifically, in order to generate problem specific chromosomes so that the search can be directed toward the search spaces where feasible and good quality solutions exist, we suggest utilizing the information gained from the LP relaxation of the CLSP with setup times. The procedure followed in creating the Smart Part is given in Figure 5.4.

SMART PART

POP: The population size

λ : *The number of problem-specific chromosomes in the population*

ε : Threshold

Step 1. Solve the LP relaxation of the problem using the Simple Plant Location formulation to get a good lower bound.

Set the cut off point to a specific value ε .

FOR *i*=1 to λ

FOR *k*=0 to (*K***P*)

Step 2. If the value of the setup variable in the LP relaxation is under ε , set *Y* (the value of the gene *k*) in chromosome *i* to 0 otherwise set it to 1.

Step 3. Increase the cut off point by ε .

Figure 5.4 Procedure to generate the *Smart Part*

It should be noted that ε giving the level of *Smart Part* in the initial population is a threshold parameter. An experimental study is made in section 5.4 to evaluate different levels of this threshold parameter. Hoping that the population will be more diverse and initiate a more effective search, we suggest also using some randomness in forming the initial population. After the *Smart Part* is generated based on the value of ε, the *Random Part* is generated following the procedure given in Figure 5.5.

RANDOM PART

POP: The population size

λ : *The number of problem-specific chromosomes in the population*

β : Threshold

FOR *i*= $(\lambda+1)$ to *POP*

FOR *k*=0 to (*K***P*)

Step 1. Pick a random number between 0 and 1.

Step 2. If the random number is less than or equal to β , set *Y* (the value of the gene *k*) in chromosome *i* to 0, otherwise set it to the value of 1.

Figure 5.5 Procedure to generate the *Random Part*

5.3.2.5.2 Generating the Setup Carryover Variables. Once the setup variables are generated, next the setup carryover variables are generated by following the logic given in Figure 5.6. The constraint (10) in the MIP model given in section two suggests that there cannot be any setup carryovers in the first period; therefore the procedure creating the setup carryover variables starts from the second period. Assuming the products are ordered according to decreasing setup costs, the logic behind creating the setup carryover is to check the setup and setup carryover variables in the previous period along with the setup costs of the products. If there is a setup and no setup carryover in the previous period, then a setup carryover can be assigned for the product in that period. Due to the infeasibility that might occur, the setup variable in the relevant period is also checked. In case there is a setup, this setup should be eliminated if a setup carryover is already assigned in this period.

```
FOR i=1 to POP
   FOR t=2 to P
    FOR j=1 to K (Assume products are ordered according to decreasing setup costs)
       IF Y_jt-1=1 and W_jt-1=0,
       THEN
       Set W_jt=1
               IF Y_jt=1
               THEN
               Set Y_jt=0
               ENDIF
       ENDIF
       BREAK (go to the next period)
```

Figure 5.6 Procedure for the second row (i.e. setup carryover variables) in the initial population

5.3.2.6 Fitness Function

Following the generation of the initial population, the fitness of each individual in the population is calculated by employing an appropriate objective function. The objective function provides a measure of a chromosome’s performance or fitness in the search spaces. In this study, the fitness of each individual is calculated based on the sum of the objective function of the MIP model presented. The total cost is the sum of the inventory and setup costs which are calculated by fixing the setup variables in the sub-problem and optimizing the remaining continuous variables. There might be some chromosomes that violate the capacity constraint during the search. In order to deal with this infeasibility, a penalty cost is added to the total cost as shown in (14).

Total cost= Setup cost + Inventory cost + Penalty cost (14)

5.3.2.7 Selection Scheme

After calculating the fitness of each chromosome in the initial population, potential parents are selected using the *modified roulette wheel selection*. Roulette wheel selection scheme scales the fitness values of the members within the population so that the sum of the rescaled fitness values is equal to 1. Based on pilot experiments it was observed that the performance of the roulette wheel selection operator was rather poor for our CLSPC. We noticed that chromosomes having low fitness values were never selected for crossover. This reduced the chance to introduce diversity into the new solutions and affected the solution quality. To deal with this issue, we added the ranking issue to the *roulette wheel selection operator*. The pseudo-code for this *rank-based roulette wheel selection operator* is given below.

Step 1. Calculate the total cost of each chromosome.

Total cost of chromosome n = Setup cost + Inventory cost + Penalty cost

Step 2. Order the chromosomes according to their total cost in an ascending order and give a rank to each chromosome. This rank shows the fitness of the chromosome.

Step 3. Calculate the selection probability of each chromosome by its rank, n, based on the following equation (Haupt & Haupt, 2004)*:*

$$selection\ probability\ of\ chromosome\ n = \frac{population\ size - n + 1}{\sum_{n=1}^{population\ size} n} \qquad (15)$$

for i=0 to (population size-1)

Step 4. Generate a random number between 0 and 1.

Step 5. Starting from the top of the list, choose the first chromosome with a cumulative probability that is greater than the random number.

This process continues until all the parents are selected.

5.3.2.8 Genetic Operators

Genetic operators such as crossover and mutation are used in order to explore the search space. The crossover operator combines the chromosomes selected by the selection scheme into a new chromosome. The mutation operator is used to maintain the diversity in the population.

A new population is produced using the *single point crossover* and *single bit flip mutation* operators. The crossover probability rate, R_c, is used to determine whether the offspring represents a blend of the parents. If no crossover takes place according to the crossover probability, the two offsprings become the clones of their parents. But, if crossover occurs, a random crossover point is selected and the parts after the crossover point are exchanged between two parents.

Similar to the crossover operation, mutation operation occurs during the evolution according to a specified mutation probability rate defined as R_M. This probability rate is used to decide how often gene values of the chromosomes will be mutated. Based on the R_M, we consider the setup and carryover variables together and apply the single bit flip mutation operator based on these variables. In the single bit flip mutation operator, the value of 1 is changed to the value of 0 and vice versa. The mutation operator is detailed in Figure 5.7.

Input: A gene showing the value of setup variable for a specific product (a) and period (b) randomly

```
IF Y_ab=0
        IF W_ab=0
        THEN Set Y_ab=1
        ENDIF
        ELSE Set W_ab=0 and Y_ab=1
                Introduce a setup carryover for product i (i≠a) in period b.
                W_ib=1
ELSE
        IF W_ab+1=0
        THEN Set Y_ab=0
                IF Y_ab-1=1
                THEN W_ab=1
                        IF there are 2 setup carryovers in period b
                        THEN Call the REPAIR Operator
                        ENDIF
                ENDIF
        ENDIF
```

Output: Mutated gene

Figure 5.7 The pseudo-code for the single bit flip mutation operator

5.3.2.9 Repair Operators

After employing the crossover and mutation operators, there might be infeasible chromosomes that do not satisfy the constraints (4), (5), (6) and (7) of the model stated in Section 5.2.2. It should be noted that infeasible chromosomes violating these constraints are not allowed in the population. In this study, the proposed repair operators are grouped into two and the details of each operator are given in the following.

1.<u>The repair operator used after crossover</u>

The single point crossover operation might create some infeasible chromosomes while producing offspring. Therefore; a number of repair operators are proposed to fix these infeasibilities. There are four kinds of repair operators in this group. The first operator is applied after the crossover operator in order to maintain the feasibility. The second one removes the setup carryovers if there are two or more setup carryovers in one period and the third one introduces a setup carryover if there is no setup carryover in the current period. The last one checks the carryover and setup variables in the previous period and repairs the infeasibilities that occur due to this situation.

<u>Crossover repair operator:</u> This repair operator corrects the infeasibilities by looking at the period in which crossover takes place. Based on the place of the crossover point, this operator looks at the following and the previous periods of the period in which crossover occurs. The details are given in Figure 5.8.

Input: Crossover point (k) indicating a specific time period and a product, an infeasible chromosome violating constraints (4), (5) and (6).

```
FOR Period t
        IF k=1 (the crossover point is at the beginning or end of a period)
        THEN
        FOR j=1 to K
                IF W_jt=1
                THEN
                Look at the previous period.
                        IF W_j,t-1=1
                        THEN
                         Check for single production in period t-1.
                                IF there is no single item production in period
                        t-1,
                                THEN
                                Set W_jt=0
                                 Set Y_jt=1
                                ENDIF
                        ENDIF
                        ELSE
                        Call A SETUP CARRYOVER WITHOUT A
                        SETUP IN THE   PREVIOUS PERIOD
                ENDIF
        ENDIF
```

Figure 5.8 The pseudo-code of the crossover repair operator

```
ELSE

    FOR j=1 to K
            IF (j < k)
               IF W_jt=1
                    IF W_jt-1=1
                       IF there is no single item production in period t-1,
                       THEN
                       Set W_jt=0
                       Set Y_jt=1
                       ENDIF
                    ENDIF
                    ELSE
                    Call A SETUP CARRYOVER WITHOUT A SETUP
                    IN THE   PREVIOUS PERIOD
              ENDIF
            ENDIF

            ELSE
             IF W_jt=1
              IF W_jt+1=1
                    IF there is no single item production in period t,
                           THEN
                           Set W_jt=0
                           Set Y_jt=1
                    ENDIF
              ENDIF
            ENDIF
IF ∑W_jt=0
THEN call HIGHEST SETUP REPAIR OPERATOR
IF ∑W_jt>=2
THEN call MORE THAN 2 SETUP CARRYOVER IN A PERIOD REPAIR
OPERATOR
```

Output: A feasible chromosome satisfying constraints (4), (5) and (6).

Figure 5.8 The pseudo-code of the crossover repair operator (cont.)

<u>More than 2 setup carryover in a period repair operator:</u> If there are two setup carryovers or more in a period, this operator keeps the setup carryover of the product with the highest setup cost (if there is a setup in the previous period for this product) and removes other setup carryovers. Figure 5.9 presents the details.

```
Input: An infeasible chromosome violating constraints (4).
 FOR Period t
          IF  ∑ W_jt ≥ 2
           FOR j=1 to K (start with the product with the lowest setup cost)
            IF W_jt=1
            THEN
             Set W_jt=0
              IF ∑ W_jt =1
              THEN BREAK!
Output: A feasible chromosome satisfying constraints (4).
```

Figure 5.9 The pseudo-code of the "More than 2 setup carryover in a period" repair operator

Highest setup repair operator: If there is no setup carryover in a period, this operator introduces a setup carryover for a product with the highest setup cost and without a setup in the previous period (Y_{jt-1}=1) as shown in Figure 5.10.

```
Input: A chromosome without setup carryover in a specific period.
FOR Period t
          FOR j=1 to K (start with the product with the highest setup cost)
           IF Y_jt=1
            IF Y_jt-1=1
            THEN
             Set W_jt=1
             Set Y_jt=0
             BREAK!
Output: A chromosome with setup carryover in a specific period.
```

Figure 5.10 The pseudo-code of the "Highest setup" repair operator

A setup carryover without a setup in the previous period: If there is a setup carryover in the current period without a setup in the previous period, this operator deals with this case. The details of the operator are shown in Figure 5.11.

```
Input: An infeasible chromosome violating constraints (5).
  FOR Period t
    IF W_jt=1
          IF Y_jt-1=0
          THEN
          Set W_jt=0
          ENDIF
    ENDIF
Output: An infeasible chromosome satisfying constraints (5).
```

Figure 5.11 The pseudo-code of the “A setup carryover without a setup in the previous period” repair operator

2. <u>The repair operators used after mutation</u>

Similar to the crossover operator, the mutation operator might lead to some infeasibility. To correct these, a repair operator is proposed. This operator is developed by modifying the repair operator which is used after the crossover operator. The only difference here is that in this operator the setup cost is not taken into consideration in removing the setup carryovers.

<u>Repair operator for more than 2 setup carryovers in a period (for mutation):</u> If two or more setup carryovers occur in a period (i.e. period b) after a mutation operator is applied this operator keeps the setup carryover of the product (i.e. product a) that has just been assigned after mutation and removes the other setup carryover in the related period (i.e. period b). The pseudo-code of the operator is given in Figure 5.12.

```
Input: An infeasible chromosome violating constraints (4).
 FOR Period b
            IF ∑ W_jb ≥ 2
             FOR j=1 to K & j ≠ a
              IF W_jb=1
              THEN
              Set W_jb=0
               IF ∑ W_jb =1
               THEN BREAK!
Output: A feasible chromosome satisfying constraints (4).
```

Figure 5.12 The pseudo-code of the "More than 2 setup carryover in a period (for mutation)" repair operator

5.3.2.10 Survival Scheme

Survival is an essential process in GAs that removes individuals with a low fitness and drives the population towards better solutions. Following the Fix-and-Optimize heuristic, a part of the existing population survives and forms a new population in the next generation. An elitist strategy, which ensures the best solution of the previous generation into the current generation, is used in this study.

5.3.2.11 GA Search Termination

Termination criterion is the last decision point by which the GA decides whether to continue searching the search space or stop evolution. In this study, the genetic search stops when the total number of generations exceeds a maximum number.

5.3.3 Proposed Hybrid Methodologies

In this section, two hybrid approaches are proposed to solve the CLSPC. The first hybrid approach is a kind of sequential hybrid approach where the GA is run for a predetermined number of generations and then Fix-and-Optimize heuristic is employed to further improve the solution quality. In the second hybrid approach, the Fix-and-Optimize heuristic is embedded into the GA. In the framework of these hybrid approaches, the original algorithms of the GA and Fix-and-Optimize remain unchanged.

5.3.3.1 Methodology of the First Hybrid Approach

The first hybrid approach combines GA and Fix-and-Optimize heuristic in a sequential way. First, GA and then the Fix-and-Optimize heuristic are performed. Figure 5.13 illustrates the control logic of the first hybrid approach.

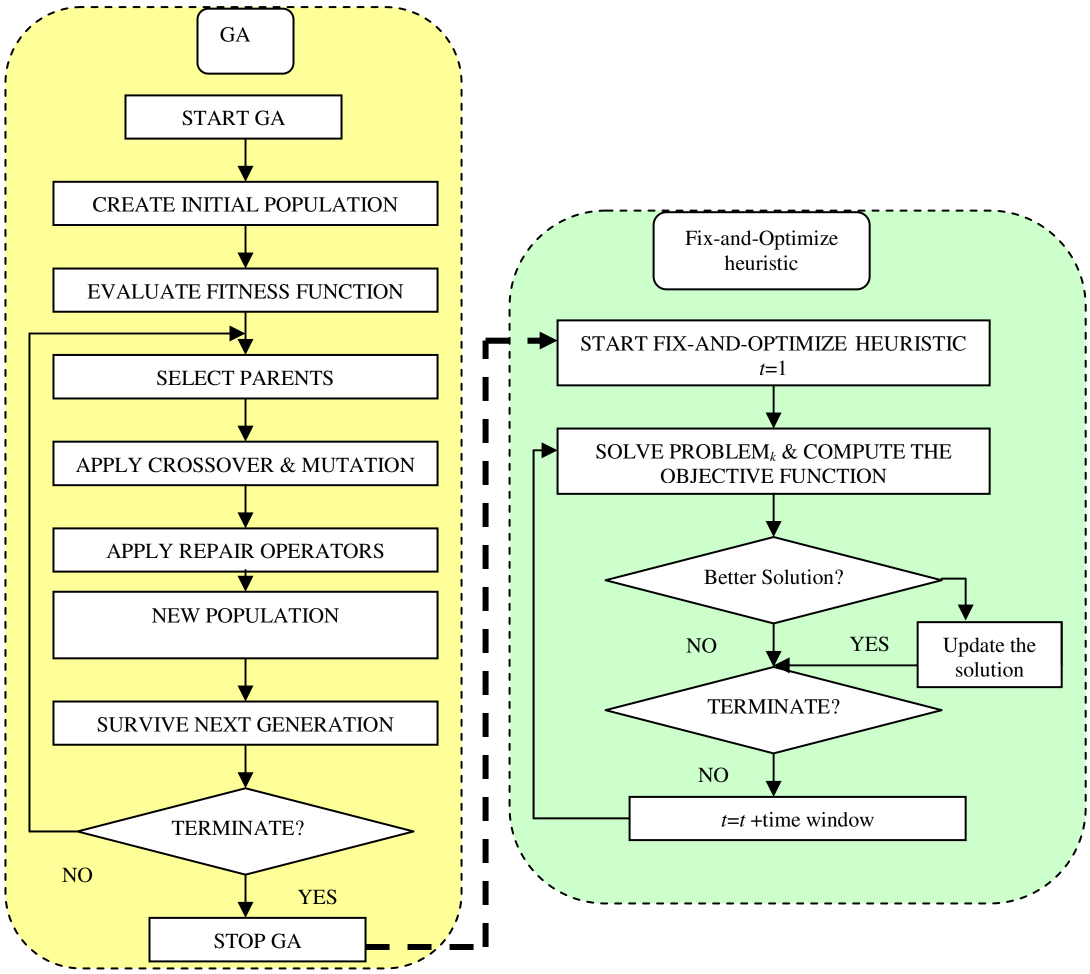


Figure 5.13 The control logic of the sequential hybrid approach

5.3.3.2 Methodology of the Second Hybrid Approach

The second hybrid approach is an example of embedded hybridization where the Fix-and-Optimize heuristic is used to improve the solution quality of each generation in GA. Figure 5.14 shows the control logic of the the *embedded hybrid approach with Fix-and-Optimize refinement*.

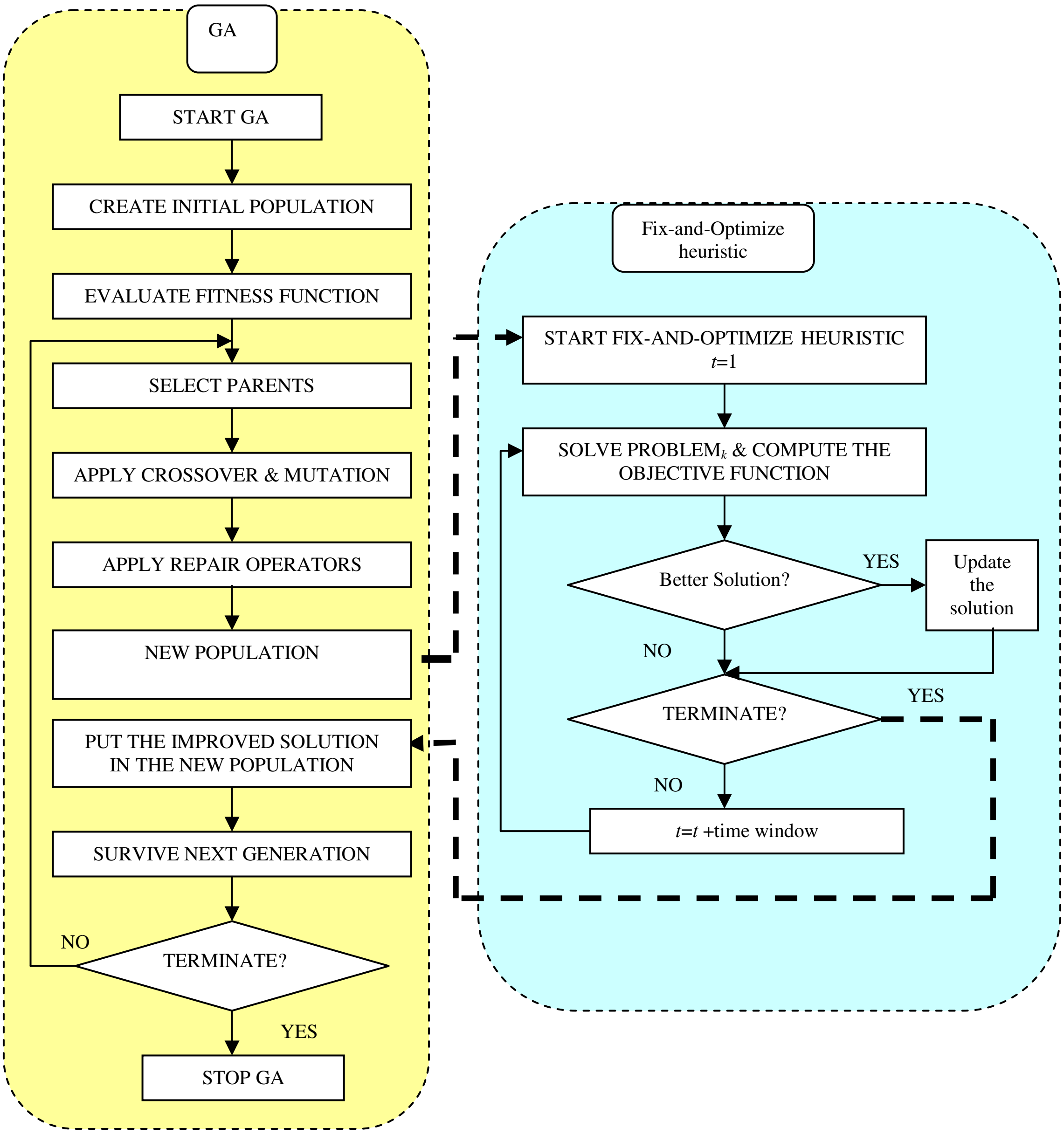


Figure 5.14 The control logic of the embedded hybrid approach

After the chromosomes are evaluated, selection, crossover and mutation take place (See Figure 5.14). In order to deal with the infeasibilities, appropriate repair operators are called. As indicated in Figure 5.14, before determining the survival of the individuals into the next generation, an individual is randomly selected from the new population and it is used as an initial solution in the Fix-and-Optimize heuristic. In doing so, a Fix-and-Optimize heuristic is used to refine the solutions, which are generated by GAs and it acts like a diversification strategy which leads the search to the regions where near optimal or optimal solutions exist. Particularly, by employing the Fix-and-Optimize heuristic, we hope to improve the solution quality of the GA-generated solutions in every generation.

5.4 Computational Experiments

Computational experiments involve three sets of experiments. First, the effects of the initialization scheme on the performance of the pure GAs were evaluated by employing three groups of problems (i.e. small, medium and large size). Next, using these three groups of problems, optimal parameter settings for GAs were identified. Finally, the performance of pure GAs and proposed hybrid approaches were evaluated using some benchmark problems reported in the literature. All the results obtained were compared to those of Gopalakrashinan et al. (2001) and Suerie and Stadtler (2003), the latest results reported in the literature. All computations were carried out on a PC with Dual Core, 2 GHz microprocessor and 2 GB RAM. The pure and hybrid GAs were coded in Visual C++ 2008 Express Edition and all problems were solved using Concert Technology of Cplex 11.2.

5.4.1 Benchmark Problems

Trigeiro et al. (1989) proposed an algorithm (referred to as TTM, Trigeiro-Thomas-McClain) to solve the CLSP and they generated various test instances to test the performance of this algorithm. These test instances were generated in three phases. Phase one involved 70 problem instances which were used for fine-tuning of the parameters of the TTM algorithm. In phase two, 141 problem instances were used to analyze different problem characteristics, and in phase three, 540 problem instances were generated to test the algorithm. However, no results have been reported in the literature for problem instances in phase one. Hence, in this study, to evaluate the performance of the proposed hybrid approaches only the problem instances generated in phase two and three (i.e. total of 681 instances) were used. Table 5.2 gives the classification of these instances (Suerie & Stadtler, 2003).

Table 5.2 Classification of TTM-test set (Suerie & Stadtler, 2003).

Phase II			
Class	# Items	# Periods	# Instances
1	6	15	116
2	6	30	5
3	12	15	5
4	12	30	5
5	24	15	5
6	24	30	5
Phase III			
7	10	20	180
8	20	20	180
9	30	20	180

Based on the number of binary variables (number of periods*number of products) which is considered as a measure of the problem complexity, these problem classes have been placed into three groups, small, medium and large (See Table 5.3).

Table 5.3 Problem sizes

Small	Medium	Large
▪ Class 1 ▪ Class 2 ▪ Class 3 ▪ Class 7	▪ Class 4 ▪ Class 5	▪ Class 6 ▪ Class 8 ▪ Class 9

5.4.2 Investigating the Proposed Initialization Scheme

The effects of the proposed initialization scheme were investigated on three groups of problems (small, medium, large). Five problem instances randomly picked from each problem class (see Table 5.3) were solved using the pure GAs with the initial population created by the proposed initialization scheme. As stated earlier in the previous section, the initial population consists of two parts: random and smart part. One of the design issues in these computational studies is to evaluate the effects of the ratio of smart part to random part on solution quality. The experimental design used in these computations is given in Table 5.4. The other design parameter is the threshold value (β) used in creating the random part of the initial population. Based on some pilot experiments, we set the threshold value (β) to 0.5 in creating the random part. Throughout all experiments the same parameters were used (i.e., combination of population size and number of generations, the crossover rate and the mutation rate were set to 20/500, 0.9 and 0.011, respectively) and the results of 10 runs using different random seeds were reported. Therefore, 20 instances for small size problems (200 runs in total), 10 instances for medium size problems (100 runs in total) and 15 instances for large size problems (150 runs in total) were solved.

Table 5.4 Experimental design for the portion of random and smart part in the initial population

	Random Part %	Smart Part %	Abbreviated
Pure Random initial population	100	0	R
Random initial population+ Smart initial population :	90	10	RS^{10}
Random initial population+ Smart initial population:	80	20	RS^{20}
Random initial population+ Smart initial population :	70	30	RS^{30}
Random initial population+ Smart initial population :	60	40	RS^{40}
Random initial population+ Smart initial population :	50	50	RS
Random initial population+ Smart initial population :	40	60	RS^{04}
Random initial population+ Smart initial population :	30	70	RS^{03}
Random initial population+ Smart initial population:	20	80	RS^{02}
Random initial population+ Smart initial population:	10	90	RS^{01}
Pure Smart initial population:	0	100	S

The results of computational experiments are summarized in Table 5.5. It should be noted that the computational time is not taken into consideration in comparisons since it is observed to be similar across all problem sizes (i.e. small, medium, large).

In Table 5.5, the results given in parenthesis show the number of infeasible solutions obtained throughout the runs. It should be noted that while calculating the average gap only feasible solutions are taken into consideration. The gap is calculated based on the formula given in the following where the lower bounds are taken from the study of Suerie and Stadtler (2003).

$$Gap = 100 * \frac{(heuristic\ solution - lower\ bound)}{lower\ bound} \tag{16}$$

As seen from the table, using pure random initial population results in large number of infeasible solutions, i.e., 150 infeasible solutions out of 200 runs for small size problems and for large problems, it is not even possible to find a feasible solution. It can be seen from Table 5.5 that as the problem size increases, the number of feasible solutions found during 10 runs of the pure GAs with a random initial population decreases.

Table 5.5 Results of Comparative Experiments

	SMALL		MEDIUM		LARGE	
	Avg. gap (average of 10 runs)	Avg. gap (best of 10 runs)	Avg. gap (average of 10 runs)	Avg. gap (best of 10 runs)	Avg. gap (average of 10 runs)	Avg. gap (best of 10 runs)
R	10.11 % (150)	8.64 % (150)	13.65 % (95)	12.92 % (95)	* (150)	* (150)
RS^{10}	6.36 %	5.48 %	6.56 % (2)	5.71 %	3.04 %	2.87 %
RS^{20}	6.31 %	5.30 %	5.56 %	4.97 %	3.01 %	2.80 %
RS^{30}	6.34 %	5.24 %	5.51 %	5.04 %	3.03 %	2.89 %
RS^{40}	6.38 %	5.36 %	5.49 %	4.98 %	2.97 %	2.83 %
RS	6.29 %	5.45 %	**5.25 %**	**4.80 %**	2.99 %	2.81 %
RS^{04}	6.33 %	5.34 %	5.36 %	4.90 %	**2.93 %**	**2.81 %**
RS^{03}	6.26 %	5.14 %	5.43 %	4.84 %	2.97 %	2.84 %
RS^{02}	**6.17 %**	**5.28 %**	5.30 %	4.89 %	2.97 %	2.82 %
RS^{01}	6.30 %	5.30 %	5.30 %	4.95 %	2.97 %	2.83 %
S	6.20 %	5.34 %	5.26 %	4.88 %	2.97 %	2.83 %

* indicates that no feasible solution is obtained under this initialization scheme

As stated in Section 5.3.2.5, the proposed initialization scheme suggests creating a smart part in the initial population by utilizing the information obtained from the LP relaxation of the CLSP which is the basis of the CLSPC. It is hoped that this information will help to generate problem specific chromosomes so that the search will be directed towards the search spaces where feasible and good quality solutions exist. As seen from Table 5.5, the results of experimental studies support our expectation and when some problem specific information is added into the initial population generation scheme, GA finds feasible solutions in nearly all runs for all problem sizes. For all problem sizes, while the worst performance is obtained with pure random initial population (R), the second best performance is obtained with pure smart initial population (S). It is also quite clear from Table 5.5 that for all problem sizes, generating some portion of the initial population randomly creates diversity in the initial population and this further improves the performance of GAs in solving the CLSPC. However, when both randomness and problem specific information are included in the initial population, the ratio of random and smart part giving minimum average gap changes for each problem size i.e. RS^{02} for small size problems, RS for medium size problems and RS^{04} for large size problems. Hence, based on the experimental results given in Table 5.5, it is not possible to draw a general conclusion regarding the ideal ratio of random and smart part in the initial population. However, in overall, we could state that keeping the portion of the smart part bigger than the portion of the random part might improve the performance of the proposed initialization scheme.

Based on these results, it can be stated that the proposed initial population generation method using problem specific information scheme has high potential to reduce the number of infeasible solutions in each generation and hence it can ensure a feasible solution at each run of the GAs. Another insight gained as a result of these experimental studies is that generating some part of the initial population randomly further improves the performance of the proposed initialization scheme. However, how to set the ratio of random part to smart part in the initial population seems to be a problem dependent parameter, hence, it is suggested to determine this parameter experimentally. Based on the computational experiments carried out in this study, this parameter is set to RS^{02}, RS and RS^{04} (see Table 5.4) for small, medium and large size problems, respectively.

5.4.3 Identifying Efficient GAs Parameters

It should be noted that identifying best search parameters which are used throughout the search process is an important step of any meta-heuristic application. So this section focuses on a critical dilemma faced in many GA applications; the selection of the efficient GA parameters to ensure high performance. The parameters of a GA namely the population size (*P*), number of generations (*G*), crossover rate (*%C*) and mutation rate (*%M*) significantly affect the convergence speed of the GAs. In order to determine the most efficient parameters, in this section a set of experiments was performed for three different problem sizes. All these experiments were carried out using the pure GAs with the proposed initial population generation scheme.

It is known that large populations and many generations which imply large number of chromosomes can result in high quality solutions if computational time is unlimited. However, it is not practical to assume unlimited computational time. In this study, like in Pongcharoen et al. (2002) the computational time is limited by fixing the total number of chromosomes at 10000. As seen in Table 5.6, all parameters have been varied in three levels. It should be noted that for all problem sizes, the mutation rate is set to the inverse of the chromosome length (Khouja et al., 1998). Since chromosomes' lengths change according to the number of products and periods in the problem, different mutation rates are used for each problem size. The overall objective is to use the most efficient GA parameters that achieve the minimum total cost and minimum spread (Pongcharoen et al., 2002). Experiments involved five problem instances from each problem size and a full factorial design with 10 runs were carried out at each design point leading to 270 runs in total.

Table 5.6 Experimental factors and levels for each problem size

Factors	Levels	Problem size		
		Small	Medium	Large
P/*G*	1	20/500	20/500	40/250
	2	50/200	50/200	100/100
	3	100/100	100/100	200/50
%*C*	1	0.5	0.5	0.5
	2	0.7	0.7	0.7
	3	0.9	0.9	0.9
%*M*	1	0.005	0.0006	0.001
	2	0.011	0.003	0.0025
	3	0.05	0.015	0.005

In order to determine which control parameter effects are significant, a statistical analysis of variance (ANOVA) is conducted for each problem size and the results are presented in Tables 5.7, 5.8 and 5.9.

Table 5.7 ANOVA results for small size problems

Source of variation	DF	F_{calc}	Prob[$F>F_{calc}$]
Within + residual	26		
P/*G*	2	7.259	0.001*
%*C*	2	0.398	0.672
%*M*	2	1437.659	0.000*
P/*G* & %*C*	4	0.345	0.848
P/*G* & %*M*	4	7.436	0.000*
%*C* & % *M*	4	1.982	0.098
P/*G* & % *C* & % *M*	8	0.843	0.565

Table 5.8 ANOVA results for medium size problems

Source of variation	DF	F_{calc}	Prob[F>F_{calc}]
Within+residual	26	184.980	0.000
P/G	2	113.975	0.000*
%C	2	0.858	0.425
%M	2	2093.087	0.000*
*P/G**%C*	4	1.070	0.372
*P/G**%M*	4	89.555	0.000*
*%C**%M*	4	2.418	0.049*
*P/G**%C*%M*	8	2.683	0.008*

Table 5.9 ANOVA results for large size problems

Source of variation	DF	F_{calc}	Prob[F>F_{calc}]
Within+residual	26	105.694	0.000
P/G	2	503.799	0.000*
%C	2	0.753	0.472
%M	2	569,016	0.000*
*P/G**%C*	4	2.828	0.025*
*P/G**%M*	4	144.286	0.000*
*%C**%M*	4	1.760	0.138
*P/G**%C*%M*	8	0.677	0.712

As stated in Table 5.7, the combination of population size and number of generations (*P/G*), the mutation rate (*%M*) and the interaction between the mutation rate and the combination of population size and number of generations (*P/G*%M*) are statistically significant factors for small size problems. For medium size problems, the combination of population size and number of generations (*P/G*), the mutation rate (*%M*), the interaction between the mutation rate and the combination of population size and number of generations (*P/G*%M*), the interaction between the crossover rate and mutation rate and the three way interaction (*P/G*%C*%M*) are all statistically significant factors (see Table 5.8). Finally, the results in Table 5.9 show that the combination of population size and number of generations (*P/G*), the mutation rate (*%M*), the interaction between the mutation rate and the combination of population size and number of generations (*P/G*%M*) and the interaction between the combination of population size and number of generations and crossover rate (*P*/*G**%*C*) are statistically significant factors for large size problems.

Having identified the mutation rate as a statistically significant factor for all problem sizes, in scatter plots below (Figures 5.15, 5.16, and 5.17) we summarized the results of experiments based on different values of mutation rate.

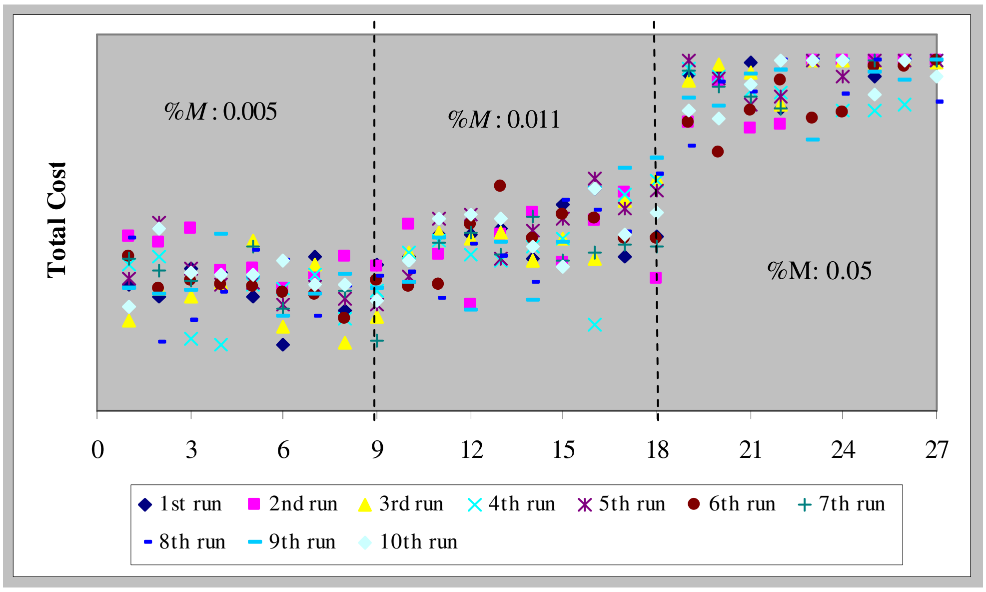


Figure 5.15 Scatter plot of total cost from ten runs for small size problems

While observing the scatter plots, the parameters which yielded the lowest total cost and smallest spread between the results were chosen and these parameter values were defined as the efficient GA parameters. Based on the observations, it is possible to recommend the values of *P*/*G*=50/200, %*C*=0.9 and %*M*=0.005 (see Figure 5.15); the values of *P*/*G*=20/500, %*C*=0.5 and %*M*=0.003 (see Figure 5.16); the values of *P*/*G*=100/100, %*C*=0.9 and %*M*=0.001 (see Figure 5.17) as efficient control parameters for small, medium and large size problems, respectively.

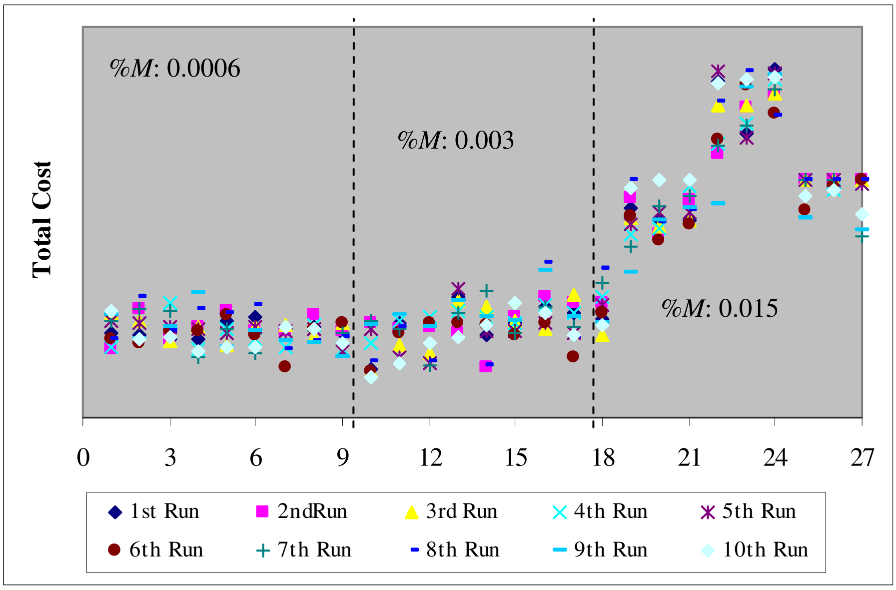


Figure 5.16 Scatter plot of total cost from ten runs for medium size problems

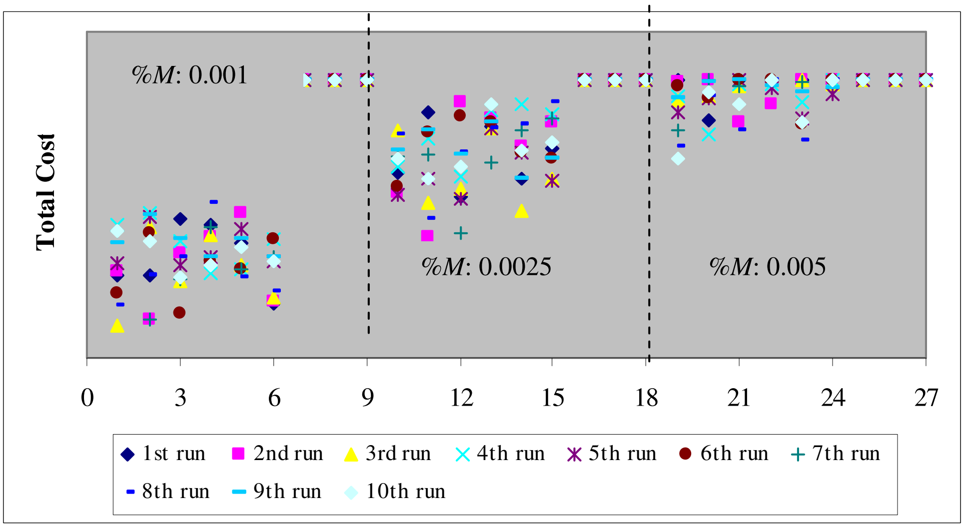


Figure 5.17 Scatter plot of total cost from ten runs for large size problems

The efficient parameter settings for each problem size can be summarized in Table 5.10.

Table 5.10 Parameter settings for each problem size

	Small	Medium	Large
P/*G*	50/200	20/500	100/100
%*C*	0.9	0.5	0.9
%*M*	0.005	0.003	0.001

5.4.4 Analysis and Discussion of the Results

This section presents the results of the computational studies comparing the performance of the pure GAs and proposed hybrid approaches to the recent results reported in the literature. It should be noted that the problem instances generated in phase two and three (i.e. total of 681 instances) were used to evaluate the performance of the proposed approaches and pure GAs which are named as H1, H2 and PGA, respectively. The experiments were repeated 10 times for every instance, the best and average performance for the solutions were recorded. All the results obtained were compared to those of Gopalakrashinan et al. (2001) and Suerie and Stadtler (2003), the most recent ones for the problem considered in this study.

The results of comparing the pure GAs and the proposed hybrid approaches to the time decomposition heuristic of Suerie and Stadtler (2003) were given in Table 5.11. To obtain compatible test results for each problem class, the computational times for the proposed hybrid approaches were set to the average computational time (i.e. the time calculated based on the computational times of 10 runs) of the pure GAs. For instance, the proposed hybrid approaches were run for 58.75 seconds for each run to solve the problem instances in Class 1 (see Table 5.11). Moreover, as a result of some preliminary tests the computational time to solve each problem using the Fix-and-Optimize heuristic was limited to 2 seconds. In order to improve the computational efficiency of the second proposed hybrid approach, while implementing the second proposed hybrid approach the Fix-and-Optimize heuristic was run only once in each generation of GAs.

Table 5.11 Experimental results for the pure GAs and proposed hybrid approaches using TTM test sets

Class	Suerie and Stadtler (2003)		PGA			H1			H2		
			Average of 10 runs		**Best of 10 runs**	**Average of 10 runs**		**Best of 10 runs**	**Average of 10 runs**		**Best of 10 runs**
	Avg Gap	Avg. Comp. Time	**Avg. Gap**	**Avg. Comp. Time**	**Avg. Gap**	**Avg. Gap**	**Avg. Number of Generations**	**Avg. Gap**	**Avg. Gap**	**Avg. Number of Generations**	**Avg. Gap**
1	2.53%	5.2 s	8.39 %	58.75 s	7.19 %	3.02 %	26.16	2.68 %	3.19 %	14.61	2.54 %
2	2.32%	8.8 s	8.20 %	72.88 s	7.24 %	2.78 %	32.22	2.64 %	2.62 %	12.4	**2.17 %**
3	0.95%	6.2 s	5.31 %	72.65 s	4.71 %	1.87 %	29.1	0.96 %	1.53 %	10.86	1.10 %
4	0.72%	11.8 s	5.13 %	77.13 s	4.66 %	1.56 %	60.78	1.15 %	1.67%	6.08	1.19%
5	0.54%	9.2 s	3.05 %	98.70 s	2.6 %	1.65 %	75.68	1.07 %	1.22%	14.18	0.80%
6	0.30%	13.2 s	2.95 %	144.21 s	2.79 %	2.10 %	16.82	1.50 %	1.65 %	8.1	1.25 %
7	3.14%	9.2 s	4.87 %	48.91 s	4.12 %	**2.10 %**	23.38	**1.71 %**	**2.31 %**	10.46	**1.80 %**
8	2.73%	12.7 s	2.81 %	99.94 s	2.53 %	**1.58 %**	18.84	**1.25 %**	**1.81 %**	12.1	**1.44 %**
9	2.21%	18.0 s	1.95 %	127.74 s	1.75 %	**1.54 %**	15.79	**1.29 %**	**1.48 %**	14.13	**1.25 %**

Table 5.12 Experimental results for the pure GAs and proposed hybrid approaches using TTM test sets (Aggregation of Classes)

Classes	Gopalakrashinan *et al*. (2001) [Pentium III, 550 MHz]		**Pure GAs [Dual Core, 2 GHz]**			**H1 [Dual Core, 2 GHz]**			**H2 [Dual Core, 2 GHz]**		
			Average of 10 runs		**Best of 10 runs**	**Average of 10 runs**		**Best of 10 runs**	**Average of 10 runs**		**Best of 10 runs**
	Avg. Gap	Avg. Comp. Time	Avg. Gap	Avg. Comp. Time	Avg. Gap	Avg. Gap	Avg. Comp. Time	Avg. Gap	Avg. Gap	Avg. Comp. Time	Avg. Gap
1,2	27.8%		**8.38 %**	65.82 s	**7.19 %**	**3.01 %**	59.33 s	**2.68 %**	**3.17 %**	59.33 s	**2.53 %**
3,4	13.9%	20.8	**5.22 %**	74.89 s	**4.69 %**	**1.71 %**	74.89 s	**1.06 %**	**1.36 %**	74.89 s	**1.15 %**
5,6	6.0%		**3.00 %**	121.46 s	**2.70 %**	**1.88 %**	121.46 s	**1.29 %**	**1.23 %**	121.46 s	**1.03 %**
7-9	12.4%	81.7	**3.21 %**	92.20 s	**2.80 %**	**1.74 %**	92.20 s	**1.42 %**	**1.86 %**	92.20 s	**1.50 %**

When compared to the pure GAs, the solution quality of the proposed hybrid approaches is quite remarkable in every problem class. As seen in Table 5.11, the average gaps obtained by hybrid approaches are smaller than those of the pure GAs. Furthermore, as expected, the Fix-and-Optimize heuristic used after GAs in the first hybrid approach improves the solution quality throughout the iterations. Likewise, in the second hybrid approach, the Fix-and-Optimize heuristic which is used in each generation of the GAs acts like a diversification tool in the search of GAs by leading the search to the regions where good quality solutions exist and as a result finds good quality solutions in less number of generations. Hence, we might state that when hybridised with a MIP-based heuristic in both ways (i.e. sequential and embedded) the solution quality of the pure GAs improves significantly.

Considering the average of ten runs for the problem instances in Classes 7, 8, and 9, the solution qualities of both of the proposed hybrid approaches are better than the time decomposition heuristic of Suerie and Stadtler (2003). Another conclusion which can be drawn from these experimental results is that the performances of the hybrid approaches improve as the problem complexity increases. In other words, the average gap for the proposed hybrid approaches tends to decrease as the number of products increases. For instance, the average gap observed for the proposed approaches in classes 3 and 4 is less than the one observed in classes 1 and 2. Likewise, in classes 7, 8 and 9 a decreasing tendency in average gap is observed. However, regarding the computational time, the proposed hybrid approaches require much more computational effort than the time decomposition heuristic.

Comparing two proposed hybrid approaches with respect to the average and best results of ten runs, it is observed that for five problem classes (i.e. Classes 2, 3, 5, 6, 9) the solution quality of the second hybrid approach is better than that of the first hybrid approach. Thus, we could state that using the Fix-and-Optimize heuristic in each generation of GAs, i.e., embedded hybrid approach, improves the solution quality much more than using it after a predetermined number of generations, i.e., sequential hybrid approach.

A comparison of the performance of the pure GAs and proposed hybrid approaches with the performance of the TS heuristic of Gopalakrishnan et al. (2001) is given in Table 5.12. The gaps in Table 5.12 are taken from Gopalakrishnan et al. (2001). As seen in the table, for all problem classes, the solution quality of both the pure GAs and proposed hybrid approaches is better than that of the TS heuristic. However, since TS deals with one solution throughout the search in contrast to the multiple solutions of the GAs it is computationally more efficient.

The general conclusion which can be drawn from this comparative experimental study is that apart from requiring a long computational time to find good quality solutions, hybridising the pure GAs with the MIP-based heuristic significantly improves its performance.

5.5 Chapter Summary

In this chapter, we proposed two novel hybrid approaches for solving the CLSPC. These proposed hybrid approaches combine a meta-heuristic and a MIP based heuristic. In the first hybrid approach, the hybridization scheme is a sequential scheme where Fix-and-Optimize heuristic approach is executed after GAs. In the second hybrid approach, the Fix-and-Optimize heuristic is embedded into the loop of GAs and it is used in every generation to improve the solution quality of the GAs.

To evaluate the performances of pure GAs and proposed hybrid approaches, we carried out comparative experiments using benchmark problems reported in the literature. The results show that the performance of pure GAs improves notably when hybridized with the Fix-and-Optimize heuristic.

In this chapter, the capacitated lot sizing problem with setup carryover is considered. Since our ultimate goal is to propose a solution approach for the capacitated lot sizing problem with setup carryover and backordering, in the next chapter, the problem is extended to include the backorder issue and the proposed hybrid approaches are modified to deal with this extended problem.

CHAPTER SIX
GENETIC ALGORITHM BASED APPROACHES FOR SOLVING THE CAPACITATED LOT SIZING PROBLEM WITH SETUP CARRYOVER AND BACKORDERING

6.1 Introduction

As mentioned in chapter five, the capacitated lot sizing problem with setup carryover and backordering is solved in two stages. The hybrid approaches proposed to solve the capacitated lot sizing problem with setup carryover (CLSPC) are presented in chapter five. In this chapter, the model presented in previous chapter is extended both to meet unsatisfied demand in future periods by backordering and also to include backorder costs which are incurred for every unit and period of the delay. Throughout this chapter, the CLSP with setup carryover and backordering is abbreviated to $CLSP^+$.

This chapter presents how the hybrid approaches proposed in chapter five are adapted to solve the $CLSP^+$. Besides adapting these hybrid approaches to the intrinsic features of the $CLSP^+$, different decomposition schemes are suggested in implementation of Fix-and-Optimize heuristic. The performance of Fix-and-Optimize heuristic is evaluated experimentally on a number of problem instances to gain insight into the effects of these decomposition schemes. To further improve the performance of the proposed hybrid approaches, efficient GA parameters are determined through some pilot experiments. Lastly, the performance of the proposed hybrid approaches under different values of backorder costs is investigated.

The rest of this chapter is organized as follows. In section 6.2, the capacitated lot sizing problem with setup carryover and backordering is defined with the necessary notations. In section 6.3, the proposed GA-based approaches are discussed in detail.

The results of comparative experiments are presented in section 6.4. In Section 6.5, the robustness of the proposed hybrid approaches with respect to backorder costs is examined. Finally, in section 6.6 the context of this chapter is summarized.

6.2 Problem Statement: The Capacitated Lot Sizing Problem with Setup Carryover and Backordering

The model to solve CLSP$^+$ determines the level and timing of production along with the semi-sequencing of the products in a period. If the demand can not be satisfied in a period, it is backordered meaning that it is satisfied in the following periods. Hence, the main difference between the problem in this chapter and the problem in the previous chapter is the consideration of the backordering issue. In solving the CLSP$^+$, the model proposed by Suerie and Stadtler (2003) is modified to include backordering costs and constraints.

Sets and Indices:
j: items $j \in K = \{1, 2, 3, ..., K\}$
t: periods $t \in P = \{1, 2, 3, ...P\}$

Parameters:

sc_j setup cost for item j
h_{jt} unit holding cost for item j in period t
C_t amount of resource available in period t
a_j time to process one unit of item j
st_j setup time of item j
M a large number
d_{jt} independent demand for end item j in period t
b_j backordering cost of item j

Decision Variables:

I^+_{jt} Inventory level for item j at the end of period t

I^-_{jt} Backorder level for item j at the end of period t

X_{jt} Production amount of item j in period t

Y_{jt} Binary setup variable for item j in period t (=1, if a setup for item j is performed in period t, =0, otherwise),

W_{jt} Binary linkage variable which indicates whether a setup state for item j is carried from period (t-1) to (t) (=1) or not (=0).

Q_t Single item variable which indicates that resource is occupied solely by item *i* on period *t* (=1) or not (=0).

The MIP Model (I&L representation)

$$Min \sum_{j=1}^{K} \sum_{t=1}^{P} (sc_j Y_{jt} + h_{jt} I_{jt}^{+} + b_j I_{jt}^{-}\ ) \qquad (1)$$

s.t.

$$I_{j,t-1}^{+} + X_{jt} - I_{jt}^{+} - I_{j,t-1}^{-} + I_{jt}^{-} = d_{jt} \quad \forall j \in K; \forall t \in P \qquad (2)$$

$$\sum_{j \in K} (a_j X_{jt} + s_j Y_{jt}) \leq C_t \ \ \forall t \in P \qquad (3)$$

$$\sum_{j \in K} W_{jt} \leq 1 \quad \forall\, t \in \{2,....,P\} \qquad (4)$$

$$W_{jt} \leq Y_{jt-1} + W_{jt-1} \quad \forall j \in K, \forall t \in \{2,....,P\} \qquad (5)$$

$$W_{jt+1} + W_{jt} \leq 1 + Q_t \quad \forall\, t \in \{1,....,P-1\} \qquad (6)$$

$$Y_{jt} + Q_t \leq 1 \quad \forall\, t \in \{1,....,P-1\} \qquad (7)$$

$$X_{jt} \leq M(Y_{jt} + W_{jt})\ \forall j \in K, \forall t \in \{1,....,P\} \qquad (8)$$

$$Q_t \geq 0 \quad \forall\, t \in \{1,....,P-1\} \qquad (9)$$

$$W_{jt} \in \{0,1\} \quad (W_{j1} = 0)\,,\, Y_{jt} \in \{0,1\} \quad \forall j \in K, \forall\, t \in P \qquad (10)$$

$$X_{jt}, \quad I_{jt}^{+} \geq 0, \quad I_{jt}^{-} \geq 0 \geq 0 \quad \forall j \in K; \forall t \in P \qquad (11)$$

$$I_{jt}^{-} = 0, \quad \forall j \in K; t = P \qquad (12)$$

The objective function (1) aims at minimizing inventory holding and setup costs. Constraints (2) are the inventory balance equations. The capacity constraints are placed in Constraints (3). Constraints (4) ensure that at most one setup state can be preserved from one period to the next on each resource. Constraints (5) guarantee that a setup can be carried over to period *t* only if either item *j* is setup in period *t*-1 or the setup state is already carried over from period *t*-2 to *t*-1. A setup state can only be preserved over two bucket boundaries, if $Q_t = 1$ in constraints (6), which is only possible if there is no setup in this period (7). Constraints (8) are the upper bounds on the production quantities. Finally, variables are restricted to be nonnegative or binary, respectively, (9) to (11). It is assumed that there are no setup carryovers in the first period as stated in constraints (10). All the demand should be satisfied during the planning horizon as stated in constraints (12).

6.3 Proposed GA Based Approaches

In this chapter, different GA based hybrid methodologies are proposed to solve the $CLSP^+$. The search in GA follows almost the same logic explained in the previous chapter. The only difference is that the LP model to solve $CLSP^+$ includes backordering constraints. Besides the time decomposition scheme, in this chapter the product decomposition scheme is integrated into the Fix-and-Optimize heuristic, and its performance is evaluated experimentally to gain an insight.

6.3.1 Elements of the Proposed GA Based Hybrid Approaches

The elements of the proposed GA based hybrid approaches are explained in the following.

6.3.1.1 Chromosome Representation

To represent the chromosomes, the matrix representation given in chapter five is used. While the first row of this matrix shows the setup variables (Y_{jt}), the second row shows the setup carryover variables (W_{jt}). The values of these binary variables for each product in each period are stated in the columns of the matrix. Thus, the length of the chromosome is the number of periods (P) multiplied by the number of products (K).

6.3.1.2 Initial Population

It was observed in chapter five that generating some portion of the initial population randomly and the remaining part in a smart way definitely improved the performance of GAs. Hence, we used this new initialization scheme for creating the initial population in solving the $CLSP^+$ and set the ratio of random part to small part to 0.5 throughout the computational experiments of this chapter. The procedure to generate the initial population is presented in Figure 6.1.

1. Creating the setup variables (Y_{jt})	2. Creating the setup carryover variables (W_{jt})
POP: The population size Solve the LP relaxation of the problem using the Simple Plant Location formulation to get a good lower bound. FOR *i*=1 to *POP*/2 (SMART PART) FOR *g*=0 to (*K***P*) Step 1. Set the cut off point to (1/ (*POP*/2)). If the value of the setup variable in the LP relaxation is under this cut off point, set *Y* (the value of the gene *g*) in chromosome *i* to 0 otherwise set it to 1. Step 2. Increase the cut off point by (1/ (*POP*/2)) and go to Step 1. FOR *i*=*POP*/2 to *POP* (RANDOM PART) FOR *g*=0 to (*K***P*) Step 1. Pick a random number between 0 and 1. Step 2. If the random number is less than or equal to 0.5, set *Y* (the value of the gene *g*) in chromosome *i* to 0, otherwise set it to the value of 1.	FOR *i*=1 to *POP* FOR *t*=2 to *P* FOR *j*=1 to *K* (Start with the product that has the highest setup cost) IF Y_{jt-1}=1 and W_{jt-1}=0 (there is a setup and no setup carryover for this product in the previous period), THEN Set W_{jt}=1 (assign a setup carryover for that product in the current period) IF Y_{jt}=1 THEN Set Y_{jt}=0 ENDIF ENDIF BREAK (go to the next period)

Figure 6.1 Initial population generation

6.3.1.3 Genetic and Repair Operators

The modified roulette wheel selection operator that was explained in the previous chapter is used to select the chromosomes for recombination. The offsprings are created using one point crossover. Single bit flip mutation operator is used as the mutation operator. To repair the infeasibilities that occur after crossover and mutation operators, the same repair operators presented in chapter five are used. Elitism is used as a survival scheme. The genetic search is terminated when the total number of generations exceeds a maximum number.

6.3.2 The Fix-and-Optimize Heuristic with Product Decomposition

As given in earlier chapter, the hybrid approaches proposed to solve the CLSPC combine the GA and the Fix-and-Optimize heuristic. The Fix-and-Optimize heuristic decomposes the problem into smaller problems by considering the periods in the planning horizon. In each iteration of the algorithm, one problem is optimized while the others are fixed to the best values obtained so far. Starting from the first period, the proposed time decomposition algorithm optimizes the binary setup and setup carryover variables in each time window (see Figure 5.1).

In this chapter, besides the time decomposition scheme, we propose to decompose the whole problem into manageable problems by considering some product specific information. Particularly, based on some product specific information such as setup costs and holding costs, first a priority is assigned to each product, and problems are ordered according to these priorities (i.e., the higher the cost the higher the priority). Starting with high priority product the relevant binary setup and setup carryover variables are optimized while the rest is set to the fixed values based on the best solution obtained so far. Figure 6.2 illustrates the implementation of the product decomposition scheme for a three product problem with five periods.

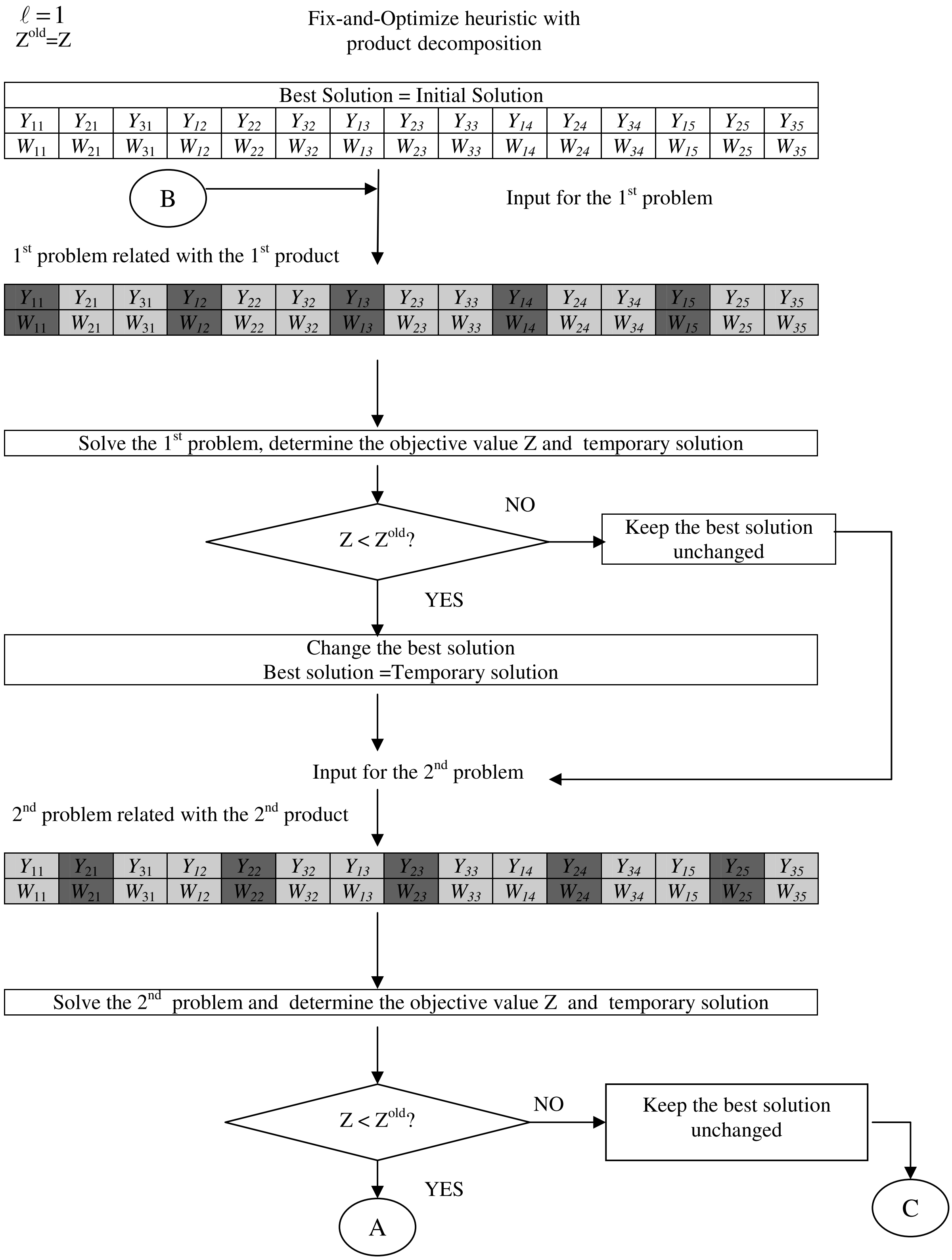


Figure 6.2 The outline of the Fix-and-Optimize heuristic with product decomposition

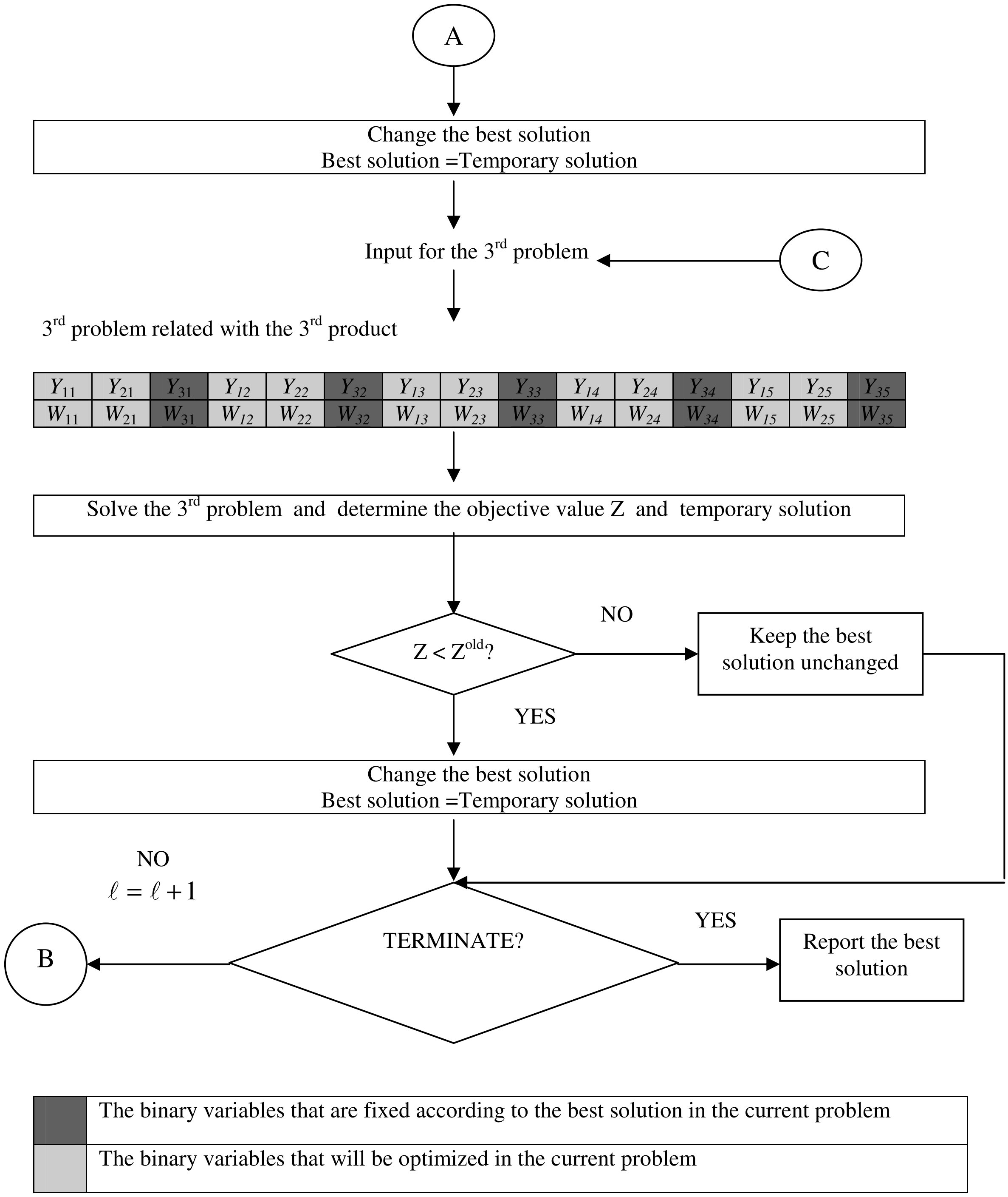


Figure 6.2 The outline of the Fix-and-Optimize heuristic with product decomposition (cont.)

In this study, five different criteria are used to give a priority to the products: the setup cost, the holding cost, the ratio of setup cost to holding cost, the total demand of a product and the total cost of a product obtained from the Economic Order Quantity (EOQ). The results of a computational study investigating the effects of these five criteria on performance of the proposed product decomposition scheme are given in section 6.4.

6.3.3. Modified Hybrid Approaches

This chapter presents how the two hybrid approaches presented in chapter five are modified to solve the $CLSP^+$. As before, these hybrid approaches include two different hybridization schemes namely, sequential and embedded. While in the sequential hybridization scheme the Fix-and-Optimize heuristic is performed after GA, in the second one, the Fix-and-Optimize heuristic is embedded into the loop of GAs to refine the solutions obtained by GAs. Unlike the two hybridization schemes presented in chapter five where the Fix-and-Optimize heuristic is applied with only time decomposition scheme, the Fix-and-Optimize heuristic in all of the hybridization schemes presented in this chapter is applied with both the time decomposition, and also product decomposition.

6.3.3.1. Sequential Hybrid Approaches

Sequential hybrid approaches are based on modification of the first hybrid approach proposed for solving the CLSPC. They are called “sequential” because first the GA is run for a predetermined number of generations and then the Fix-and-Optimize heuristic is applied with both time and also product decomposition schemes.

The main idea of the sequential hybrid approaches is presented in Figure 6.3. As seen in Figure 6.3, first a GA is run until a termination criterion is met (Step 2) and the best solution (Step 1) of these generations is identified as the initial solution for the Fix-and-Optimize heuristic (Step 3). Then a Fix-and-Optimize heuristic is employed to further improve the solution quality (Step 4). This approach can be classified as a kind of a sequential hybridization scheme in which the Fix-and-Optimize heuristic is executed after GAs.

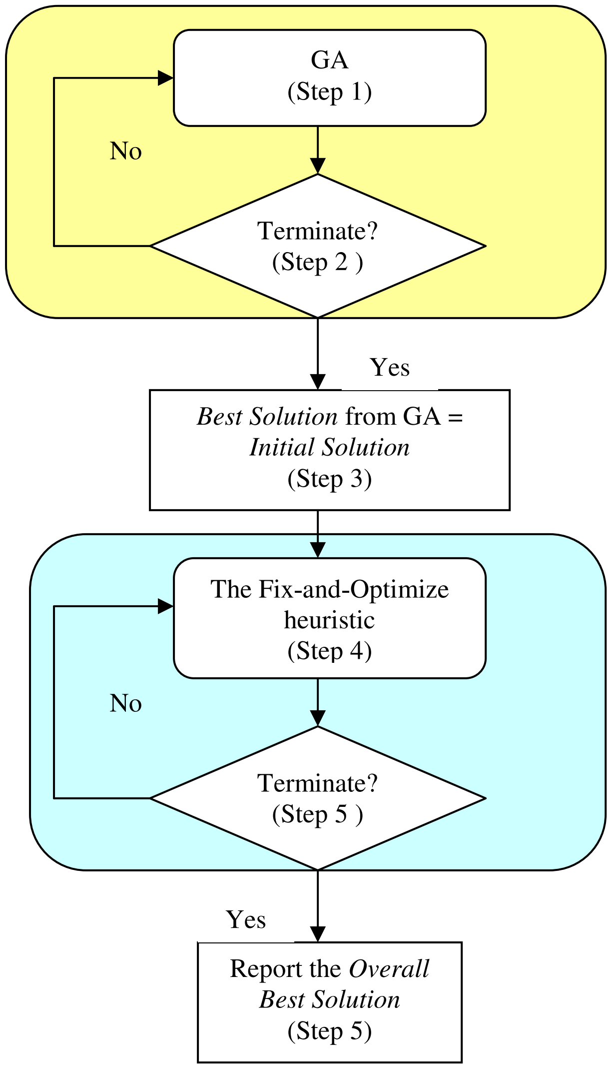


Figure 6.3 The control logic of the sequential hybrid approaches

Applying the decomposition schemes in Fix-and-Optimize heuristic in different ways, we developed the following sequential GA based hybrid approaches:

Hybrid Approach 1 (H1):	GAs first and then the Fix-and-Optimize heuristic with *time decomposition* only.
Hybrid Approach 2 (H2):	GAs first and then the Fix-and-Optimize heuristic with *product decomposition* only.
Hybrid Approach 3 (H3):	GAs first and then the Fix-and-Optimize heuristic with *time decomposition* first, then *product decomposition.*
Hybrid Approach 4 (H4):	GAs first and then the Fix-and-Optimize heuristic with *product decomposition* first, then *time decomposition*.

It should also be noted that the procedure to implement H3 and H4 is different from that of H1 and H2 (see Figure 6.4). Unlike H1 and H2 in which one decomposition scheme is used at each iteration to improve the best solution generated so far, while implementing H3 and H4 the two decomposition schemes are used sequentially at each iteration (i.e. either in the sequence of time and product decomposition or product and time decomposition).

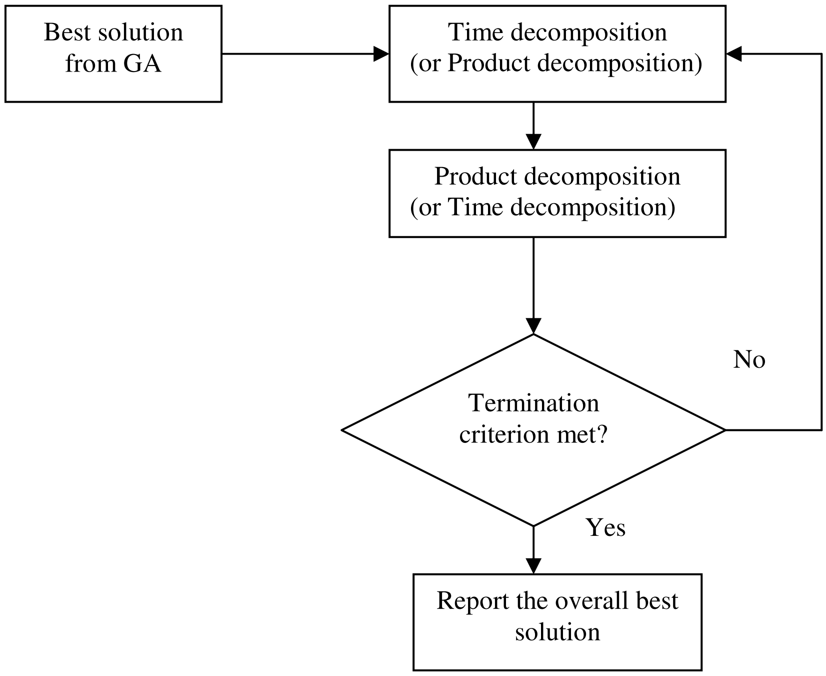


Figure 6.4 The control logic for H3 and H4.

6.3.3.2. Embedded Hybrid Approaches

Embedded hybrid approaches are based on modification of the second hybrid approach proposed for solving the CLSPC in chapter five. They are called “embedded” because the Fix-and-Optimize heuristic is embedded into the loop of GAs in these approaches. During the search of GAs, after a new population is formed, a random solution is chosen from the new population and it is set as the initial solution in the Fix-and-Optimize heuristic. Then, the Fix-and-Optimize heuristic improves this solution until a termination criterion is met. The improved solution is then placed back into the new population and the genetic search continues (see Figure 6.5). Thus, in these approaches the Fix-and-Optimize heuristic is called in every generation which is quiet different from the sequential hybrid approaches.

Like sequential hybrid approaches, two decomposition schemes are employed in four different ways to form the following embedded hybrid approaches:

Hybrid Approach 5 (H5):	The Fix-and-Optimize heuristic with *time decomposition* in the loop of GAs.
Hybrid Approach 6 (H6):	The Fix-and-Optimize heuristic with *product decomposition* in the loop of GAs.
Hybrid Approach 7 (H7):	The Fix-and-Optimize heuristic with *time decomposition* in one generation and *product decomposition* in another generation in the loop of GAs.
Hybrid Approach 8 (H8):	The Fix-and-Optimize heuristic with two decomposition schemes in the sequence of *product* and *time decomposition* in the loop of GAs.

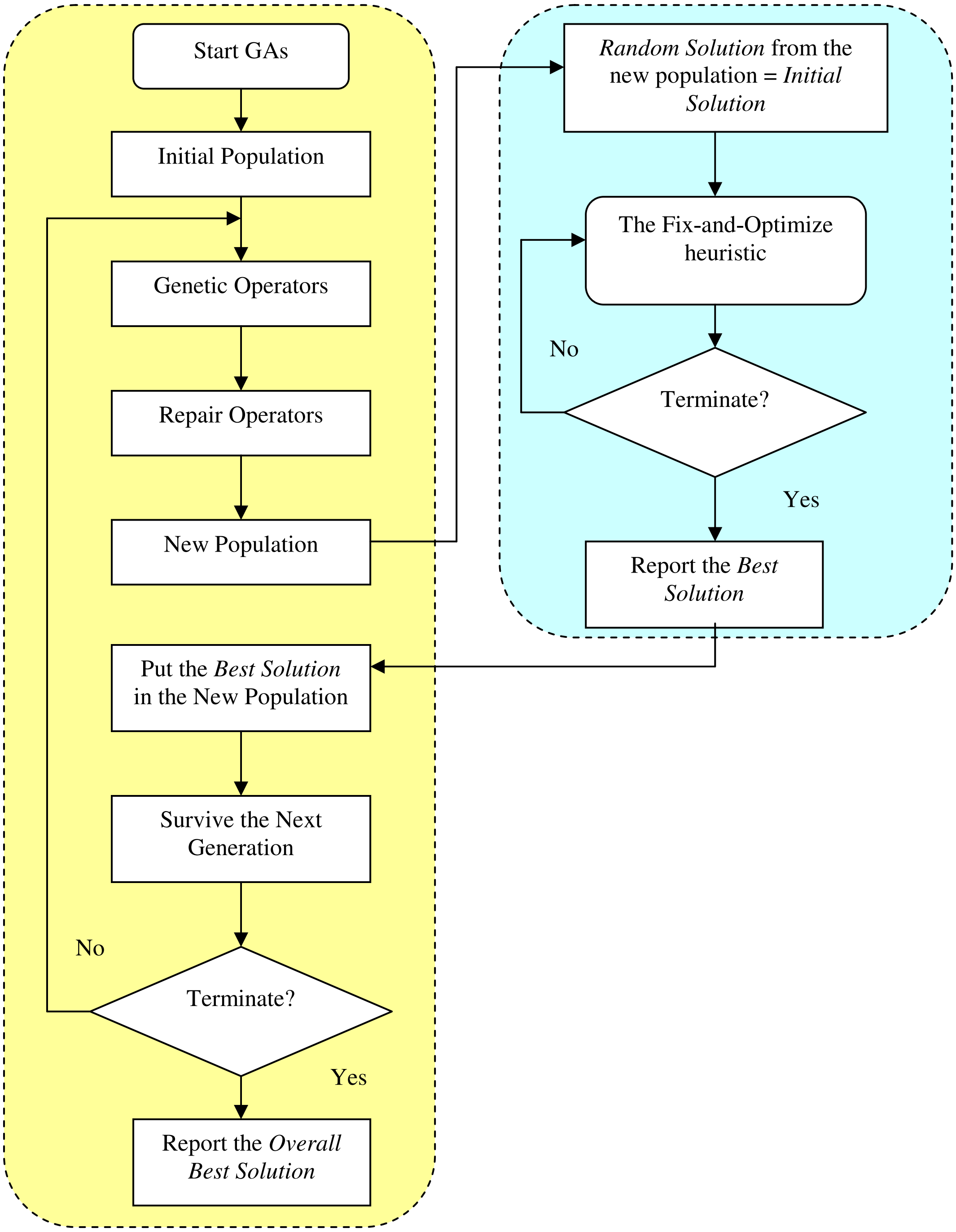


Figure 6.5 The control logic of embedded hybrid approaches

6.4 Computational Results

To evaluate the performance of the proposed hybrid approaches for solving the $CLSP^+$, we carried out experiments on a number of problem instances.

As a result of some preliminary tests the computational time to solve each problem using Fix-and-Optimize heuristic was limited to 2 seconds. All computations were carried out on a PC with Dual Core, 2 GHz microprocessor and 2 GB RAM. The pure and hybrid GAs were coded in Visual C++ 2008 Express Edition and all problems were solved using Concert Technology of Cplex 11.2.

6.4.1 Benchmark Problems

We found no published test problems with backorder costs to evaluate the performance of the proposed hybrid approaches. So to form the test problems, we modified the problem instances given in Trigeiro et al. (1989) by adding the backorder costs. Specifically, to introduce backordering, the demands were modified and multiplied by 1.1 and backorder cost was defined as a linear function of the holding cost ($b=fh$), where $f=2$ as in Millar and Yang (1994). Table 6.1 presents the features of the data instances studied.

Table 6.1 Classification of data instances

	Problem Class					
	1	2	3	4	5	6
Number of products	6	12	24	6	12	24
Number of periods	15	15	15	30	30	30
Number of instances	5	5	5	5	5	5

The results were compared to the lower bounds obtained from the Simple Plant Location (SPL) formulation of the MIP model since there was no study in the literature to which the results could be compared. The solution quality of the proposed approaches was measured by computing the gap as follows.

$$Gap = 100 * \frac{(heuristic\ solution - lower\ bound)}{lower\ bound} \tag{13}$$

6.4.2 Identifying Efficient GA Parameters

Similar to the sub-section 5.4.3 in the previous chapter, this sub-section also focuses on a critical dilemma faced in many GA applications; determination of efficient GA parameters to ensure high performance. Prior to evaluating the performance of proposed hybrid approaches, a preliminary analysis was performed to identify optimal GA control parameters. The aim is to further improve the performance of proposed hybrid approaches in solving $CLSP^+$.

The parameters of a GA namely the population size (*P*), number of generations (*G*), crossover rate (*%C*) and mutation rate (*%M*) significantly affect the convergence speed of the GAs. To identify efficient GA parameters, the experiments were conducted on a medium complexity $CLSP^+$ (i.e. 10 products and 20 periods). To carry out the experiments a full factorial design given in Table 6.2 has been employed and 10 runs were carried out at each design point leading to 270 runs in total.

As shown in Table 6.2, three different combinations of population size and the number of generations (*P*/*G*), 20/500, 40/ 250 and 100/100 were used by fixing the total number of chromosomes at 10000.

In order to determine statistically significant control parameters, a statistical analysis of variance (ANOVA) was conducted and the results are presented in Table 6.3.

Table 6.2 Experimental factors and levels

Factors	Levels	
P/*G*	1	20/500
	2	50/200
	3	100/100
%C	1	0.5
	2	0.75
	3	0.95
%M	1	0.001
	2	0.005
	3	0.025

Table 6.3 ANOVA results

Source of variation	DF	F_{calc}	Prob[F>F_{calc}]
Within + residual	26	39.276	0.000
P/*G*	2	5.633	0.004*
%C	2	0.454	0.636
%M	2	489.442	0.000*
P/*G* & *%C*	4	0.704	0.590
P/*G* & *%M*	4	2.598	0.037*
%C & *% M*	4	2.973	0.020*
P/*G* & *% C* & *% M*	8	0.628	0.754

As stated in Table 6.3, the combination of population size and number of generations (*P*/*G*), the mutation rate (*%M*), the interaction between the mutation rate and the combination of population size and number of generations (*P*/*G**%*M*), the interaction between the crossover rate and mutation rate (*%C**%*M*) are all significant factors. The scatter plot of total cost for 270 runs is graphically shown in Figure 6.5.

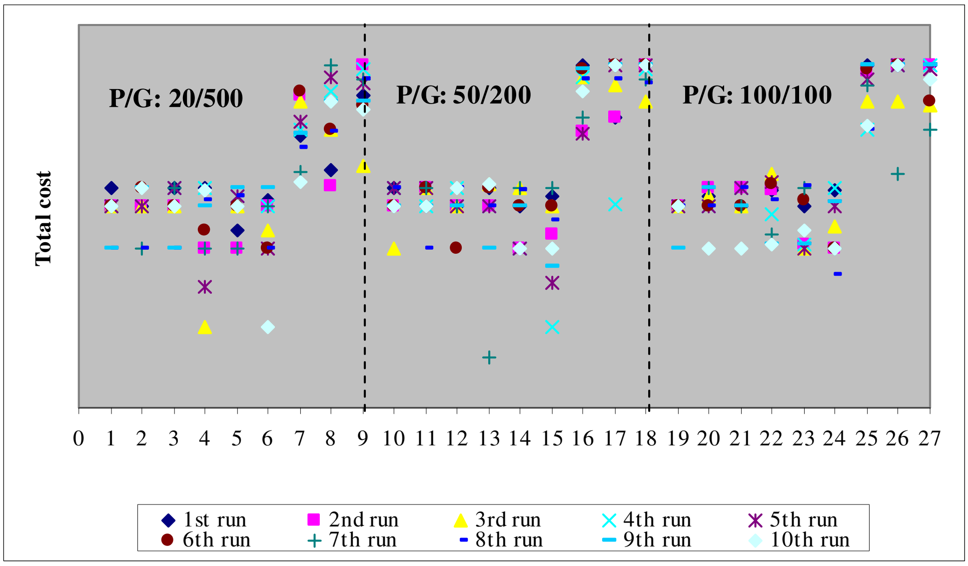


Figure 6.6 Scatter plot of total cost from ten runs

From Figure 6.6, it can be clearly seen that as mutation rate increases, the solution quality of GA deteriorates, i.e., the total cost increases (Experiments 7, 8, 9, 16, 17, 18, 25, 26, and 27). Therefore, the results of experiments do not suggest using a high mutation rate. Moreover, a population size of 100 with 100 generations results in smallest spread among the experiments with low and medium mutation rates (Experiment 24). Therefore, the parameter values used in experiment 24 were recommended as efficient parameters. Based on these experiments, population size of 100, crossover rate of 0.95 and mutation rate of 0.005 were used for 100 generations in the computational experiments given in the next sub-section.

6.4.3 Analysis and Discussion of the Results

This section presents the results of computational studies investigating the performance of proposed hybrid approaches for solving the $CLSP^+$. It must be noted that in all comparative experiments, the same GA parameter settings presented above in section 6.4.2 were used.

6.4.3.1 Results of Pure GAs

First, the performance of pure GAs (PGAs) for each problem class is summarized. The average gaps over lower bounds obtained from the SPL formulation are given in in terms of both the average of 10 runs and the best result among these runs (see Table 6.4). The "computational time" shows the average time required by the PGAs to solve the problems instances in a given class and "feasibility" shows the fraction of problem instances that could be solved by the PGAs. As seen in Table 6.4, all problem instances could be solved using the PGAs; however the large gaps from the lower bounds indicate that the overall performance of PGAs is rather poor for solving $CLSP^+$.

Table 6.4 The results of PGAs

PGA	Avg. % gap over Lower Bound		Computational Time (s)	Feasibility %
	Avg. of 10 runs	Best among 10 runs		
Class 1	30.16 %	26.4 %	252.37	100.00
Class 2	34.98 %	33.8 %	198.6	100.00
Class 3	17.25 %	15.4 %	226.7	100.00
Class 4	16.80 %	15.8 %	287.56	100.00
Class 5	8.64 %	7.9 %	302.75	100.00
Class 6	8.82 %	8.5 %	357.05	100.00

6.4.3.2 Results of Sequential GA-based Hybrid Approaches

This sub-section presents the results of computational studies investigating the performance of the proposed sequential GA-based hybrid approaches for solving the $CLSP^+$. Prior to carrying out a comparative experimental study, first a set of experiments was conducted to select a criterion for the implementation of the proposed product decomposition scheme. As stated above, to obtain compatible results the CPU-time limit for solving each class of problem was set to the computational times given in Table 6.4. For instance, the CPU-time limit to solve the problem instances in Class 1 using proposed hybrid approaches was set to 252.37 s.

(see Table 6.4) and half of this duration was devoted to the GAs and the other half was devoted to the Fix-and-Optimize heuristic.

6.4.3.2.1 Experimental Results for Product Decomposition Scheme. As mentioned earlier, the proposed product decomposition scheme uses five types of product specific criteria in forming the problems used in the Fix-and-Optimize heuristic. To decide whether the problems should be formed in ascending, descending or random ordering of these five criteria a pilot experiment was carried out and noted that the performance of the proposed product decomposition scheme was not substantially affected by the order of these criteria. However, since descending order of the criteria resulted in slightly better performance than the other two, in the following computational experiments, descending order of these criteria was used. To evaluate the performance of H2 (GA+product decomposition), 10 independent runs were carried out using descending order of these five criteria. Experimental results given in Table 6.5 are based on the average of these runs.

Table 6.5 Average % gap over lower bound (based on 10 runs)

	According to "Setup cost"	According to "Holding cost"	According to "(setup cost/holding cost)"	According to "Total demand"	According to the "Total cost"
Class 1	27.7%	28.07%	27.6%	28.14 %	**27.4 %**
Class 2	**34.3%**	34.5%	34.5%	34.5 %	34.5 %
Class 3	**15.1%**	15.4%	15.3%	15.3 %	15.2 %
Class 4	**14.2%**	14.8%	14.6%	14.6 %	14.6 %
Class 5	6.4%	6.5%	**6.2%**	6.3 %	6.3 %
Class 6	**6.8%**	7.04%	7%	6.9 %	7.03 %

As seen in Table 6.5, for the majority of the problem classes, the product decomposition based on setup cost resulted in a better performance than the others. Hence, we decided to employ this decomposition criterion in the following computational comparative studies.

6.4.3.2.2 Comparative Experimental Results. In this section, the experimental studies evaluating the performance of the proposed sequential GA based hybrid approaches are given. To obtain compatible results the performance of the proposed sequential hybrid approaches and PGA were tested under the same run time limitation. For instance, the time limitation to solve problem instances in Class 1 was set to 252.37 s. (see Table 6.4).

The overall results obtained across all problem classes are shown in Table 6.6. It is quite clear from Table 6.6 that the proposed hybridized approaches outperform the PGA which has the highest gap for all problem classes. As for the performance of the proposed hybrid approaches, while H4 (i.e., GAs first and then the Fix-and-Optimize heuristic in the sequence of *product decomposition* and *time decomposition)* slightly outperforms H1 and H3, H2 (i.e., GAs first and then the Fix-and Optimize heuristic with product decomposition only) has the worst performance among four approaches. Overall we could state that H4 and H3 employing Fix-and-Optimize heuristic with two decomposition schemes are better than H1 and H2 having one decomposition scheme.

Table 6.6 Overall results of sequential proposed hybrid approaches

	Based on average of 10 runs							
Decomp. scheme	**Time**		**Product**		**Time+Product**		**Product+Time**	
Abbrev.	**H1**		**H2**		**H3**		**H4**	
	Avg. Gap	Avg. # gen.	Avg. Gap	Avg. # gen.	Avg. Gap	Avg. # gen.	Avg. Gap	Avg. # gen.
Class 1	21.10 %	54.04	27.70 %	44.66	**21.05 %**	56.16	21.07 %	58.66
Class 2	26.05 %	37.86	34.27 %	34.8	26.10 %	40.2	**25.95 %**	38.72
Class 3	10.10 %	41.52	15.14 %	35.78	9.912 %	42.56	**8.69 %**	43.04
Class 4	10.57 %	43.54	14.24 %	43.98	10.36 %	45.04	**10.07 %**	43.34
Class 5	5.57 %	36.14	6.40 %	37.74	4.58 %	50.88	**4.57 %**	44.52
Class 6	6.86 %	33.54	6.81 %	36.22	5.67 %	40.06	**5.60 %**	38.16

Moreover, it has been observed that H3 and H4 substantially outperform H2 and also outperform H1 on more difficult instances. Next, the behaviour of the H3 and H4 heuristics is analyzed in more detail. The summary of this analysis is given in Tables 6.7 and 6.8 which show the improvements (i.e. the decrease in the gaps) obtained throughout the iterations of the Fix and Optimize heuristic in H3 and H4, respectively. In these tables, while the second column shows the average number of iterations in the Fix-and-Optimize heuristic, the third column gives the gap of the best solution from GAs which is set as an initial solution in the Fix-and-Optimize heuristic. The other columns show the average of the gaps in ten runs obtained after the first, fifth, tenth, fifteenth, twentieth and thirtieth iterations.

As mentioned earlier, the proposed hybrid approaches start from an initial solution which is generated with GA and iteratively improve it. Results of experiments summarized in Tables 6.7 and 6.8 show that for all problem classes both in H3 and H4, a great portion of the improvement in solution quality occurs in the first five iterations and further iterations yield only a negligible improvement in solution quality. Another thing which can be observed from these two tables is that as expected the Fix-and-Optimize heuristic converges more quickly for small-size problems (i.e. Class 1).

Based on the results of these experimental studies, we could state that using a decomposition scheme is definitely improving the performance of the proposed hybrid approaches. It has been also observed during the experiments that the time decomposition scheme outperforms the product decomposition scheme. As seen in Table 6.6, for all problem classes, H1, H3 and H4 employing the time decomposition scheme outperform H2 employing only the product decomposition scheme. Moreover, it has been observed that using two decomposition schemes within the same hybrid approach in a sequential manner further improves the performance of these approaches. It should be noted that in H3 and H4 the two decomposition schemes were used sequentially at each iteration (i.e. either in the sequence of time and product decomposition or product and time decomposition) (see Figure 6.4).

In order to clarify the additive effect of implementing two decomposition schemes sequentially within the same hybrid approach we presented a more detailed analysis of the results in Table 6.9. For instance, the second column of Table 6.9 shows the percentage of the improvement achieved by the time decomposition in H3 after the product decomposition (i.e. the initial solution for the time decomposition is the best solution obtained after the product decomposition). This additive effect is measured using the following equation:

$$\textit{improvement with time decomposition} = \frac{\textit{solution after product decomposition} - \textit{solution after time decomposition}}{\textit{lower bound}} * 100 \quad (14)$$

Improvements given in the remaining columns of Table 6.9 are also measured in a similar way.

Table 6.7 Experimental Results for H3 (based on 10 runs)

H3	Avg. # of iterations	Best solution gap from GA	1st iteration	5th iteration	10th Iteration	15th iteration	20th iteration	30th iteration
Class 1	30.74	30.78 %	21.9696 %	21.0478 %	21.0478 %	21.0478 %	21.0478 %	21.0478 %
Class 2	13.28	36.12 %	27.0339 %	26.1083 %	26.1028 %	*	*	*
Class 3	14.96	19.67 %	10.7734 %	9.9694 %	9.927 %	9.912 %	*	*
Class 4	10.82	17.83 %	11.7469 %	10.4634 %	10.3599 %	*	*	*
Class 5	15.38	9.67 %	5.1291 %	4.6248 %	4.5925 %	4.5778 %	*	*
Class 6	9.88	8.89 %	6.242 %	5.6974 %	5.684 %	5.67 %	*	*

Table 6.8 Experimental Results for H4 (based on 10 runs)

H4	Avg. # of iterations	Best solution gap from GA	1st iteration	5th iteration	10th iteration	15th iteration	20th iteration	30th iteration
Class 1	30.68	30.63 %	22.1932 %	21.0718 %	21.0718 %	21.0705 %	21.0705 %	21.0705 %
Class 2	13.14	35.75 %	26.9399 %	25.9501 %	25.9497 %	*	*	*
Class 3	14.62	19.18 %	9.66 %	8.7155 %	8.6912 %	8.6912 %	*	*
Class 4	10.52	17.36 %	11.4317 %	10.1612 %	10.0724 %	*	*	*
Class 5	14.6	9.50 %	5.4429 %	4.6439 %	4.5719 %	4.568 %	*	*
Class 6	9.28	8.88 %	6.6067 %	5.6986 %	5.5979 %	*	*	*

Table 6.9 illustrates the improvements obtained for each problem class.

Table 6.9 Comparative analysis of the improvements obtained (based on all iterations in 10 runs).

Problem Class	H3		H4	
	Improvement with time decomposition %	Improvement with product decomposition %	Improvement with product decomposition %	Improvement with time decomposition %
Class 1	0.034	**0.2156**	0.011	0.011
Class 2	**0.0445**	0.0149	0.0141	**0.4695**
Class 3	0.0162	**0.0233**	0.0245	**0.3163**
Class 4	0.0521	**0.0665**	0.0546	**0.3357**
Class 5	**0.0845**	0.0541	0.0430	**0.0903**
Class 6	**0.3917**	0.1491	**0.087**	0.0591

For H3, it can be easily seen that both decomposition schemes improve the solution quality; however, it is not possible to draw a general conclusion about which decomposition type contributes the most. For medium-size problems (see problem classes 3 and 4), while the improvement by product decomposition is more than the improvement by the time decomposition, for large-sized problems (see problem classes 5 and 6) the time decomposition improves the solution quality more than the product decomposition. However, for H4, the results of our analysis are quite clear. In comparison to the product decomposition, the contribution of time decomposition to the solution quality is remarkable. As given in Table 6.9, using the time decomposition after the product decomposition further improves the solution quality in four problem classes out of six. These results further clarify why H4 outperforms H3 in most of the problem classes (see Table 6.6).

The general conclusion which can be drawn from this comparative experimental study is that the proposed hybrid approaches which combine the PGA with a MIP-based heuristic significantly outperforms the PGA in solving the $CLSP^+$. Moreover, it has been observed that the performance of these hybrid approaches is affected by both the type of the decomposition scheme and also in which sequence to use these schemes. In summary, the proposed H4 approach using both decompositions schemes in the sequence of time and product decomposition has the best performance among the four proposed sequential hybrid approaches.

6.4.3.3 Comparative Experimental Results for Embedded GA-based Approaches

In this section, the experimental studies evaluating the performance of the proposed embedded GA-based hybrid approaches are given. Similar to the sequential hybrid approaches, the PGA and embedded proposed hybrid approaches were tested under the same run time limitation in order to obtain compatible results. Having observed outperformance of implementing decomposition schemes in the sequence of time and product decomposition schemes we evaluated the performance of embedded hybrid approaches also in the same way, i.e. first time decomposition, and then product decomposition. The results of experiments are summarized in Table 6.10. The table presents the average of gaps obtained based on 10 runs and the number of generations in the genetic search.

Similar to the results obtained for sequential hybrid approaches (i.e. H2), H6 employing only product decomposition in the loop of GAs has the worst performance and H8 which uses both decomposition schemes in the loop of GAs slightly outperforms H5 and H7. Overall, it can be stated that compared to using a single decomposition scheme in the loop of GAs, using both decomposition schemes sequentially has more potential to improve the solution quality. Moreover, it has been observed that the performance of proposed embedded hybrid approaches is affected by the type of decomposition scheme and the embedded hybrid approaches employing time decomposition scheme have better performance (H5, H7 and H8).

Table 6.10 Results of experiments (based on average of 10 runs)

	Proposed Embedded Hybrid Approaches							
Abbrev.	**H5**		**H6**		**H7**		**H8**	
Decomp. scheme	**Time**		**Product**		**Time in one generation & Product in other generation**		**Product + Time**	
	Avg. Gap	Avg. # gen.	Avg. Gap	Avg. # gen.	Avg. Gap	Avg. # gen.	Avg. Gap	Avg. # gen.
Class 1	19.93 %	39.38	22.57 %	53.96	20.05 %	43.38	**19.80 %**	34.22
Class 2	25.80 %	17.4	29.09 %	37.86	25.81 %	24.3	**25.67 %**	15.32
Class 3	9.65 %	24.96	11.53 %	29.38	9.77 %	29.2	**9.46 %**	18.6
Class 4	11.18 %	17.44	12.29 %	39.76	11.55 %	21.64	**11.11 %**	12.56
Class 5	5.06 %	29.8	5.26 %	31.66	4.80 %	29.04	**4.58 %**	18.62
Class 6	6.91 %	19.44	6.30 %	18.14	6.51 %	18.04	**6.19 %**	10.68

6.4.4 Summary of the findings

In this chapter, the performances of the proposed hybrid approaches are compared to that of PGA in solving the CLSP$^+$. The percentages of the average gaps obtained by each approach for each problem class are summarized in Table 6.11. As seen in the table, PGA has the worst performance among all approaches and all proposed hybrid approaches work very well to improve the solution quality across all problem classes. This can be attributed to the success of employing the Fix-and-Optimize heuristic in proposed hybrid approaches. As mentioned earlier, the GAs can locate the promising region for global optimum, but in large search spaces they often have tendency to converge to local optimum. In this study, to improve the solution quality of GAs two types of hybrid approaches, i.e., sequential and embedded have been proposed. These hybrid approaches integrate GAs with a MIP based heuristic, the Fix-and-Optimize heuristic in different ways. In the sequential hybrid approaches, the final solution of GA is used as the initial solution for the Fix-and-Optimize heuristic. In doing so, it is hoped that GA will find a promising region in the search space and following, the Fix-and-Optimize heuristic will further explore this promising region to find the best solution. On the other hand, in the embedded hybrid approaches, the Fix-and-Optimize heuristic, which is embedded into the loop of GA, acts like a diversification strategy and improves the solution quality in each generation by guiding the GAs to the promising areas in the search space.

Overall, it can be stated that the proposed sequential hybrid approaches have better performance than proposed embedded hybrid approaches. Namely, sequential hybrid approaches outperformed embedded hybrid approaches in four problem classes out of six (see Table 6.11).

Table 6.11 Summary of experimental studies (based on average of 10 runs)

	PGA	Proposed Hybrid Approaches							
		Sequential Approaches				Embedded Approaches			
		H1	H2	H3	H4	H5	H6	H7	H8
Class 1	30.16 %	21.10 %	27.70 %	21.05 %	21.07 %	19.93 %	22.57 %	20.05 %	**19.80 %**
Class 2	34.98 %	26.05 %	34.27 %	26.10 %	25.95 %	25.80 %	29.09 %	25.81 %	**25.67 %**
Class 3	17.25 %	10.10 %	15.14 %	9.912 %	**8.69 %**	9.65 %	11.53 %	9.77 %	9.46 %
Class 4	16.80 %	10.57 %	14.24 %	10.36 %	**10.07 %**	11.18 %	12.29 %	11.55 %	11.11 %
Class 5	8.64 %	5.57 %	6.40 %	4.58 %	**4.57 %**	5.06 %	5.26 %	4.80 %	4.58 %
Class 6	8.82 %	6.86 %	6.81 %	5.67 %	**5.60 %**	6.91 %	6.30 %	6.51 %	6.19 %
Avg.	19.44 %	13.38 %	17.43 %	12.95 %	**12.66 %**	13.09 %	14.51 %	13.08 %	12.80 %

In order to check whether these performance differences, i.e., average gaps, are statistically significant, we employed one-way ANOVA test. For each problem class, five instances were generated and each instance was run 10 times to compare eight hybrid approaches. The results are stated in Tables 6.12 - 6.17.

Table 6.12 ANOVA table for Class 1

Source	Sum of squares	Degrees of freedom	Mean square	*F*-Ratio	*P*-Value
Between groups	0.269	7	3.850E-02	5.126	0.000
Within groups	2.944	392	7.510E-03		
Total (corrected)	3.213	399			

Table 6.13 ANOVA table for Class 2

Source	Sum of squares	Degrees of freedom	Mean square	*F*-Ratio	*P*-Value
Between groups	0.366	7	5.226E-02	16.254	0.000
Within groups	1.260	392	3.215E-03		
Total (corrected)	1.626	399			

Table 6.14 ANOVA table Class 3

Source	Sum of squares	Degrees of freedom	Mean square	*F*-Ratio	*P*-Value
Between groups	0.140	7	1.994E-02	29.877	0.000
Within groups	0.262	392	6.673E-04		
Total (corrected)	0.401	399			

Table 6.15 ANOVA table for Class 4

Source	Sum of squares	Degrees of freedom	Mean square	*F*-Ratio	*P*-Value
Between groups	7.907E-02	7	1.130E-02	12.022	0.000
Within groups	0.368	392	9.396E-04		
Total (corrected)	0.447	399			

Table 6.16 ANOVA table for Class 5

Source	Sum of squares	Degrees of freedom	Mean square	*F*-Ratio	*P*-Value
Between groups	1.784E-02	7	2.549E-03	31.857	0.000
Within groups	3.136E-02	392	8.000E-05		
Total (corrected)	4.920E-02	399			

Table 6.17 ANOVA table for Class 6

Source	Sum of squares	Degrees of freedom	Mean square	*F*-Ratio	*P*-Value
Between groups	8.904E-03	7	1.272E-03	13.761	0.000
Within groups	3.623E-02	392	9.243E-05		
Total (corrected)	4.514E-02	399			

As seen in Tables 6.12 – 6.17, the performance differences for each problem class are statistically significant. Moreover, to carry out all pairwise comparisons of the proposed hybrid approaches we employed Tukey's test that is a single step multi comparison procedure to be used in conjunction with ANOVA. The results for Class 1 are presented in Tables 6.18 and the other results are given in Appendix A.

Table 6.18 Multiple comparisons for Class 1

Proposed Hybrid Approach (I)	Proposed Hybrid Approaches (J)	Mean Difference (I-J)	Sig. (*p*)
H1	H2	-6.5600E-02*	0.004
	H3	2.000E-04	1.000
	H4	-5.5511E-17	1.000
	H5	1.140E-02	0.998
	H6	-3.4800E-02	0.476
	H7	1.000E-02	0.999
	H8	1.340E-02	0.994
H2	H1	6.560E-02*	0.004
	H3	6.580E-02*	0.004
	H4	6.560E-02*	0.004
	H5	7.700E-02*	0.000
	H6	3.080E-02	0.636
	H7	7.560E-02*	0.000
	H8	7.900E-02*	0.000
H3	H1	-2.0000E-04	1.000
	H2	-6.5800E-02*	0.004
	H4	-2.0000E-04	1.000
	H5	1.120E-02	0.998
	H6	-3.5000E-02	0.469
	H7	9.800E-03	0.999
	H8	1.320E-02	0.995
H4	H1	5.551E-17	1.000
	H2	-6.5600E-02*	0.004
	H3	2.000E-04	1.000
	H5	1.140E-02	0.998
	H6	-3.4800E-02	0.476
	H7	1.000E-02	0.999
	H8	1.340E-02	0.994
H5	H1	-1.1400E-02	0.998
	H2	-7.7000E-02*	0.000
	H3	-1.1200E-02	0.998
	H4	-1.1400E-02	0.998
	H6	-4.6200E-02	0.133
	H7	-1.4000E-03	1.000
	H8	2.000E-03	1.000

Table 6.18 Multiple comparisons for Class 1 (cont.)

H6	H1	3.480E-02	0.476
	H2	-3.0800E-02	0.636
	H3	3.500E-02	0.469
	H4	3.480E-02	0.476
	H5	4.620E-02	0.133
	H7	4.480E-02	0.161
	H8	4.820E-02	0.100
H7	H1	-1.0000E-02	0.999
	H2	-7.5600E-02*	0.000
	H3	-9.8000E-03	0.999
	H4	-1.0000E-02	0.999
	H5	1.400E-03	1.000
	H6	-4.4800E-02	0.161
	H8	3.400E-03	1.000
H8	H1	-1.3400E-02	0.994
	H2	-7.9000E-02*	0.000
	H3	-1.3200E-02	0.995
	H4	-1.3400E-02	0.994
	H5	-2.0000E-03	1.000
	H6	-4.8200E-02	0.100
	H7	-3.4000E-03	1.000

* shows that the mean difference is significant at the 0.05 level

As seen in Table 6.18 and in Appendices A1 - A4, for the problem instances in almost all classes, i.e., Classes 1 to 5, the difference between H2 and other hybrid approaches with respect to average gap is found to be statistically significant in great majority of pair-wise comparisons. Likewise, based on the observations reported in Table 6.18 and in Appendices A1 – A4, the same statistical inference can be made for the performance of H6. As given in earlier section, H2 and H6 using only the product decomposition scheme were identified as poorly performing hybrid approaches. So, the Tukey's test further verified these results by providing statistical support for the low performance of H2 and H6 (i.e., having large average gap against other hybrid approaches).

Table 6.19 summarizes the total number of statistically significant differences among proposed hybrid approaches for each problem class.

Table 6.19 Summary of results

Proposed Hybrid Approach	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6
H1	1	2	2	2	5	2
H2	6	6	7	6	7	3
H3	1	2	2	2	3	6
H4	1	2	2	2	3	5
H5	1	2	2	2	2	3
H6	0	6	7	5	6	2
H7	1	2	2	1	3	2
H8	1	2	2	2	3	3

As noted before in earlier sections, the proposed hybrid approaches employing only the time decomposition scheme (H1 and H5) or combination of time and product decomposition schemes in different forms (H3, H4, H7, H8) outperformed the hybrid approaches employing only the product decomposition scheme (H2 and H6) and this was statistically supported by the results from Tukey's test.

Having identified H4 and H8 as best performing hybrid approaches, in the next section, we investigated how the performances of these two approaches and also PGA are affected by the changes in values of various problem-specific parameters including backorder costs, setup times, setup costs, capacity utilization and demand variability.

6.5 Investigation of the Robustness of the Proposed Hybrid Approaches

In this sub-section, the performances of the PGA and the two hybrid approaches proposed, i.e. H4 and H8 are further analyzed.

6.5.1 The Experimental Design

As stated in Section 6.4.1, we found no published benchmark problems for evaluating the performance of the proposed hybrid approaches. To generate test problems, we added the backorder costs to the problems with 10 products and 20 periods in Trigeiro et al. (1989). Table 6.20 summarizes the experimental factors used in generating these problem instances: capacity utilization (low and high), time between orders (TBO) (low, medium or high), coefficient of variation (CV) in demand (low and high), and setup time (low or medium). The TBO is changed by changing the setup cost. It should be noted that the holding cost is assumed to be constant for all products.

Table 6.20 The experimental design

Parameter	Values used	Total values
Backorder cost	2*holding cost (Low)	2
	6*holding cost (High)	
Capacity utilization	75% (Low)	2
	85% (High)	
Time between Orders (TBO)	1 period (Low)	3
	2 periods (Medium)	
	4 periods (High)	
Setup time	11 units of capacity (Low)	2
	43 units of capacity (High)	
CV in demand	0.35 (Low)	2
	0.59 (High)	
	Total parameter combinations	48
	Number of problems/combinations	5
	Total problems	240

For each combination 5 random problems were generated. Thus, a total of 240 problems were generated to test the performances of the proposed approaches. For the purpose of the experiments, backorder cost was defined as a linear function of the holding cost ($b=fh$), where $f>=1$, as in Millar and Yang (1994). In this study, the actual value of the backorder cost is not the primary issue, but rather, the quality of the performances of the proposed approaches when backordering is allowed.

6.5.2 The Statistical Analysis of the Proposed Approaches

In summarizing the results, the means and standard deviations of the gaps for five problem instances in each of the 48 categories were computed. As stated in Millar and Yang (1994), the standard deviations provide a measure of the stability of the algorithms which is very important in the design of heuristics. It should be noted that the best solutions (i.e. minimum solution) found during 10 runs is used to calculate the gaps.

Table 6.21 shows a summary of statistics for each of the proposed solution approach. The performances of H4 and H8 appear to be comparable, though H8 slightly outperforms H4. PGA has the worst mean and standard deviation of the gaps as stated in Table 6.21.

Table 6.21 Summary statistics for the proposed solution approaches

	PGA	H4	H8
Sample size	240	240	240
Minimum	0	0	0
Maximum	0.60378	0.392949	0.382354
Mean value	0.160215	0.124868	0.123501
Standard Deviation	0.134757	0.118936	0.117927

In order to identify the statistically significant factors that affect the performance of the proposed approaches, we carried out analysis of variance (ANOVA). Table 6.22 summarizes the results of analysis for all the main effects, two-way, and all higher level interactions using 720 observations (240 problem instances tested for 3 solution approaches).

Table 6.22 Analysis of variance

Source of variation	Degrees of freedom	F	Sig. (*p*)
Approach	2	178.636	**0.000**
CV	1	23.908	**0.000**
TBO	2	8716.091	**0.000**
Setuptime	1	3.388	0.066
Capacity utilization	1	166.681	**0.000**
Backorder cost	1	119.809	**0.000**
Approach * CV	2	17.777	**0.000**
Approach * TBO	4	16.728	**0.000**
Approach * Setuptime	2	0.383	0.682

Table 6.22 Analysis of variance (cont.)

Approach * Capacity utilization	2	3.087	**0.046**
Approach * Backorder cost	2	3.159	**0.043**
CV * TBO	2	23.657	**0.000**
CV * Setuptime	1	8.450	**0.004**
CV * Capacity utilization	1	42.269	**0.000**
CV * Backorder cost	1	1.393	0.238
TBO * Setuptime	2	10.398	**0.000**
TBO* Capacity utilization	2	1.090	0.337
TBO* Backorder cost	2	141.876	**0.000**
Setuptime * Capacity utilization	1	23.132	**0.000**
Setuptime * Backorder cost	1	0.034	0.853
Capacity utilization * Backorder cost	1	4.561	**0.033**
Approach * CV * TBO	4	1.257	0.286
Approach * CV * Setuptime	2	1.265	0.283
Approach * CV * Capacity utilization	2	0.692	0.501
Approach * CV * Backorder cost	2	0.026	0.974
Approach * TBO* Setuptime	4	2.121	0.077
Approach * TBO* Capacity utilization	4	0.068	0.992
Approach * TBO* Backorder cost	4	1.681	0.153
Approach * Setuptime * Capacity utilization	2	1.108	0.331
Approach * Setuptime * Backorder cost	2	0.011	0.989
Approach * Capacity utilization * Backorder cost	2	0.383	0.682
CV * TBO* Setuptime	2	5.262	**0.005**
CV * TBO* Capacity utilization	2	12.801	**0.000**
CV * TBO* Backorder cost	2	0.315	0.730
CV * Setuptime * Capacity utilization	1	6.661	**0.010**
CV * Setuptime * Backorder cost	1	0.031	0.861
CV * Capacity utilization * Backorder cost	1	0.595	0.441
TBO* Setuptime * Capacity utilization	2	0.134	0.874
TBO* Setuptime * Backorder cost	2	0.126	0.882
TBO* Capacity utilization * Backorder cost	2	0.140	0.869

Table 6.22 Analysis of variance (cont.)

Setuptime * Capacity utilization * Backorder cost	1	0.124	0.725
Approach * CV * TBO* Setuptime	4	0.809	0.520
Approach * CV * TBO* Capacity utilization	4	0.445	0.776
Approach * CV * TBO* Backorder cost	4	0.302	0.877
Approach * CV * Setuptime * Capacity utilization	2	1.720	0.180
Approach * CV * Setuptime * Backorder cost	2	0.096	0.909
Approach * CV * Capacity utilization * Backorder cost	2	0.084	0.919
Approach * TBO* Setuptime * Capacity utilization	4	2.482	**0.043**
Approach * TBO* Setuptime * Backorder cost	4	0.027	0.999
Approach * TBO* Capacity utilization * Backorder cost	4	0.365	0.834
Approach * Setuptime * Capacity utilization * Backorder cost	2	0.135	0.874
CV * TBO* Setuptime * Capacity utilization	2	0.086	0.918
CV * TBO* Setuptime * Backorder cost	2	0.185	0.831
CV * TBO* Capacity utilization * Backorder cost	2	0.434	0.648
CV * Setuptime * Capacity utilization * Backorder cost	1	0.453	0.501
TBO* Setuptime * Capacity utilization * Backorder cost	2	0.138	0.871
Approach * CV * TBO* Setuptime * Capacity utilization	4	1.704	0.148
Approach * CV * TBO* Setuptime * Backorder cost	4	0.028	0.998
Approach * CV * TBO* Capacity utilization * Backorder cost	4	0.077	0.989
Approach * CV * Setuptime * Capacity utilization * Backorder cost	2	0.192	0.825
Approach * TBO* Setuptime * Capacity utilization * Backorder cost	4	0.105	0.981

Table 6.22 Analysis of variance (cont.)

CV * TBO* Setuptime * Capacity utilization * Backorder cost	2	0.323	0.724
Approach * CV * TBO* Setuptime * Capacity utilization * Backorder cost	4	0.111	0.979
Error	576		

It can be seen in the table that the main effects such as solution approaches, CV in demand, TBO, capacity utilization and backorder cost are statistically significant. Moreover, a couple of two-way interactions, a few three-way interactions and one four-way interactions are found to be statistically significant.

Since the ultimate goal in this experimental study is to investigate the sensitivity of the three approaches to the changes in backorder costs, the results obtained were further analyzed under two category: 1. Backorder cost is two times of holding cost, (*b=2h,* see Tables 6.23 – 6.29)*,* 2. Backorder cost is six times of holding cost (*b=6h*, see Tables 6.30 – 6.36). The effects of interactions between different factors are examined under two different levels of backorder cost.

In Table 6.23, for all problem instances, the average gap between the proposed approaches and the lower bound under different levels of experimental factors are shown.

Table 6.23 The effects of different experimental factors on average gap (*b*=2*h*)

	TBO			Capacity utilization		CV in demand		Setup time	
	Low	Medium	High	Low	High	Low	High	Low	High
PGA	2.9 %	11.9 %	37.2 %	16.2 %	18.5 %	17.1 %	17.5 %	17.4 %	17.3 %
H4	1.5 %	7.4 %	31.0 %	12.4 %	14.2 %	14.2 %	12.6 %	13.5 %	13.1 %
H8	1.4 %	7.3 %	30.9 %	12.3 %	14.0 %	14.1 %	12.2 %	13.4 %	13.0 %

- From Table 6.23, it can be seen that the most significant effect on the gap is due to the TBO which is related to the setup cost. As setup cost increases (i.e. TBO increases), the gap increases significantly for all proposed solution approaches.

- As far as the capacity utilization, there are minor differences in the average gaps. Millar and Yang (1994) state that for a special case of the capacitated lot sizing problem with backordering the performance of the algorithms gets worse as the capacity becomes tighter. The results in Table 6.23 also support this idea.
- Regarding the CV in demand, the gaps in the proposed hybrid approaches decrease as the CV in demand increases. However, the performance of the PGA deteriorates under high level of CV in demand.
- As the setup time increases, the performances of the solution approaches improve (i.e. the gaps decrease).

The insight gained as a result of analyzing the effects of interaction between different experimental factors when backorder cost is two times of holding cost is given in the following.

- In comparison to the performance of PGA and H4 under low setup times, their performance under high setup times and different levels of TBO (i.e. low and medium) is better. The performance of H8 is observed to be good under high setup times and all levels of TBO.
- As the simultaneous effect of TBO and CV is considered (Table 6.25), H4 and H8 perform better for high CV and medium and high TBO cases. When low CV is considered, the performances of H4 and H8 are better in low TBO cases. On the other hand, the performance of PGA is better in low CV cases than the performance in high CV cases. For low setup costs and low CV, all proposed solution approaches perform better than medium and high TBO cases.
- The simultaneous effects of TBO and capacity utilization (see Table 6.26) show that all solution approaches perform better for low capacity cases.
- Across all levels of setup times, H4 and H8 perform better for high CV cases (see Table 6.27). However, levels of CV and setup time have a different effect on the performance of the PGA. The average gap is the lowest for high CV and low setup time cases but the highest for high CV and high setup time cases.

- Like the case in Table 6.27, all solution approaches perform better for low capacity cases across all levels of setup time when the effect of interaction of setup time and capacity utilization is considered (see Table 6.28).
- The simultaneous effect of CV and capacity utilization on the gap is shown in Table 6.29. Across all levels of capacity, H4 and H8 perform better for high CV cases. However, the performance of PGA changes according to the level of CV. In low CV and capacity cases, the performance of PGA is better than the performance in low CV and high capacity cases.

Table 6.24 The effects of interaction of TBO and setup time on gap (b=2h)

	PGA			H4			H8		
	TBO			TBO			TBO		
Setup time	Low	Medium	High	Low	Medium	High	Low	Medium	High
Low	3.83 %	11.96 %	36.37 %	1.95%	7.51%	30.93%	1.90%	7.43%	30.88%
High	1.97%	11.75 %	38.07%	0.96%	7.26%	31.21%	0.93%	7.09%	30.85%

Table 6.25 The effect of interaction of TBO and CV on gap (b=2h)

	PGA			H4			H8		
	TBO			TBO			TBO		
CV	Low	Medium	High	Low	Medium	High	Low	Medium	High
Low	1.99 %	11.84 %	37.56 %	1.36 %	8.40 %	32.80 %	1.33 %	8.35 %	32.71 %
High	3.81 %	11.87 %	36.88 %	1.56 %	6.37 %	29.35 %	1.49 %	6.17 %	29.03 %

Table 6.26 The effect of interaction of TBO and capacity utilization on gap (b=2h)

	PGA			H4			H8		
	TBO			TBO			TBO		
Capacity utilization	Low	Medium	High	Low	Medium	High	Low	Medium	High
Low	1.38 %	10.80 %	36.27 %	0.43 %	6.71 %	30.17 %	0.39 %	6.61 %	29.95 %
High	4.42 %	12.90 %	38.17 %	2.48 %	8.05 %	31.97 %	2.44 %	7.91 %	31.79 %

Table 6.27 The effect of interaction of setup time and CV on gap (b=2h)

	PGA		H4		H8	
	Setup time		Setup time		Setup time	
CV	Low	High	Low	High	Low	High
Low	17.62 %	16.65 %	14.64 %	13.73 %	14.47 %	13.80 %
High	17.16 %	17.87 %	12.29 %	12.56 %	12.34 %	12.12 %

Table 6.28 The effect of interaction of setup time and capacity utilization on gap (b=2h)

	PGA		H4		H8	
	Setup time		Setup time		Setup time	
Capacity utilization	Low	High	Low	High	Low	High
Low	15.94 %	16.36 %	12.14 %	12.73 %	12.07 %	12.57 %
High	18.84 %	18.16 %	14.78 %	13.56 %	14.74 %	13.35 %

Table 6.29 The effect of interaction of CV and capacity utilization on gap (b=2h)

	PGA		H4		H8	
	Capacity utilization		Capacity utilization		Capacity utilization	
CV	Low	High	Low	High	Low	High
Low	15.42 %	18.84 %	12.59 %	15.78 %	12.57 %	15.70 %
High	16.88 %	18.15 %	12.28 %	12.57 %	12.07 %	12.40 %

In Table 6.30, for all problem instances, the average gaps between the proposed approaches and the lower bound under different levels of experimental factors are shown.

Table 6.30 The effects of different experimental factors on average gap (b=6h)

	TBO			Capacity utilization		CV in demand		Setup time	
	Low	Medium	High	Low	High	Low	High	Low	High
PGA	3.07 %	11.66 %	29.42 %	12.94 %	16.50 %	14.28 %	15.16 %	14.78 %	14.65 %
H4	1.66 %	7.70 %	25.65 %	10.38 %	12.66 %	12.52 %	11.34 %	11.86 %	11.48 %
H8	1.61 %	7.60 %	25.34 %	10.53 %	12.81 %	12.31 %	10.72 %	11.81 %	11.23 %

The results stated in Table 6.30 are very similar to those presented in Table 6.23. The insight gained can be summarized as follows.

- The gap increases for all proposed solution approaches as setup cost increases (i.e. TBO increases).
- Minor differences in the average gaps are observed when the capacity utilization is changed from low to high.
- As for the CV in demand is concerned, the gaps in the proposed hybrid approaches decrease as the CV in demand increases. However, the performance of the PGA deteriorates under high level of CV in demand. So we can conclude that unlike PGA, proposed hybrid approaches perform better when the variability of demand increases.
- The performances of the solution approaches improve (i.e. the gaps decrease) as the setup time increases.

The effects of interaction between different experimental factors when backorder cost is six times of holding cost are given in the following.

Table 6.31 The effects of interaction of TBO and setup time on gap (b=6h)

	PGA			H4			H8		
	TBO			TBO			TBO		
Setup time	Low	Medium	High	Low	Medium	High	Low	Medium	High
Low	4.03 %	11.94 %	28.39 %	2.17 %	7.99 %	25.42 %	2.099 %	7.89 %	25.43 %
High	2.12 %	11.39 %	30.46 %	1.14 %	7.41 %	25.87 %	1.12 %	7.31 %	25.25 %

Table 6.32 The effect of interaction of TBO and CV on gap (b=6h)

	PGA			H4			H8		
	TBO			TBO			TBO		
CV	Low	Medium	High	Low	Medium	High	Low	Medium	High
Low	2.12 %	11.74 %	28.98 %	1.49 %	8.37 %	27.08 %	1.50 %	8.37 %	27.07 %
High	4.02 %	11.58 %	29.86 %	1.82 %	7.03 %	24.22 %	1.72 %	6.83 %	23.61 %

Table 6.33 The effect of interaction of TBO and capacity utilization on gap (b=6h)

	PGA			H4			H8		
	TBO			TBO			TBO		
Capacity utilization	Low	Medium	High	Low	Medium	High	Low	Medium	High
Low	1.20 %	10.27 %	27.34 %	0.41 %	6.62 %	24.57 %	0.39 %	6.47 %	24.27 %
High	4.95 %	13.05 %	31.50 %	2.90 %	8.79 %	26.73 %	2.83 %	8.73 %	26.41 %

Table 6.34 The effect of interaction of setup time and CV on gap (b=6h)

	PGA		H4		H8	
	Setup time		Setup time		Setup time	
CV	Low	High	Low	High	Low	High
Low	14.82 %	13.74 %	12.67 %	11.96 %	12.70 %	11.93 %
High	14.74 %	15.57 %	11.06 %	11.00 %	10.92 %	10.53 %

Table 6.35 The effect of interaction of setup time and capacity utilization on gap (b=6h)

	PGA		H4		H8	
	Setup time		Setup time		Setup time	
Capacity utilization	Low	High	Low	High	Low	High
Low	12.79 %	13.08 %	10.14 %	10.93 %	10.07 %	10.68 %
High	16.77 %	16.22 %	13.59 %	12.02 %	13.54 %	11.78 %

Table 6.36 The effect of interaction of CV and capacity utilization on gap (b=6h)

	PGA		H4		H8	
	Capacity utilization		Capacity utilization		Capacity utilization	
CV	Low	High	Low	High	Low	High
Low	12.16 %	16.40 %	10.64 %	13.99 %	10.51 %	14.12 %
High	13.71 %	16.60 %	10.43 %	11.62 %	10.25 %	11.20 %

The insight gained as a result of analyzing interactions between different factors can be summarized as follows:

- For high setup times and different levels of TBO (i.e. low and medium), PGA and H4 perform better than low setup times. However, the performance of H8 is good for high setup times and all levels of TBO.

- As the simultaneous effect of TBO and CV is considered (Table 6.32), H4 and H8 perform better for high CV and medium and high TBO cases. When low CV is taken into account, the performance of H4 and H8 deteriorates as TBO increases. For low TBOs and low CV, all proposed solution approaches perform better than medium and high TBO cases.
- When simultaneous effects of TBO and capacity utilization (see Table 6.33) are concerned all solution approaches perform better for low capacity cases.
- Across all levels of setup times, H4 and H8 perform better for high CV cases (see Table 6.34). However, levels of CV and setup time have a different effect on the performance of the PGA. The average gap is the lowest for high CV and low setup time cases but the highest for high CV and high setup time cases.
- When the effect of interaction of setup time and capacity utilization is considered, all solution approaches perform better for low capacity cases across all levels of setup time (see Table 6.35).
- Across all levels of capacity, H4 and H8 perform better for high CV cases. However, the performance of PGA is better in low CV cases than the performance in the high CV cases (see Table 6.36).

Having analyzed the effects of changing backorder cost on performances of PGA, H4 and H8 under various experimental factors, we can state that all approaches show a similar performance. In other words, their performance is quite robust against the changes in backorder cost. Regardless of the level of backorder cost, proposed hybrid approaches, H4 and H8 perform better when the demand variability is high and the PGA performs better when the demand variability is low. Another insight gained is that the performances of all approaches are better under low setup costs and low capacity utilization and lastly, all approaches have better performances under high setup times in comparison to low setup times.

6.6 Chapter Summary

Lot sizing is one of the most well-known optimization problem in production planning. Most of the previous studies in this field focus on solving the CLSP which is known to be NP-Hard. Including the setup carryover and backordering to the model makes it more complex so exact algorithms are not capable of producing good quality solutions in reasonable computational time. In this chapter, a number of hybrid approaches are presented to solve the $CLSP^+$. These hybrid approaches are based on the integration of GAs with a MIP based heuristic, Fix-and-Optimize heuristic. To our knowledge, this is the first extensive study proposing GAs to deal with the $CLSP^+$.

An extensive computational study is made to compare the performance of these hybrid approaches to the pure GAs. Since there is no study in the literature to which the results can be compared, the results are compared to the lower bounds obtained from the Simple Plant Location (SPL) formulation of the MIP model. As a result of this comparative study, we could state that the performance of GAs can be improved noticeably by hybridizing it with Fix-and-Optimize heuristic in different ways. Moreover, the type of decomposition scheme applied in the Fix-and-Optimize heuristic has been observed to have an important effect on the solution quality of the proposed hybrid approaches.

In order to investigate how the performances of proposed hybrid approaches are affected by changes in problem specific parameter values including backorder costs, setup times, setup costs, capacity utilization and demand variability an extensive experimental analysis is carried out. The results of these experiments are summarized with respect to two levels of backorder cost and it is observed that the performances of the proposed hybrid approaches are not sensitive to changes in backorder cost.

CHAPTER SEVEN
CONCLUSION

7.1 Summary of the Thesis

Lot sizing studies aim at determining the timing and quantity of production in order to satisfy the customer demand while minimizing the total cost. Lot sizing problem is one of the most challenging production planning problems and solving this problem optimally has an important impact on the efficiency of production and inventory systems.

The Capacitated Lot Sizing Problem (CLSP) has received a lot of attention of the researchers. Due to their importance in production planning and inventory management, lot sizing problems have been studied for many years with different extensions such as setup times, setup carryover, backordering, parallel machines etc. Since most lot sizing problems are hard to solve, the main focus of the research in this area has been on developing heuristic approaches to deal with the complexity of the problems in reasonable computational time. Among these heuristic approaches, evolutionary computation has received increasing attention in recent years. The most well known evolutionary computation method is GAs that have been employed to solve different optimization problems across various disciplines due to their flexibility and simplicity.

Based on the extensive literature review of GA applications for solving lot sizing problems given in chapter four, it was noted that most of the research in this area has addressed the solution of the CLSP with setup times in single level. Although various applications of GAs have been proposed for solving lot sizing problem with different modeling features, we have not noted any application of GAs in solving the capacitated lot sizing problem with setup times, setup carryover and backordering in single level.

Regarding methodological contributions in this area, it was noted that generally, pure GAs were employed in most of the studies to solve lot sizing problems and not much research was noted in the area of hybridizing GAs with other solution approaches. However, since pure GAs are not good at finding the exact minimum in a large search space, the solutions found by pure GAs can still be improved by integrating GAs with other metaheuristics or exact methods.

Regarding these perceived research gaps in relevant literature, this Ph.D study aims at proposing efficient GA-based hybrid approaches for solving the $CLSP^+$. Solution approaches to this complex problem have been developed in two stages. In the first stage we focused on the CLSPC, which is less complicated than the $CLSP^+$ and proposed two hybrid approaches combining GAs with Fix-and-Optimize heuristic. While in the first hybrid approach, two approaches were hybridized sequentially where the Fix-and-Optimize heuristic was performed after GAs, in the second hybrid approach, the Fix-and-Optimize heuristic was embedded into the loop of GAs. Before forming a new population at each generation of GAs, a random solution is chosen and it is set as an initial solution for the Fix-and-Optimize heuristic. Thus, the Fix-and-Optimize heuristic is used to refine the solution quality in each generation of GA and help GA direct the search toward promising regions in the search space. In these two hybrid approaches, to divide the problem into smaller ones the Fix-and-Optimize heuristic is applied with the time decomposition scheme.

Moreover, to further improve the performance of the proposed hybrid approaches, a novel initialization scheme was proposed. The new initialization scheme utilizes problem specific information and randomness to create the initial population. Problem specific information is obtained by solving the LP relaxation of the CLSP. An important experimental design issue in the implementation of this new initialization scheme is the determination of ratio of smart part to random part in the initial population. This ratio was determined by testing the proposed initialization scheme on a set of benchmark problems. Additionally, to maintain the feasibility during the search of GAs, a number of repair operators were proposed. To evaluate the performance of the proposed approaches, a number of comparative experiments were carried out and it was noted that the proposed hybrid approaches outperform the recent results reported in the published literature.

In the second stage of this Ph.D. study, eight GA-based hybrid approaches were proposed to solve the $CLSP^+$. While the first four of these approaches are developed by modifying sequential hybrid approaches, the remaining four are the modifications of embedded hybrid approaches. The Fix-and-Optimize heuristic in these eight hybrid approaches was applied with the product decomposition scheme besides the time decomposition scheme. The performances of proposed hybrid approaches were tested on various instances and it is noted that combining GAs with the Fix-and-Optimize heuristic in different ways improves the solution quality notably for solving the $CLSP^+$. Moreover, a statistical analysis was carried out to examine whether the differences between the proposed hybrid approaches are statistically significant. Finally, the effects of various parameters including backorder costs, setup times, setup costs, capacity utilization and demand variability on performances of proposed hybrid approaches were investigated experimentally and it was observed that the performances of proposed hybrid approaches are quite robust.

7.2 Contributions

The original contributions of this thesis can be summarized as follows.

- This is the first extensive study employing GAs to solve the capacitated lot sizing problem with setup carryover and backordering.

  Although there have been numerous solution approaches proposed for solving lot sizing problems with different modeling features, there has not been enough emphasis on application of GAs for solving the capacitated lot sizing problem with setup times, setup carryover and backordering. Hence, to fill this research gap, we proposed a number of GA-based hybrid solution approaches.

- To our knowledge, this is also the first study proposing novel hybrid approaches that combine GAs with Fix-and-Optimize heuristic in lot sizing area. These hybrid approaches integrate two methods in two ways: sequential and embedded. To divide the problem into smaller ones the Fix-and-Optimize heuristic is implemented with the time decomposition scheme.

  Pure GAs can quickly identify promising areas in the search space; however they are not good at reaching the optimum in large complex search spaces. In such cases, the performance of GAs can still be improved. Therefore, in order to improve the performance of GAs in solving the CLSPC, we proposed two hybridization schemes.

  The first methodology hybridizes the GAs and Fix-and-Optimize heuristic sequentially, where the Fix-and-Optimize heuristic is implemented after GAs. The second methodology is an example of embedded hybridization where the Fix-and-Optimize heuristic is embedded into the loop of GAs.

- Second, to solve the $CLSP^+$, eight hybrid approaches are proposed.

  While the first four of these approaches are the modified versions of the sequential hybrid approach the others are developed by modifying the embedded hybrid approach. The Fix-and-Optimize heuristic in these hybrid approaches utilizes the product decomposition scheme besides the time decomposition scheme. Moreover, to divide the problem into smaller ones, these two decomposition schemes were combined in different ways while implementing the proposed hybrid approaches. An extensive experimental analysis was carried out to investigate the effects of these combinations on the performance of proposed approaches.

- Unlike the current relevant literature generally employing random initialization, in this study, a new approach is proposed to generate the initial population of GA.

  Considering the complexity of the problem studied in this thesis that requires a very large space to be searched for, we proposed a new population initialization scheme to improve the search efficiency. The proposed initialization scheme consists of both problem specific information and randomness. Problem specific chromosomes, called smart part were generated based on the solution of the LP relaxation of the CLSP, which was the basis of two problems considered in this study. An experimental study was carried out to determine the ratio of smart part to random part in order to obtain the best results.

- Problem-specific repair operators are proposed to fix the infeasibilities that occur after crossover and mutation operators.

  The feasibility control has an important effect on the genetic search. To maintain the feasibility during the search of GAs, infeasible individuals were converted into feasible ones by using some problem specific repair operators.

7.3 Future Research Directions

Future research directions can be summarized in two categories.

From the perspective of problem features, some of the future research directions can be stated as follows:

- This study considers the capacitated lot sizing problem on a single machine. To show the effectiveness of the proposed approaches in solving lot sizing problems, the CLSP with setup carryover and backordering can be extended to include features such as parallel machines, lost sales and sequence dependent setup times and costs and the proposed hybrid approaches can be modified to deal with these new features.

- In this study, we only consider the capacitated lot sizing problem in single level. The proposed hybrid approaches can also be modified to solve the capacitated lot sizing problems in multi level.

- This study focuses on an example of deterministic lot sizing problem where demand is assumed to be known. The proposed hybrid methodologies can be adapted to solve the stochastic version of the problem studied.

From the perspective of methodological research, some of the future research directions can be stated as follows:

- To improve the computational efficiency of the proposed hybrid approaches the computer programs implementing the proposed methods can be optimized so that they execute more rapidly.

- Another approach to improve the computational efficiency of the proposed approaches could be to design parallel GAs and distribute the basic elements of GAs to different processors which can work on different parts of the program at the same time.

- The problem can be solved in a multi-objective manner by using pareto optimization methods.

Lastly, the performance of proposed hybrid approaches can be investigated over real-life problems to identify the limitations of the proposed approaches.

REFERENCES

Absi, N., & Sidhoum, S.K. (2007). MIP-based heuristics for multi-item capacitated lot sizing problem with setup times and shortage costs. *RAIRO-Operations Research*, *41* (2), 171-192.

Afentakis, P. (1987). Parallel heuristic algorithm for lot-sizing in multistage production systems. *IIE Transactions, 19* (1), 34–42.

Akartunalı, K., & Miller, A.J. (2008). A heuristic approach for big bucket multi-level production planning problems. *European Journal of Operations Research*, *193* (2), 396-411.

Alfieri, A., Brandimarte, P., & D'Orazio, S. (2002). LP-based heuristics for the capacitated lot-sizing problem: the interaction of model formulation and solution algorithm. *International Journal of Production Research*, *40*, 441–458.

Almada-Lobo, B., Klabjan, D., Carravilla, & M.A., Oliveira, J.F. (2007). Single machine multi-product capacitated lot sizing with sequence-dependent setups. *International Journal of Production Research*, *45* (20), 4873-4894.

Almeder, C. (2010). A hybrid optimization approach for multi-level capacitated lot-sizing problem. *European Journal of Operational Research*, *200*, 599-606.

Aytuğ, H., Khouja, M., & Vergara, F. E. (2003). Use of genetic algorithms to solve production and operations management problems: A review. *International Journal of Production Research, 41* (17), 3955–4009.

Bahl, H.C. (1983). Column generation based heuristic algorithm for multi-item scheduling. *IIE Transactions*, *15*, 136–141.

Bahl, H.C., Ritzman, L. P., & Gupta, J. N. D. (1987). Determining lot sizes and resource requirements: A review. *Operations Research*, *35* (3), 329–345.

Barany, I., Van Roy T.J., &Wolsey, L.A. (1984). Strong formulations formulti-item capacitated lot sizing. *Management Science, 30*, 1255–1261.

Barbarosoğlu, G., & Özdamar, L. (2000). Analysis of solution space-dependent performance of simulated annealing: the case of the multi-level capacitated lot sizing problem. *Computers and Operations Research*, *27*, 895-903.

Belvaux, G., & Wolsey, L.A. (2000). bc-prod: a specialized branch-and-cut system for lot-sizing problems. *Management Science, 46*, 724–738.

Belvaux, G., & Wolsey, L.A. (2001). Modelling practical lot-sizing problems asmixed-integer programs. *Management Science, 47*, 993–1007.

Berretta, R., & Rodrigues, L.F. (2004). A memetic algorithm for a multistage capacitated lot-sizing problem. *International Journal of Production Economics, 87*, 67–81.

Berretta, R., França, P.M., & Armentano, V.A. (2005). Metaheuristic approaches for the multilevel resource constrainedlot-sizing problem with setup and lead times. *Asia-Pacific Journal of Operational Research*, *22*, 261–286.

Billington, P.J., McClain, J.O., & Thomas, L.J. (1986). Heuristics for multilevel lot-sizing with a bottleneck. *Management Science*, *32*, 989-1006.

Bitran, G.R., & Matsuo, H. (1986). The multi-item capacitated lot size problem: error bounds of Manne's formulations. *Management Science, 32*, 350–359.

Bomberger, E. (1966). A dynamic programming approach to a lot scheduling problem. *Management Science*, *12*, 778–784.

Boyabatlı, O. & Sabuncuoglu, I. (2004). Parameter Selection in Genetic Algorithms, Systemics, *Cybernetics and Informatics, 2* (4), 78-83.

Buschkühl, L., Sahling, F., Helber, S., & Tempelmeier, H. (2008). Dynamic capacitated lot sizing problems: a classification and review of solution approaches. *OR Spectrum*, *32*, 231-261.

Brahimi, N., Dauzère-Pérès, S., & Najid, NM. (2006). Capacitated multi-item lot-sizing problems with time windows. *Operations Research, 54*, 951–967.

Campbell, G.M., & Mabert, V.A. (1991). Cyclical schedules for capacitated lot sizing with dynamic demands. *Management Science, 37*, 409–427.

Caserta, M. & Voss, S. (2009). Metaheuristics: Intelligent problem solving. In: V. Maniezzo, T. Stützle and S. Voß (Eds.) *Matheuristics: Hybridizing Metaheuristics and Mathematical Programming, Book Series* "*Annals of Information Systems*," 10, Springer, Berlin, 1-38.

Cattrysse, D., Maes, J., & Van Wassenhove, L.N. (1990). Set partitioning and column generation heuristics for capacitated dynamic lotsizing. *European Journal of Operations Research, 46*, 38–47.

Cattrysse, D., Salomon, M., Kuik, R., Van Wassenhove, L.N. (1993). A dual ascent and column generation heuristic for the discrete lotsizing and scheduling problem with setup times. *Management Science*, *39* (4), 477-486.

Chang, P. T., Yao, M. J., Huang, S. F., & Chen, C. T. (2006). A genetic algorithm for solving economic fuzzy lot-size scheduling problem. *International of Production Economics, 102*, 265–288.

Chatfield, D. C. (2007). The economic lot scheduling problem: A pure genetic search approach. *Computers & Operations Research*, *34*, 2865–2881.

Chen, H., & Chu, C. (2003). A lagrangian relaxation approach for supply chain planning with order/setup costs and capacity constraints. *Journal of Systems Science and Systems Engineering*, *12*, 98–110.

Chen, W-H., Thizy, J-M. (1990). Analysis of relaxations for the multi-item capacitated lot-sizing problem. *Annals of Operations Research*, *26*, 29-72.

Clark, A.R., & Armentano, V.A. (1995). A heuristic for a resource-capacitated multi-stage lot-sizing problem with lead times. *Journal of Operational Research Society*, *46*, 1208-1222.

Coley, D. (2003). *An Introduction to Genetic Algorithms for Scientists and Engineers*. World Scientific Press, Singapore.

Dantzig, George B. & Wolfe, Philip (1960). Decomposition Principle for Linear Programs. *Operations Research*, *8*, 101–111.

De Bodt, M.A., Gelders, L.F., & Van Wassenhove, L.N. (1984). Lot Sizing under Dynamic Demand Conditions: A review. *Engineering Costs and Production Economiccs*, *8*, 165-187.

Defersha, F. M., & Chen, M., (2008). A Linear Programming Embedded Genetic Algorithm for an Integrated Dynamic Cell Formation and Lot Sizing Considering Product Quality. *European Journal of Operational Research*, *187*, 46-69.

De Jong, K.,A. (1993) Genetic algorithms are NOT function optimizers. In D. Whitley (Eds.), *Foundations of Genetic Algorithms 2.* Morgan Kaufmann, San Mateo, CA, 5–18.

Degraeve, Z., & Jans, R. (2007). A New Dantzig-Wolfe Reformulation and Branch-and-Price Algorithm for the Capacitated Lot Sizing Problem with Set Up Times. *Operations Research*, *55* (5), 909-920.

Dellaert, N., & Jeunet, J. (2000). Solving large unconstrainedmultilevel lot-sizing problems using a hybrid genetic algorithm. *International Journal of Production Research, 38* (5), 1083–1099.

Dellaert, N., Jeunet, J., & Jonard, N. (2000). A genetic algorithm to solve the general multi level lot-sizing problem with time-varying costs. *International Journal of Production Economics, 68*, 241–257.

Denizel, M., & Süral, H. (2006). On alternative mixed integer programming formulations and LP-based heuristics for lot-sizing with setup times. *Journal of the Operational Research Society*, *57*, 389-399.

Denizel, M., Altekin, F.T., Süral, H., & Stadtler, H. (2008). Equivalence of the LP relaxations of two strong formulations for the capacitated lot-sizing problem with setup times. *OR Spectrum, 30*, 773–785.

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

Diaby, M., Bahl, H.C., Karwan, M.H., & Zionts, S. (1992b). A lagrangean relaxation approach for very-large-scale capacitated lot-sizing. *Management Science, 38*, 1329–1340.

Dillenberger, C., Escudero, L.F., Wollensak, A., & Zhang, W. (1993). Operations research in production planning and control. in: G. Fandel, T. Gulledge, A. Jones (Ed.). *On solving a large-scale resource allocation problem in production planning* (105-119), Heidelberg, Springer.

Dillenberger, C., Escudero, L.F., Wollensak, A., & Zhang, W. (1994). On practical resource allocation for production planning and scheduling with period overlapping setups. *European Journal of Operations Research*, *75*, 275-286.

Dixon, P.S., & Silver, E.A. (1981). A heuristic solution procedure for themulti-item, single-level, limited capacity lot-sizing problem. *Journal of Operations Management*, *2*, 23–39.

Doğramacı, A., Panayiotopoulos, J.C., & Adam NR (1981). The dynamic lot-sizing problem for multiple items under limited capacity. *AIIE Transactions, 13*, 294–303.

Dorigo, M., Maniezzo, V., & Colorni A. (1996). Ant System: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man and Cybernetics - Part B*, *26* (1), 29-41.

Dorigo, M., Di Caro, G., & Gambardella, L. M. (1999). Ant algorithms for discrete optimization. *Artificial Life*, *5* (2), 37–172.

Drexl, A., & Haase, K., (1995). Proportional Lotsizing and Scheduling. *International Journal of Production Economics*, *40*, 73-87.

Drexl, A., & Haase, K., (1996). Sequential analysis based randomized-regret methods for lot-sizing and scheduling. *Journal of the Operational Research Society*, *47*, 251-265.

Drexl, A., & Kimms, A. (1997). Lot sizing and scheduling-survey and extensions. *European Journal of Operational Research*, *99*, 221– 235.

Duda, J. (2005). Lot-sizing in a foundry using genetic algorithm and repair functions. *Lecture Notes in Computer Science, 3448*, 101–111.

Duda, J., & Osyczka, A. (2007). A genetic algorithm for lot sizing optimization with a capacity loading criterion. *In the Proceedings of 2007 IEEE Congress on Evolutionary Computation*, September 25-28, Singapore.

Dzielinski, B.P., & Gomory, R.E. (1965). Optimal programming of lot sizes, inventory and labor allocations. *Management Science, 11*, 874–890.

Eiben, A.,E., Hinterding, R.,& Michalewicz, Z. (1999). Parameter control in evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, *3* (2), 124–141.

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

El-Mihoub, T.A., Hopgood, A.A., Nolle, L., & Battersby, A. (2006). Hybrid genetic algorithms: A review. *Engineering Letters*, *13*, 124-137.

Eppen, G.D., & Martin, R.K. (1987). Solving multi-item capacitated lot-sizing problems using variable redefinition. *Operations Research, 35*, 832–848.

Fakhrzad M.,B., & Zare, H.K. (2009). Combination of genetic algorithm with Lagrange multipliers for lot-size determination in multi-stage production scheduling problems. *Expert Systems with Applications*, *36*, 10180-10187.

Federgruen, A., Meissner, J., & Tzur, M. (2007). Progressive interval heuristics for multi-item capacitated lot-sizing problems. *Operations Research, 55*, 490–502.

Fleischmann, B. (1990). The Discrete Lot-Sizing and Scheduling Problem. *European Journal of Operational Research*, *44*, 337-348.

Fleischmann, B. (1994). The Discrete Lot-Sizing and Scheduling Problem with Sequence-Dependent Setup Costs. *European Journal of Operational Research*, *75*, 395-404.

Florian, M., Lenstra, J.K., & Rinnooy, Kan A.H.G., (1980). Deterministic production planning: Algorithms and complexity. *Management Science*, *26*, 669-679.

França, P.M., Armentano, V.A., Berretta, R.E., & Clark, A.R. (1997). A heuristic method for lot-sizing in multi-stage systems. *Computers and Operations Research, 24*, 861–874.

Gaafar, L. (2006). Applying genetic algorithms to dynamic lot sizing with batch ordering. *Computers & Industrial Engineering, 51*(3), 433–444.

Gao, Y. (1998). *Models and Algorithms for Multi-Item Dynamic Lot Sizing Problems with Complex Setup Structures*. Phd Thesis, Graduate Faculty of Auburn University.

Gen M., & Cheng R. (1997). *Genetic Algorithms and Engineering Design*. New York: John Wiley and Sons.

Glover, F. (1986). Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research*, *13*, 533–549.

Goldberg, D. E. (2002). *Design of innovation: Lessons from and for competent genetic algorithms*. Boston, MA: Kluwer Acadamic Publishers.

Gonçalves, J., F., & Sousa, P., S.A. (2011). A Genetic Algorithm for Lot Size and Scheduling under Capacity Constraints and Allowing Backorders. *International Journal of Production Research*, *49* (9), 2683-2703.

Gopalakrishnan, M., Ding, K., Bourjolly, J-M., & Mohan, S. (2001). A Tabu Search Heuristic for the Capacitated Lot-Sizing Problem with Setup Carryover. *Management Science*, *47* (6), 851-863.

Gören, G., H., Tunalı, S., & Jans, R. (2010). A Review of applications of genetic algorithms in lot sizing. *Journal of Intelligent Manufacturing*, *21* (4), 575-590.

Gören, G., H., Tunalı, S., & Jans, R. (2011). A Hybrid Approach for the Capacitated Lot Sizing Problem with Setup Carryover. *International Journal of Production Research* (Article in Press).

Groff, G. (1979). A lot sizing rule for time-phased component demand. *Production and Inventory Management*, *20*, 47–53.

Gupta, D. & Magnusson, T. (2005). The capacitated lot sizing and scheduling problem with sequence-dependent setup costs and setup times. *Computers & Operations Research*, *32*, 727-747.

Günther, H-O. (1987). Planning lot sizes and capacity requirements in a single stage production sytem. *European Journal of Operational Research, 31*, 223–231.

Hansen, P., & Mlaydenovi´c, N. (1999). Metaheuristics: advances and trends in local search paradigms for optimization in: S. Voß, S. Martello, I. Osman, C. Roucairol (Eds). *An introduction to variable neighborhood search* (433-458). Dordrecht: Kluwer.

Hansen, P., & Mlaydenovi´c, N. (2001). Variable neighborhood search: principles and applications. *European Journal of Operational Research, 130*, 449–467.

Harrison, T.P., & Lewis, H.S. (1996). Lot sizing in serial assembly systems with multiple constrained resources. *Management Science, 42*, 19–36.

Haupt, R. L., & Haupt, S. E. (2004). *Practical Genetic Algorithms*. Second edition with CD, Wiley, New York, NY.

Helber, S. (1995). Lot sizing in capacitated production planning and control systems. *OR Spektrum, 17*, 5–18.

Helber, S., & Sahling, F. (2010). A fix-and-optimize approach for the multi-level capacitated lot sizing problem. *International Journal of Production Economics*, *123*, 247-256.

Hernandez, W., & Süer, A. G. (1999). Genetic algorithms in lot sizing decisions. *Proceedings of the 1999 Congress on Evolutionary Computation, 3*, 2280–2286.

Heuvel, van den W., & Wagelmans, A. P. M. (2005). A comparison of methods for lot-sizing in a rolling horizon environment. *Operations Research Letters*, *33*, 486–496.

Hindi, K.S. (1996). Solving the CLSP by a tabu search heuristic. *The Journal of the Operational Research Society*, *47*, 151–161.

Hindi, K.S., Fleszar, K., & Charalambous, C. (2003). An effective heuristic for the CLSP with setup times. *The Journal of the Operational Research Society, 54*, 490–498.

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems*. The University of Michigan Press, Ann Arbor, Michigan.

Hop Van, N., & Tabucanon, M. T. (2005). Adaptive genetic algorithm for lot-sizing problem with self-adjustment operation rate. *International of Production Economics, 98*, 129–135.

Huisman, D., Jans, R., Peeters, M., & Wagelmans, A.P.M. (1 January 2003). *Combining column generation and Lagrangian relaxation.* Retrieved July 15, 2011 from http://ssrn.com/abstract=496704.

Hung, Y-F., & Hu, Y-C. (1998). Solving mixed integer programming production planning problems with setups by shadow price information. *Computers & Operations Research, 25*, 1027–1042.

Hung, Y. F., Shih, C.-C. & Chen, C.-P. (1999). Evolutionary algorithms for production planning problems with setup decisions. *The Journal of the Operational Research Society, 50* (8), 857–866.

Hung, Y-F., & Chien, K.L. (2000). A multi-class multi-level capacitated lot sizing model. *The Journal of Operational Research Society*, *51*, 1309–1318.

Hung, Y-F., Chen, C-P., Shih, C-C., & Hung, M-H. (2003). Using tabu search with ranking candidate list to solve production planning problems with setups. *Computers and Industrial Engineering, 45*, 615–634.

Jans, R. (2002). *Capacitated Lot Sizing Problems: New Applications, Formulations and Algorithms*. Phd Thesis, Katholieke Universiteit Leuven.

Jans, R., & Degraeve, Z. (2004). Improved Lower Bounds for the Capacitated Lot Sizing Problem with Set Up Times. *Operations Research Letters*, *32*, 185-195.

Jans, R., & Degraeve, Z. (2007). Meta-heuristics for dynamic lot sizing: A review and comparison of solution approaches. *European Journal of Operational Research*, *177* (3), 1855–1875.

Jans, R., & Degraeve, Z. (2008). Modeling industrial lot sizing: A review. *International Journal of Production Research, 46* (6), 1619– 1643.

Jordan, C., & Drexl, A. (1998). Discrete Lot Sizing and Scheduling by Batch Sequencing. *Management Science*, *44* (5), 698-713.

Jung, H., Song, I. & Jeong, B. (2007). Genetic algorithm-based integrated production planning considering manufacturing partners. *International Journal of Advanced Manufacturing Technology, 32*, 547 556.

Karmarkar, U.S., & Schrage, L. (1985). The Deterministic Dynamic Product Cycling Problem. *Operations Research*, *33* (2), 326-345.

Karmarkar, U.S., Kekre, S., & Kekre, S. (1987). The Dynamic Lot-Sizing Problem with Startup and Reservation Costs. *Operations Research*, *35* (3), 389-398.

Karimi, B., & Ghomi, F. (2002). A new heuristic for the CLSP problem with backlogging and setup carryover. *International Journal of Advanced Manufacturing Systems*, *5* (2), 66-77.

Karimi, B., Ghomi, S.M.T.F., & Wilson, J.M. (2003). The capacitated lot sizing problem: a review of models and algorithms. *Omega*, *31*, 365–378.

Karimi, B., Ghomi, Fatemi S.M.T., & Wilson, J.M. (2006). A tabu search heuristic for the CLSP with backlogging and setup carryover. *The Journal of the Operational Research Society*, *57*, 140-157.

Katok, E., Lewis, H.S, & Harrison, T.P. (1998). Lot sizing in general assembly systems with setup costs, setup times, and multiple constrained resources. *Management Science*, *44*, 859–877.

Khouja, M., Michalewics, Z., &Wilmot, M. (1998). The use of genetic algorithms to solve the economic lot size scheduling problem. *European Journal of Operational Research, 110*, 509–524.

Kırca, Ö., & Kökten, M. (1994). A new heuristic approach for the multi-item dynamic lot sizing problem. *European Journal of Operational Research, 75*, 332–341.

Kimms, A. (1996a). Multi-level, Single-Machine Lot Sizing and Scheduling (with Initial Inventory). *European Journal of Operational Research*, *89*, 86-99.

Kimms, A. (1996b). Competitive methods for multi-level lot sizing andscheduling: tabu search and randomized regrets. *International Journal of Production Research*, *34* (8), 2279-2298.

Kimms, A. (1999). A genetic algorithm for multi level, multi-machine lot sizing and scheduling. *Computers & Operations Research, 26*, 829–848.

Kirkpatrick, S., Gelatt, C.D., & Vecchi, M.P. (1983). Optimization by simulated annealing. *Science, 220*, 671–680.

Kohlmorgen, U., Schmeck, H., & Haase, K. (1999). Experience with fine-grained parallel genetic algorithms. *Annals of Operations Research, 90*, 203–219.

Krarup, J., & Bilde, O. (1977). in L. Collatz, G. Meinardus, W. Wetterling (Eds). Numerische Methoden bei Optimierungsaufgaben Band 3. *Plant location, set covering and economic lot size: An o(mn)-algorithm for structured problems* (155-180). Basel: Birkhäuser.

Kuik, R., & Salomon, M. (1990). Multi-level lot-sizing problem: evaluation of a simulated-annealing heuristic. *European Journal of Operational Research*, *45*, 25–37.

Kuik, R., Salomon, M., Van Wassenhove, L.N., & Maes, J. (1993). Linear programming, simulated annealing and tabu search heuristics for lotsizing in bottleneck assembly systems. *IIE Transactions, 25*, 62–72.

Kuik, R., Salomon, M., & Van Wassenhove, L. N. (1994). Batching decisions: Structure and models. *European Journal of Operational Research, 75*, 243–263.

Kuo, H., & Inman, R. (1990). A practical heuristic for the group technology economic lot scheduling problem. *International Journal of Production Research*, *28*, 709–722.

Kämpf, M., & Köchel, P. (2006). Simulation-based sequencing and lot size optimization for a production-and-inventory system with multiple items. *International Journal of Production Economics, 104* (1), 191–200.

Lambrecht, M.R., & Vanderveken, H. (1979). Heuristic procedures for the single operation, multi-item loading problem. *AIIE Transactions, 11*, 319–326.

Lasdon, L.S., & Terjung, R.C. (1971). An efficient algorithm for multi-item scheduling. *Operations Research*, *19*, 946–969.

Leung, J.M.Y., Magnanti, T.L. & Vachani, R. (1989). Facets and Algorithms for Capacitated Lot Sizing. *Mathematical Programming*, *45*, 331-359.

Li, Y., Chen, J., & Cai, X. (2007). Heuristic genetic algorithm for capacitated production planning problemswith batch processing and remanufacturing. *International Journal of Production Economics, 105* (2), 301–317.

Lozano, M., & García-Martínez, C. (2010). Hybrid metaheuristics with evolutionary algorithms specializing in intensification and diversification: Overview and progress report. *Computers & Operations Research*, *37* (3): 481-497.

Maaranen H., Miettinen K. & Mäkelä M. M. (2004). A quasi-random initial population for genetic algorithms, *Computers and Mathematics with Applications*, *47*, 1885-1895.

Maes, J., & VanWassenhove, L. (1988). Multi-item single-level capacitated dynamic lot-sizing heuristics: a general review. *The Journal of Operational Research Society*, *39*, 991–1004.

Maes, J., McClain, J.O., & VanWassenhove, L.N. (1991). Multilevel capacitated lotsizing complexity and LP-based heuristics. *European Journal of Operational Research, 53*, 131–148.

Manne, A.S. (1958). Programming of economic lot sizes. *Management Science, 4*, 115–135.

Megala, N., & Jawahar, N. (2006). Genetic algorithm and hopfield neural network for a dynamic lot sizing problem. *International Advanced Manufacturing Technologies, 27*, 1178–1191.

Mercé, C., & Fontan, G. (2003). MIP-based heuristics for capacitated lotsizing problems. *International Journal of Production Economics, 85*, 97–111.

Millar, H., & Yang, M., (1994). Lagrangian Heuristics for the capacitated multi-item lot-sizing problem with back-ordering. *International Journal of Production Economics*, *34*, 1-15.

Miller, A.J., Nemhauser, G.L., & Savelsbergh M.W.P. (2000). Solving multi-item capacitated lot-sizing problems with setup times by branch-and-cut. Université Catholique de Louvain, Belgium. Core discussion paper 00/39.

Mirabi, M., & Ghomi, S.M.T.F. (2010). A Hybrid Simulated Annealing for the Single Machine Capacitated Lot-Sizing and Scheduling Problem with Sequence-Dependent Setup Times and Costs. *In the Proceedings of International Conference on Industrial Engineering and Engineering Management*, December 7-10, Macao.

Mohammadi, M., & Ghomi, S.M.T.F. (2011). Genetic algorithm-based heuristic for capacitated lotsizing problem in flow shops with sequence-dependent setups. *Expert Systems with Applications* (Article In Press).

Moon, I., Silver, E. A., & Choi, S. (2002). Hybrid genetic algorithm for the economic lot-scheduling problem. International Journal of Production Research, *40* (4), 809–824.

Moon, I. K., Cha, B. C., & Bae, H. C. (2006). Hybrid genetic algorithm for group technology economic lot scheduling problem. *International Journal of Production Research, 44* (21), 4551–4568.

Moorkanat, J. (2000). Studies in certain resource loading, scheduling and production control problems in multi-stage production control problems in multi-stage production inventory systems. Phd. Thesis, Indian Institute of Technology, Bombay.

Nagano, M.S., Ruiz, R., & Lorena, L.A.N. (2008). A Constructive Genetic Algorithm for permutation flowshop scheduling. *Computers and Industrial Engineering*, *55*, 195-207.

Nahmias S. (2005). *Production and Operations Analysis* (5th ed.), Boston: Mc-Graw-Hill/ Irwin.

Nascimento, M.C., & Toledo, F.M.B. (2008). A Hybrid Heuristic for the Multi-Plant Capacitated Lot Sizing Problem with Setup Carry-Over. *Journal of the Brazilian Computer Society*, *14* (4), 7-15.

Newson, E.F.P. (1975a). Multi-item lot size scheduling by heuristic, part I: with fixed resources. *Management Science, 21*, 1186–1193.

Newson, E.F.P. (1975b). Multi-item lot size scheduling by heuristic, part II: with variable resources. *Management Science, 21*, 1194–1203.

Özdamar, L., & Barbarosoğlu, G., (1999). Hybrid heuristics for the multi-stage capacitated lot sizing and loading problem. *The Journal of the Operational Research Society*, *50*, 810–825.

Özdamar, L., & Barbarosoğlu, G., (2000). An integrated Lagrangean relaxation-simulated annealing approach to the multi-level multi-item capacitated lot sizing problem. *International Journal of Production Economics*, *68*, 319-331.

Özdamar, L., & Birbil, S. I., (1998). Hybrid heuristics for the capacitated lot sizing and loading problem with setup times and overtime decisions. *European Journal of Operational Research*, *110*, 525–547.

Özdamar, L., Birbil, S. I., & Portmann, M. C., (2002). Technical note: New results for the capacitated lot sizing problem with overtime decisions and setup times. *Production Planning and Control*, *13*, 2–10.

Özdamar, L., & Bozyel, M. A. (2000). Capacitated lot sizing problem with overtime decisions and setup times. *IIE Transactions*, *32* (11), 1043–1057.

Pham, D.T., & Karaboga, D. (2000). *Intelligent Optimisation Techniques*, New York: Springer.

Pitakaso, R., Almeder, C., Doerner, K.F., & Hartl, R.F. (2006). Combining exact and population-based methods for the constrained multilevel lot sizing problem. *International Journal of Production Research*, *44*, 4755–4771.

Pitakaso, R., Almeder, C., Doerner, K. F., & Hartl, R. F. (2007). A MAX-MIN ant system for unconstrained multi level lot-sizing problems. *Computers&Operations Research, 34* (9), 2533–2552.

Pochet, Y., & Wolsey, L.A. (1988). Lot-Size Models with Backlogging: Strong Reformulations and Cutting Planes. *Mathematical Programming*, *40*, 317-335.

Pochet, Y. (1988). Valid Inequalities and Separation for Capacitated Economic Lot Sizing. *Operations Research Letters*, *7* (3), 109-115.

Pochet, Y., & Wolsey, L.A. (1991). Solving multi-item lot-sizing problems using strong cutting planes. *Management Science*, *37*, 53-67.

Pochet, Y., & Wolsey, L.A. (1993). Lot-Sizing with Constant Batches: Formulation and Valid Inequalities. *Mathematics of Operations Research*, *18* (4), 767-785.

Pochet Y., & Wolsey L.A. (2000). *Production Planning by Mixed Integer Programming*, Springer, NewYork, NY.

Pongcharoen, P., Hicks, C., Braiden, P.M., & Stewardson, D.J. (2002). Determining optimum genetic algorithm parameters for scheduling the manufacturing and assembly of complex products, *International Journal of Production Economics, 78* (3), 311–322.

Porkka, P., Versalainen, P.J., & Kuula, M., (2003). Multiperiod production planning over setup time. *International Journal of Production Research*, *41* (6), 1133-1148.

Prasad, P. S. S., & Chetty, O. V. K. (2001). Multi-level lot sizing with a genetic algorithm under fixed and rolling horizons. *International Journal of Manufacturing Technology, 18*, 520–527.

Preux, Ph., & Talbi, E-G., (1999). Towards hybrid evolutionary algorithms. *International Transactions in Operational Research*, *6*, 557-570.

Quadt, D., & Kuhn, H. (2008). Capacitated lot-sizing with extensions: a review. 4*OR*: *A Quarterly Journal of Operations Research*, *6*, 61–83.

Quadt, D., & Kuhn, H., (2009). Capacitated lot-sizing and scheduling with parallel machines, back-orders, and setup carry-over. *Naval Research Logistics*, *56* (4), 366-384.

Rao, V.,D.,P., & Subbaiah, K.,V., Raju V.R. (2009). Fuzzy Genetic Approach to Economic Lot-Size Scheduling Problem. *Jordan Journal of Mechanical and Industrial Engineering*, *3* (1), 9-16.

Raidl, G., R. (2006). A unified view on hybrid metaheuristics. *Lecture Notes in Computer Science*, *4030*, 1-12.

Rosling, K. (1986). Optimal lot-sizing for dynamic assembly systems. In: Axsäter S, Schneeweiss C, Silver E (Eds.) *Multi-stage production planning and inventory control*. Springer, Berlin, 119–131.

Sahling, F., Buschkühl, L., Tempelmeier, H., & Helber, S., (2009). Solving a multi-level capacitated lot sizing problem with multi-period setup carry-over via a fix-and-optimize heuristic. *Computers and Operations Research*, *36* (9), 2546-2553.

Salomon, M., Kuik, R., & Van Wassenhove, L.N. (1993). Statistical search methods for lotsizing problems. *Annals of Operations Research*, *41*, 453–468.

Sambasivan, M., & Yahya, S. (2005). A Lagrangean-based heuristic for multi-plant, multi-item, multi-period capacitated lot-sizing problems with inter-plant transfers. *Computers & Operations Research*, *32*, 537–555.

Santos, M.O., Massago, S., & Almada-Lobo, B. (2010). Infeasibility handling in genetic algorithm using nested domains for production planning. *Computers & Operations Research*, *37*, 1113-1122.

Sarker, R., & Newton, C. (2002). A genetic algorithm for solving economic lot size scheduling problem. *Computers & Industrial Engineering, 42*, 189–198.

Silver, E.A., & Meal, H.C. (1969). A simple modification of the EOQ for the case of a varying demand rate. *Production and Inventory Management*, *10*, 51–55.

Silver, E.A., & Meal, H.C. (1973). A heuristic for selecting lot size quantities for the case of a deterministic timevarying demand rate and discrete opportunities for replenishment. *Production and Inventory Management, 14*, 64–74.

Simpson, N.C. (2001). Questioning the relative virtues of dynamic lot sizing rules. *Computers and Operations Research*, *28*, 899-914.

Sox, C.R., & Gao, Y. (1999). The capacitated lot sizing problem with setup carry-over. *IIE Transactions*, *31*, 173–181.

Srinivas, M., & Patnaik, L. M. (1994). Adaptive probabilities of crossover and mutation in genetic algorithms. *IEEE Transactions on Systems, Man, and Cybernetics, 24* (4), 656–667.

Stadtler, H. (1996). Mixed integer programming model formulations for dynamicmulti-item multi-level capacitated lotsizing. *European Journal of Operational Research*, *94*, 561–581.

Stadtler, H. (1997). Reformulations of the shortest routemodel for dynamicmulti-item multi-level capacitated lotsizing. *OR Spektrum, 19*, 87–96.

Stadtler, H. (2003). Multilevel lot sizing with setup times and multiple constrained resources: internally rolling schedules with lot-sizing windows. *Operations Research*, *51*, 487–502.

Sun, H., Huang, H., & Jaruphongsa, W. (2009). A genetic algorithm for the economic lot scheduling problem under the extended basic period and power-of-two policy. *CIRP Journal of Manufacturing Science and Technology*, *2*, 29-34.

Suerie, C., & Stadtler, H. (2003). The Capacitated Lot-sizing problem with Linked Lot sizes. *Management Science*, *49* (8), 1039-1054.

Süral, H., Denizel, M., & Van Wassenhove, L. (2009). Lagrangean relaxation based heuristics for lot sizing with setup times. *European Journal of Operational Research*, *194*, 51-63.

Talbi, E. G. (2002). A Taxonomy of Hybrid Metaheuristics. *Journal of Heuristics*, *8*, 541–564.

Tang, O. (2004). Simulated annealing in lot sizing problems. *International Journal of Production Economics, 88* (2), 173–181.

Taşan Özmehmet, S. (2007). *Solving Simple and Mixed-Model Assembly Line Balancing Problems Using Hybrid Metaheuristic Approaches*. Phd. Thesis, Dokuz Eylül University Graduate School of Natural and Applied Science, Department of Industrial Engineering, Izmir.

Tempelmeier, H., & Helber, S. (1994). A heuristic for dynamic multi-item multi-level capacitated lotsizing for general product structures. *European Journal of Operational Research*, *75*, 296–311.

Tempelmeier, H., & Derstroff, M. (1993). Mehrstufige Mehrprodukt-Losgrößenplanung bei beschränkten Ressourcen und genereller Erzeugnisstruktur. *OR Spektrum, 15*, 63–73.

Tempelmeier, H., & Derstroff, M. (1996). A Lagrangean-based heuristic for dynamic multilevel multiitem constrained lotsizing with setup times. *Management Science*, *42*, 738–757.

Tempelmeier, H. (2010). A Column Generation Heuristic for Dynamic Capacitated Lot Sizing with Random Demand under a Fillrate Constraint. *In the Proceedings of International Conference on Industrial Engineering and Engineering Management*, December 7-10, Macao.

Trigeiro, W.W. (1987). A dual-cost heuristic for the capacitated lot sizing problem. *IIE Transactions*, *19*, 67–72.

Trigeiro, W.W., Thomas, L.J., & McClain, J.O. (1989). Capacitated Lot Sizing with setup times. *Management Science*, *35*, 353-366.

Trigeiro, W.W. (1989). A simple heuristic for lot sizing with setup times. *Decision Sciences*, *20*, 294-303.

Van Wyne, M. (2003). Algorithms for single item constant capacity in lot sizing problems. In Series CORE Discussion Papers with number 2003007.

Yao, M. J., & Huang, J. X. (2005). Solving the economic lot scheduling problem with deteriorating items using genetic algorithms. *Journal of Food Engineering, 70*, 309–322.

Wagelmans, A., Van Hoesel, S., & Kolen, A. (1992). Economic lot sizing: An o (n log n) algorithm that runs in linear time in the Wagner–Whitin case. *Operations Research*, *40*, 145–156.

Wagner, H. M., & Whitin, T. M. (1958). Dynamic version of the economic lot size model. *Management Science*, *5* (1), 89–96.

Wolsey, L.A. (1989). Uncapacitated Lot-Sizing Problems with Start-Up costs. *Operations Research*, *37* (5), 741-747.

Wolsey, L. A. (1995). Progress with single-item lot-sizing. *European Journal of Operational Research, 86*, 395–401.

Wu, T., Shi, L., & Duffie, N.A. (2010). An HNP-MP Approach for the Capacitated Multi-Item Lot Sizing Problem With Setup Times. *IEEE Transactions on Automation Science and Engineering*, *7* (3), 500-511.

Xiao, Y., & Kaku, I. (2010). Variable Neighborhood Search Based Approach for Solving Multilevel Lot-Sizing Problems, *In The Proceedings of the 9th WSEAS International Conference on Applications of Computer Engineering*, March 23-25, Penang, Malaysia.

Xie, J. (1995).An application of genetic algorithms for general dynamic lot sizing problems. *In The Proceedings of Genetic Algorithms in Engineering Systems: Innovations and Applications*, September 12-14, Sheffield, UK.

Xie, J., & Dong, J. (2002). Heuristic genetic algorithms for general capacitated lot sizing problems. *Computers & Mathematics with Applications*, *44*, 263–276.

APPENDICES

APPENDIX A

MULTIPLE COMPARISONS FOR CHAPTER 6

Appendix A1 Multiple comparisons for Class 2

Proposed Hybrid Approach (I)	Proposed Hybrid Approaches (J)	Mean Difference (I-J)	Sig. (*p*)
H1	H2	-8.2600E-02*	0.000
	H3	-1.2000E-03	1.000
	H4	-6.0000E-04	1.000
	H5	2.400E-03	1.000
	H6	-4.9400E-02*	0.000
	H7	2.400E-03	1.000
	H8	3.400E-03	1.000
H2	H1	8.260E-02*	0.000
	H3	8.140E-02*	0.000
	H4	8.200E-02*	0.000
	H5	8.500E-02*	0.000
	H6	3.320E-02	0.067
	H7	8.500E-02*	0.000
	H8	8.600E-02*	0.000
H3	H1	1.200E-03	1.000
	H2	-8.1400E-02*	0.000
	H4	6.000E-04	1.000
	H5	3.600E-03	1.000
	H6	-4.8200E-02*	0.001
	H7	3.600E-03	1.000
	H8	4.600E-03	1.000
H4	H1	6.000E-04	1.000
	H2	-8.2000E-02*	0.000
	H3	-6.0000E-04	1.000
	H5	3.000E-03	1.000
	H6	-4.8800E-02*	0.000
	H7	3.000E-03	1.000
	H8	4.000E-03	1.000
H5	H1	-2.4000E-03	1.000
	H2	-8.5000E-02*	0.000
	H3	-3.6000E-03	1.000
	H4	-3.0000E-03	1.000
	H6	-5.1800E-02*	0.000
	H7	0.0000	1.000
	H8	1.000E-03	1.000
H6	H1	4.940E-02*	0.000
	H2	-3.3200E-02	0.067
	H3	4.820E-02*	0.001
	H4	4.880E-02*	0.000
	H5	5.180E-02*	0.000
	H7	5.180E-02*	0.000
	H8	5.280E-02*	0.000

Appendix A1 Multiple comparisons for Class 2 (cont.)

H7	H1	-2.4000E-03	1.000
	H2	-8.5000E-02*	0.000
	H3	-3.6000E-03	1.000
	H4	-3.0000E-03	1.000
	H5	0.0000	1.000
	H6	-5.1800E-02*	0.000
	H8	1.000E-03	1.000
H8	H1	-3.4000E-03	1.000
	H2	-8.6000E-02*	0.000
	H3	-4.6000E-03	1.000
	H4	-4.0000E-03	1.000
	H5	-1.0000E-03	1.000
	H6	-5.2800E-02*	0.000
	H7	-1.0000E-03	1.000

* shows that the mean difference is significant at the 0.05 level

Appendix A2 Multiple comparisons for Class 3

Proposed Hybrid Approach (I)	Proposed Hybrid Approaches (J)	Mean Difference (I-J)	Sig. (*p*)
H1	H2	-4.9800E-02*	0.000
	H3	3.200E-03	0.999
	H4	1.800E-03	1.000
	H5	4.800E-03	0.983
	H6	-2.4600E-02*	0.000
	H7	4.000E-03	0.994
	H8	5.600E-03	0.960
H2	H1	4.980E-02*	0.000
	H3	5.300E-02*	0.000
	H4	5.160E-02*	0.000
	H5	5.460E-02*	0.000
	H6	2.520E-02*	0.000
	H7	5.380E-02*	0.000
	H8	5.540E-02*	0.000
H3	H1	-3.2000E-03	0.999
	H2	-5.3000E-02*	0.000
	H4	-1.4000E-03	1.000
	H5	1.600E-03	1.000
	H6	-2.7800E-02*	0.000
	H7	8.000E-04	1.000
	H8	2.400E-03	1.000
H4	H1	-1.8000E-03	1.000
	H2	-5.1600E-02*	0.000
	H3	1.400E-03	1.000
	H5	3.000E-03	0.999
	H6	-2.6400E-02*	0.000
	H7	2.200E-03	1.000
	H8	3.800E-03	0.996

Appendix A2 Multiple comparisons for Class 3 (cont.)

H5	H1	-4.8000E-03	0.983
	H2	-5.4600E-02*	0.000
	H3	-1.6000E-03	1.000
	H4	-3.0000E-03	0.999
	H6	-2.9400E-02*	0.000
	H7	-8.0000E-04	1.000
	H8	8.000E-04	1.000
H6	H1	2.460E-02*	0.000
	H2	-2.5200E-02*	0.000
	H3	2.780E-02*	0.000
	H4	2.640E-02*	0.000
	H5	2.940E-02*	0.000
	H7	2.860E-02*	0.000
	H8	3.020E-02*	0.000
H7	H1	-4.0000E-03	0.994
	H2	-5.3800E-02*	0.000
	H3	-8.0000E-04	1.000
	H4	-2.2000E-03	1.000
	H5	8.000E-04	1.000
	H6	-2.8600E-02*	0.000
	H8	1.600E-03	1.000
H8	H1	-5.6000E-03	0.960
	H2	-5.5400E-02*	0.000
	H3	-2.4000E-03	1.000
	H4	-3.8000E-03	0.996
	H5	-8.0000E-04	1.000
	H6	-3.0200E-02*	0.000
	H7	-1.6000E-03	1.000

* shows that the mean difference is significant at the 0.05 level

Appendix A3 Multiple comparisons for Class 4

Proposed Hybrid Approach (I)	Proposed Hybrid Approaches (J)	Mean Difference (I-J)	Sig. (*p*)
H1	H2	-3.7800E-02*	0.000
	H3	1.000E-03	1.000
	H4	6.400E-03	0.968
	H5	-7.6000E-03	0.920
	H6	-2.7000E-02*	0.000
	H7	-1.1000E-02	0.624
	H8	-5.2000E-03	0.990
H2	H1	3.780E-02*	0.000
	H3	3.880E-02*	0.000
	H4	4.420E-02*	0.000
	H5	3.020E-02*	0.000
	H6	1.080E-02	0.646
	H7	2.680E-02*	0.000
	H8	3.260E-02*	0.000

Appendix A3 Multiple comparisons for Class 4 (cont.)

H3	H1	-1.0000E-03	1.000
	H2	-3.8800E-02*	0.000
	H4	5.400E-03	0.988
	H5	-8.6000E-03	0.856
	H6	-2.8000E-02*	0.000
	H7	-1.2000E-02	0.511
	H8	-6.2000E-03	0.973
H4	H1	-6.4000E-03	0.968
	H2	-4.4200E-02*	0.000
	H3	-5.4000E-03	0.988
	H5	-1.4000E-02	0.303
	H6	-3.3400E-02*	0.000
	H7	-1.7400E-02	0.086
	H8	-1.1600E-02	0.556
H5	H1	7.600E-03	0.920
	H2	-3.0200E-02*	0.000
	H3	8.600E-03	0.856
	H4	1.400E-02	0.303
	H6	-1.9400E-02*	0.033
	H7	-3.4000E-03	0.999
	H8	2.400E-03	1.000
H6	H1	2.700E-02*	0.000
	H2	-1.0800E-02	0.646
	H3	2.800E-02*	0.000
	H4	3.340E-02*	0.000
	H5	1.940E-02*	0.033
	H7	1.600E-02	0.152
	H8	2.180E-02*	0.009
H7	H1	1.100E-02	0.624
	H2	-2.6800E-02*	0.000
	H3	1.200E-02	0.511
	H4	1.740E-02	0.086
	H5	3.400E-03	0.999
	H6	-1.6000E-02	0.152
	H8	5.800E-03	0.981
H8	H1	5.200E-03	0.990
	H2	-3.2600E-02*	0.000
	H3	6.200E-03	0.973
	H4	1.160E-02	0.556
	H5	-2.4000E-03	1.000
	H6	-2.1800E-02*	0.009
	H7	-5.8000E-03	0.981

* shows that the mean difference is significant at the 0.05 level

Appendix A4 Multiple comparisons for Class 5

Proposed Hybrid Approach (I)	Proposed Hybrid Approaches (J)	Mean Difference (I-J)	Sig. (p)
H1	H2	-1.0000E-02*	0.000
	H3	8.800E-03*	0.000
	H4	9.600E-03*	0.000
	H5	4.800E-03	0.128
	H6	-3.6000E-03	0.473
	H7	6.400E-03*	0.008
	H8	9.600E-03*	0.000
H2	H1	1.000E-02*	0.000
	H3	1.880E-02*	0.000
	H4	1.960E-02*	0.000
	H5	1.480E-02*	0.000
	H6	6.400E-03*	0.008
	H7	1.640E-02*	0.000
	H8	1.960E-02*	0.000
H3	H1	-8.8000E-03*	0.000
	H2	-1.8800E-02*	0.000
	H4	8.000E-04	1.000
	H5	-4.0000E-03	0.330
	H6	-1.2400E-02*	0.000
	H7	-2.4000E-03	0.883
	H8	8.000E-04	1.000
H4	H1	-9.6000E-03*	0.000
	H2	-1.9600E-02*	0.000
	H3	-8.0000E-04	1.000
	H5	-4.8000E-03	0.128
	H6	-1.3200E-02*	0.000
	H7	-3.2000E-03	0.628
	H8	-2.0817E-17	1.000
H5	H1	-4.8000E-03	0.128
	H2	-1.4800E-02*	0.000
	H3	4.000E-03	0.330
	H4	4.800E-03	0.128
	H6	-8.4000E-03*	0.000
	H7	1.600E-03	0.987
	H8	4.800E-03	0.128
H6	H1	3.600E-03	0.473
	H2	-6.4000E-03*	0.008
	H3	1.240E-02*	0.000
	H4	1.320E-02*	0.000
	H5	8.400E-03*	0.000
	H7	1.000E-02*	0.000
	H8	1.320E-02*	0.000

Appendix A4 Multiple comparisons for Class 5 (cont.)

H7	H1	-6.4000E-03*	0.008
	H2	-1.6400E-02*	0.000
	H3	2.400E-03	0.883
	H4	3.200E-03	0.628
	H5	-1.6000E-03	0.987
	H6	-1.0000E-02*	0.000
	H8	3.200E-03	0.628
H8	H1	-9.6000E-03*	0.000
	H2	-1.9600E-02*	0.000
	H3	-8.0000E-04	1.000
	H4	2.082E-17	1.000
	H5	-4.8000E-03	0.128
	H6	-1.3200E-02*	0.000
	H7	-3.2000E-03	0.628

* shows that the mean difference is significant at the 0.05 level

Appendix A5 Multiple comparisons for Class 6

Proposed Hybrid Approach (I)	Proposed Hybrid Approaches (J)	Mean Difference (I-J)	Sig. (*p*)
H1	H2	-2.0000E-04	1.000
	H3	1.180E-02*	0.000
	H4	1.100E-02*	0.000
	H5	-1.2000E-03	0.999
	H6	2.000E-03	0.968
	H7	3.400E-03	0.642
	H8	5.800E-03	0.052
H2	H1	2.000E-04	1.000
	H3	1.200E-02*	0.000
	H4	1.120E-02*	0.000
	H5	-1.0000E-03	1.000
	H6	2.200E-03	0.947
	H7	3.600E-03	0.570
	H8	6.000E-03*	0.038
H3	H1	-1.1800E-02*	0.000
	H2	-1.2000E-02*	0.000
	H4	-8.0000E-04	1.000
	H5	-1.3000E-02*	0.000
	H6	-9.8000E-03*	0.000
	H7	-8.4000E-03*	0.000
	H8	-6.0000E-03*	0.038
H4	H1	-1.1000E-02*	0.000
	H2	-1.1200E-02*	0.000
	H3	8.000E-04	1.000
	H5	-1.2200E-02*	0.000
	H6	-9.0000E-03*	0.000
	H7	-7.6000E-03*	0.002
	H8	-5.2000E-03	0.121

Appendix A5 Multiple comparisons for Class 6 (cont.)

H5	H1	1.200E-03	0.999
	H2	1.000E-03	1.000
	H3	1.300E-02*	0.000
	H4	1.220E-02*	0.000
	H6	3.200E-03	0.711
	H7	4.600E-03	0.245
	H8	7.000E-03*	0.007
H6	H1	-2.0000E-03	0.968
	H2	-2.2000E-03	0.947
	H3	9.800E-03*	0.000
	H4	9.000E-03*	0.000
	H5	-3.2000E-03	0.711
	H7	1.400E-03	0.996
	H8	3.800E-03	0.498
H7	H1	-3.4000E-03	0.642
	H2	-3.6000E-03	0.570
	H3	8.400E-03*	0.000
	H4	7.600E-03*	0.002
	H5	-4.6000E-03	0.245
	H6	-1.4000E-03	0.996
	H8	2.400E-03	0.917
H8	H1	-5.8000E-03	0.052
	H2	-6.0000E-03*	0.038
	H3	6.000E-03*	0.038
	H4	5.200E-03	0.121
	H5	-7.0000E-03*	0.007
	H6	-3.8000E-03	0.498
	H7	-2.4000E-03	0.917

* shows that the mean difference is significant at the 0.05 level