Appl Intell (2015) 42:805–816
DOI 10.1007/s10489-014-0626-x

Solving the capacitated lot sizing problem with setup carryover using a new sequential hybrid approach

Hacer Güner Gören · Semra Tunalı

Published online: 31 December 2014

Abstract The aim of lot sizing problems is to determine the periods where production takes place and the quantities to be produced in order to satisfy the customer demand while minimizing the total cost. Due to its importance on the efficiency of the production and inventory systems, lot sizing problems are one of the most challenging production planning problems and have been studied for many years with different modelling features. Among these problems, the capacitated lot sizing problem (CLSP) has received a lot of attention from researchers. Having motivated from our earlier study, this study proposes a new hybrid approach for solving the CLSP with the extension of setup carryover. Moreover, the initialization scheme proposed in the earlier study has also been investigated comprehensively. Lastly, an experimental study evaluating the solution quality of the proposed approach is carried out using various problem instances and promising results are obtained when compared to the recent results in the literature.

Keywords Production planning · Lot sizing · Setup carryover · Sequential hybrid approach · Genetic algorithms · Fix-and-optimize heuristic

H. Güner Gören (✉)
Department of Industrial Engineering, Kinikli Campus, Pamukkale University, 20070, Denizli, Turkey
e-mail: hgoren@pau.edu.tr

S. Tunalı
Department of Business Administration, Izmir University of Economics, Sakarya Caddesi, No: 156, 35330, Balcova-Izmir, Turkey
e-mail: semra.tunali@izmirekonomi.edu.tr

1 Introduction

Today's competitive world places a tremendous pressure on companies to use their resources efficiently and satisfy customer requirements on time. Achieving these goals requires one to deal with a series of production planning problems. Among these problems, the capacitated lot sizing problem (CLSP) has received a lot of attention from researchers. CLSP is a typical example of big bucket models where the length of a period is long enough to produce more than one product. The CLSP considers only the quantities and timings of the production and ignores the sequence of the products within a period [1]. Different from big bucket models, small bucket models (i.e. Discrete Lot Sizing and Scheduling Problem, Proportional Lot Sizing and Scheduling, Continous Setup LotSizing and Scheduling) also deal with the sequencing decisions. In recent years, a new model combining the properties of the big bucket and small bucket models, called CLSP with Setup Carryover (CLSPC) has emerged. Like big bucket models, the CLSPC allows producing more than one product per period and like small bucket models it carries over the setup state of a product from one period to the other. From the solution of the problem, it is possible to obtain the production quantities along with the semi-sequencing (i.e. first and last products produced in a period) of the products in a period. The Capacitated Lot Sizing Problem with Setup Carryover is the extension of the Capacitated Lot Sizing Problem (CLSP) which is known for its computational intractability. Florian et al. [2] have shown that the general case of the single-item CLSP is NP Hard. Trigeiro et al. [3] state that when set-up times are introduced in the multi-item CLSP, even the feasibility problem becomes NP Complete. Naturally, it becomes even harder to solve the CLSPC. Since it is computationally very difficult to find optimal solutions to large-scale problems, nearly all earlier studies employed heuristic approaches. The heuristic approaches dealing with this problem can be classified as mathematical programming based approaches (Dillenberger et al. [4], Sox and Gao [5], Suerie and Stadtler [6], Sahling et al.[7], Helber and Sahling [8], Lang and Shen [9], Xiao et al. [10]), metaheuristics (Gopalakrashinan et al. [11], Karimi et al. [12], Nascimento and Toledo [13]), greedy heuristics (Haase [14], Haase [15], Karimi and Ghomi [16], Gupta and Magnusson [17] and hybrid approaches (Gören et al. [1], Seanner et al. [18]). Table 1 chronologically presents the studies conducted during the last 20 years based on the solution approach proposed, type, properties and complexity of the lot sizing problems. It should be noted that the complexity is given in terms of the number of binary variables in the mathematical model of the lot sizing problem considered in the study. Unlike most conventional methods and some of the metaheuristics (i.e. tabu search, simulated annealing) which conduct a single directional search, GAs perform multiple directional searches using a set of candidate solutions, require no domain knowledge and use stochastic transition rules to guide the search (Gen and Cheng [19]). Different applications of GAs can be found in literature such as maritime transportation (Kang et al. [20]), vehicle routing (Ombuki et al. [21]) or grouping (Korkmaz [22]) problems. However, it is well known that pure GAs can locate the promising regions for global optima in a search space, but in a large and complex solution space, they have difficulty in finding the exact minimum/maximum of these optima. Therefore, in this study, to further improve the performance of GAs in solving the CLSPC, a new hybrid approach is proposed. This study can be seen as the continuation of our earlier study Gören et al. [1] in which Fix-and-Optimize heuristic is embedded into the loop of GA to refine the solutions during the generations of GA. Different from our earlier study, this study suggests using a sequential hybridization scheme in which the GAs and then Fix-and-Optimize heuristic are run sequentially. By doing so, we hope that the GAs can locate the promising region where good quality solutions exist and then the Fix-and-Optimize heuristic further searches this region for the best solution. Another issue affecting the performance of the meta-heuristics is the choice of the initial solution/solutions (Mohammadi and Ghomi [23]). 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. In this study, to further improve the performance of the proposed heuristic, we used the initialization scheme introduced in Gören et al. [1] which creates half of the initial population randomly and the other half using problem specific information. Encouraged from success of this initialization scheme in our previous study, in this study we aimed at using this scheme more effectively. For this purpose, we carried out a comprehensive experimental analysis to determine the most appopriate portion of the random and problem-specific part for each problem size.

The remainder of the study is organized as follows. In the following section, the capacitated lot sizing problem with setup carryover is defined. The details of the proposed hybrid approach to solve this problem is presented in Section 3. Section 4 presents the results of experimental study comparing the proposed approach to the recent results in literature. Finally, the last section summarizes the findings of this study and presents some future research directions.

The Capacitated Lot Sizing Problem with Setup Carryover (CLSPC)

In this study, to solve the CLSPC the model proposed by Suerie and Stadtler [6] has been employed under the following assumptions:

- The planning horizon T is fixed and divided into time buckets $(1, \ldots.., T)$.
- Several products requiring a unique set-up state are produced on each resource in each period (property of a big bucket model).
- The resource consumption to produce a product j on a resource 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, such that no setup activity is necessary in the second period.
- 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.

The mixed integer programming model (MIP) with the inventory and lot size representation is given with the sets, indices, parameters and variables in the following.

Sets and indices:

j:	items $j \in P = \{1, 2, 3, \ldots P\}$
t:	periods $t \in T = \{1, 2, 3, \ldots T\}$

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}	demand for item j in period t

Table 1 Proposed heuristic approaches for solving the capacitated lot sizing problem with setup carryover

Reference	Year	Type of lot sizing problem	Properties of the problem	Complexity	Solution approach
Dillenberger et al. [4]	1993	Single level	Capacitated/single machine/setup carryover	NT	Fix-and-relax heuristic
Haase [14]	1994	Single level	Capacitated/single machine/setup carryover	NT	Greedy heuristic
Haase [15]	1998	Single level	Capacitated/single machine/setup carryover	NT	Greedy heuristic
Sox and Gao [5]	1999	Single level	Capacitated/single machine/setup carryover	NT	Lagrangian decomposition heuristic
Gopalakrishnan et al. [11]	2001	Single level	Capacitated/single machine/setup carryover	NT	Tabu search
Karimi and Ghomi [16]	2002	Single level	Capacitated/single machine/setup carryover/backlogging	NT	Greedy heuristic
Suerie and Stadtler [6]	2003	Single level	Capacitated/single machine/setup carryover	NT	Time decomposition heuristic
Gupta and Magnusson [17]	2005	Single level	Capacitated/single machine/setup carryover/sequence dependent setup times	NT	Greedy heuristic
Karimi et al. [12]	2006	Single level	Capacitated/single machine/setup carryover/ backlogging	NT	Tabu search
Nascimento and Toledo [13]	2008	Single level	Capacitated/single machine/setup carryover/ multi plants	PNT	Greedy Randomized Adaptive Search Procedure (GRASP)
Sahling et al. [7]	2009	Multi level	Capacitated/setup carryover	NT	Fix-and-optimize heuristic
Helber and Sahling [8]	2010	Multi level	Capacitated/setup carryover/lead times	NT	Fix-and-optimize heuristic
Lang and Shen [9]	2011	Single level	Capacitated/single machine/setup carryover/ substitutions/sequence dependent setups	NT	Fix-and-optimize heuristic/ Relax-and-fix heuristic
Gören et al. [1]	2012	Single level	Capacitated/single machine/setup carryover	NT	Hybrid approach (Genetic Algorithms +Fix-and-optimize heuristic)
Seanner et al. [18]	2013	Multi level	Capacitated/setup carryover	MNT	Hybrid approach (Variable neighborhood decomposition search + Fix-and-optimize heuristic)
Xiao et al. [10]	2013	Single level	Capacitated/parallel machines/setup carryover/ sequence dependent setups/time windows/preference constraints	MNT(N+1)	Fix-and-optimize heuristic

N: number of products T: number of periods M: number of machines P: number of plants

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 resource is occupied solely by item i in period t (=1) or not (=0).

$$Min \sum_{j=1}^{P}\sum_{t=1}^{T}\left(sc_j Y_{jt} + h_j I_{jt}\right) \tag{1}$$

$s.t.$

$$I_{j,t-1} + X_{jt} - I_{jt} = d_{jt} \quad \forall j \in P;\ \forall t \in T \tag{2}$$

$$\sum_{j \in K}\left(a_j X_{jt} + st_j Y_{jt}\right) \leq C_t \quad \forall t \in T \tag{3}$$

$$\sum_{j \in K} W_{jt} \leq 1 \qquad \forall t \in \{2, ..., T\} \tag{4}$$

$$W_{jt} \leq Y_{jt-1} + W_{jt-1} \quad \forall j \in P,\ \forall t \in \{2, ..., T\} \tag{5}$$

$$W_{jt+1} + W_{jt} \leq 1 + Q_t \quad \forall\, t \in \{1, ..., T-1\} \tag{6}$$

$$Y_{jt} + Q_t \leq 1 \quad \forall\, t \in \{1, ..., T-1\} \tag{7}$$

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

$$Q_t \geq 0 \quad \forall\, t \in \{1, ..., T-1\} \tag{9}$$

$$W_{jt} \in \{0, 1\} \quad \left(W_{j1} = 0\right), \quad Y_{jt} \in \{0, 1\}\ \forall j \in P; \forall t \in T \tag{10}$$

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

The minimization of inventory holding and setup costs is given in the objective function (1). The inventory balance equations are stated in constraints (2). Constraints (3) show the capacity constraints. Constraints (4) state that at most one setup state can be preserved from one period to the next. Constraints (5) ensure 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). The upper bounds on the production quantities are expressed in constraints (8). Finally, the non-negativity conditions on production and inventory quantities and the binary nature of setup variables are given by (9) to (11). It is assumed that there are no setup carryovers in the first period as stated in constraints (10).

2 The proposed hybrid approach

The main idea of the proposed sequential hybrid approach is to run GAs for a predetermined number of generations and use the overall best solution as the initial solution for the Fix-and-Optimize heuristic. The Fix-and-Optimize heuristic improves the best solution coming from the GAs throughout the iterations. In doing so, we hope that the GAs will locate a promising region where good quality solutions exist and then the Fix-and-Optimize heuristic will refine this region to obtain the best solution.

The setup $\left(Y_{jt}\right)$ and setup carryover $\left(W_{jt}\right)$ variables given in the MIP model are the main elements in the search of the GAs. The values of these binary variables are stated in each chromosome. Fixing these values to zeros and ones in the MIP model results in a linear programming (LP) model from which the objective function value is easily calculated. Therefore, each solution visited in the search space is a chromosome and its fitness is the sum of the objective value of the LP model and fixed setup costs.

2.1 Fix-and-optimize heuristic

Fix-and-Optimize heuristic is a MIP based heuristic in which a sequence of MIP models is solved over all real-valued decision variables and a subset of the binary variables. Note that the number of binary setup and setup carryover variables determines the numerical effort required to solve the MIP model. In recent years, the Fix-and-Optimize heuristic has been widely used in lot sizing area. Sahling et al. [7] employ the Fix-and-Optimize heuristic to solve the multi level capacitated lot sizing problem with setup times and setup carryover and they test product, resource and process oriented decomposition strategies. Lang and Shen [9] develop Fix-and-Optimize and Relax-and-Fix heuristics to address the single machine capacitated lot sizing problem with item substitution options. A new heuristic which combines the principles of Variable Neighborhood Decomposition Search and Fix-and-Optimize heuristic is presented in Seeanner et al. [18] to solve the multi level lot sizing and scheduling problems. Two different Fix-and-Optimize heuristics are presented in Xiao et al. [10] to deal with the

CLSP with sequence-dependent setup times, time windows, machine eligibility and preference constraints.

The idea in the Fix-and-Optimize heuristic is to systematically solve a series of sub-problems that are derived from the model stated in Section 2. In each iteration of the algorithm, one sub-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 in a given sub-problem. Then the problem is solved using a standard MIP solver. In the next iteration, a new sub-problem with a different subset of fixed binary variables is solved. The solution with respect to the binary variables is a fixed parameter for the next MIPs that optimize other binary variables [8]. Therefore, in each sub-problem the complete set of real-valued decision variables are considered. The optimization of the model is done within the MIP solver and therefore the approach is flexible [7]. The definition of the sub-problem and algorithm are given in detail in the following sections.

Definition of sub-problems

Simply, a sub-problem can be derived from the model given in Section 2 by adding the following constraints.

$$Y_{kt} = \bar{Y}_{kt} \quad \forall (k,t) \in PT^{fix}_{Y,s} \tag{12}$$

$$W_{kt} = \bar{W}_{kt} \quad \forall (k,t) \in PT^{fix}_{W,s} \tag{13}$$

The explanations of the parameters used above can be found in Table 2. Fix-and-Optimize heuristic starts with an initial solution. This initial solution yields an initial objective value which is shown by Z^{new}. After initialization, the algorithm iterates through the ordered set of sub-problems according to time decomposition either once or until it reaches a local optimum.

Time decomposition: With respect to the setup and setup carryover variables, each sub problem k is defined for a time window of five consecutive periods. For example, in one iteration of the Fix-and-Optimize heuristic, three sub-problems for a problem with 15 periods. (The reader can refer to Gören et al. [1] for further details on time decomposition).

The algorithm

The basic structure of the algorithm is outlined in Fig. 1. The algorithm needs an initial solution to start and goes through the sub-problems defined by time decomposition either once ($\ell = \ell_{\max}$) or until it reaches a local optimum.

Each solution to a sub-problem yields an objective value of Z which is at least as good as the current best solution. So, a new solution is accepted only if it yields an objective value lower than the current best solution.

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

2.2 Genetic algorithms

Chromosome representation

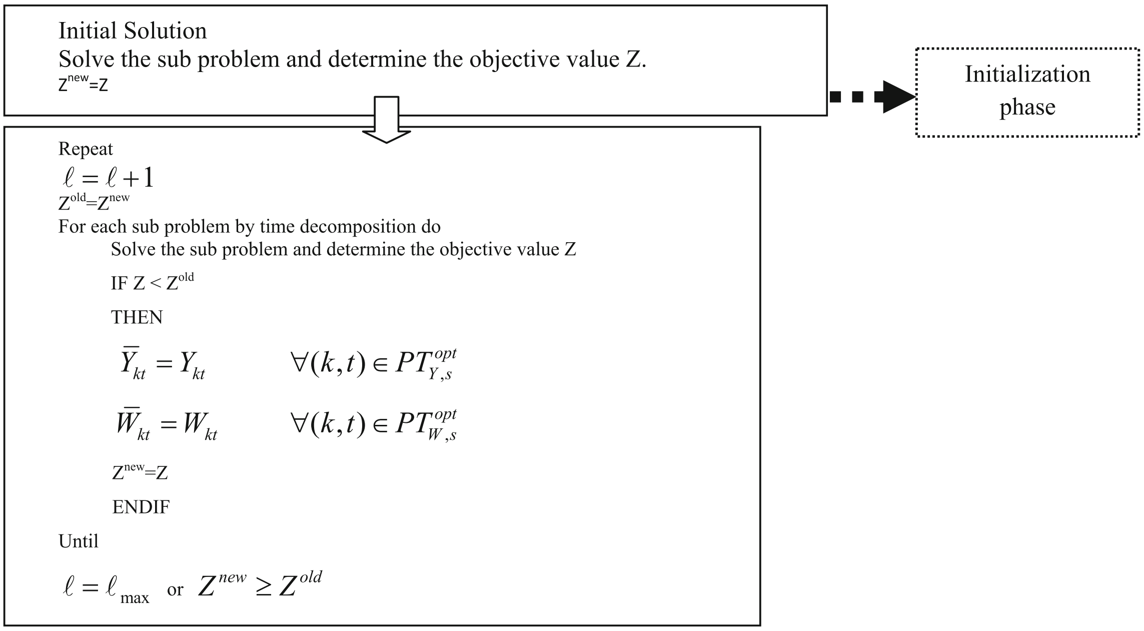
A matrix is used for representing the chromosomes (see Gören et al. [1]). The setup variables (Y_{jt}) and setup carryover variables (W_{jt}) are stated in the rows of the chromosomes. The setup and setup carryover variables for each product in each period are stated in the columns of the matrix. Thus, a chromosome with the length of the number of products (P) multiplied by the number of periods (T) is obtained.

Proposed Initialization scheme

The initialization scheme generates the chromosomes in two steps. First, setup variables which take place in the first row of the chromosomes are generated. Next, setup

Table 2 Additional notation for the definition of a sub-problem

Sets:	
$(k,t) \in PT$	Set of all product-period combinations
$PT^{opt}_{Y,s} \subseteq PT$	Set of product-period combinations which binary setup variables Y_{kt} are optimized in the current sub-problem
$PT^{opt}_{W,s} \subseteq PT$	Set of product-period combinations which binary setup carryover W_{kt} variables are optimized in the current sub-problem
$PT^{fix}_{Y,s} \subseteq PT$	Set of product-period combinations which binary setup variables Y_{kt} are fixed in the current sub-problem
$PT^{fix}_{W,s} \subseteq PT$	Set of product-period combinations which binary setup carryover variables W_{kt} are fixed in the current sub-problem
Parameters:	
$\bar{Y}_{kt}$	Exogenous value of the fixed setup variable Y_{kt}
$\bar{W}_{kt}$	Exogenous value of the fixed setup carryover variable W_{kt}



```
Initial Solution
Solve the sub problem and determine the objective value Z.
$Z^{new}$=Z

Repeat
$\ell = \ell + 1$
$Z^{old}$=$Z^{new}$
For each sub problem by time decomposition do
    Solve the sub problem and determine the objective value Z
    IF Z < $Z^{old}$
    THEN
    $\bar{Y}_{kt} = Y_{kt}$    $\forall (k,t) \in PT^{opt}_{Y,s}$
    $\bar{W}_{kt} = W_{kt}$    $\forall (k,t) \in PT^{opt}_{W,s}$
    $Z^{new}$=Z
    ENDIF
Until
$\ell = \ell_{\max}$ or $Z^{new} \geq Z^{old}$
```

Fig. 1 The outline of the Fix-and-Optimize heuristic

carryover variables in the second row of the chromosomes are created (Gören et al. [1]). In creating the first rows of chromosomes, i.e. setup variables, some of them which we call *Smart Part* are created by utilizing the information gained from the LP relaxation of the CLSP with setup times, and the others called *Random Part* are created randomly. The intention is that using problem specific information for creating the smart part of the initial population will direct the search towards the search spaces where feasible and good quality solutions exist. The procedure for creating the Smart and Random parts of initial population is explained in Fig. 2. Note that λ shows the number of chromosomes that are generated using problem specific information. The results of an experimental study evaluating different levels of this parameter are given in Section 4. β is another threshold parameter which is used for determining the values of genes in the chromosomes of the Random Part. Based on pilot experiments, this parameter is set to 0.5.

Once the setup variables are generated, the products are prioritized based on descending order of setup costs, next the setup carryover variables are generated following the logic given in the pseudo-code below. It must be noted that the setup carryover variables are generated starting from the second period (see the constraint (10) in the MIP model). A setup carryover for a product in the current period can only be generated if there is a setup and no setup carryover in the previous period. As can be seen in the pseudo-code below, if there is a setup variable in the current period, it is eliminated to attain the feasibility.

```
$i$ =1 to $POP$
 $t$ =2 to $T$
  $j$ =1 to $P$ (Products are descending order based on 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)
```

Genetic operators

The modified roulette wheel selection operator defined in Gören et al. [1] is used for selecting the chromosomes for recombination. The single point crossover and single bit flip mutation operators are used as the crossover and mutation operators, respectively. Infeasible chromosomes that violate the constraints related to the semi-sequencing of the products in a period (constraints 4, 5, 6, 7) are repaired and they are put back into the population. Moreover, to reduce the chance of selecting chromosomes that violate the capacity constraint stated in (3), a very high penalty cost is added to the fitness function. Elitism is used as a survival scheme and the search in GAs is terminated when the total number of generations exceeds a maximum number.

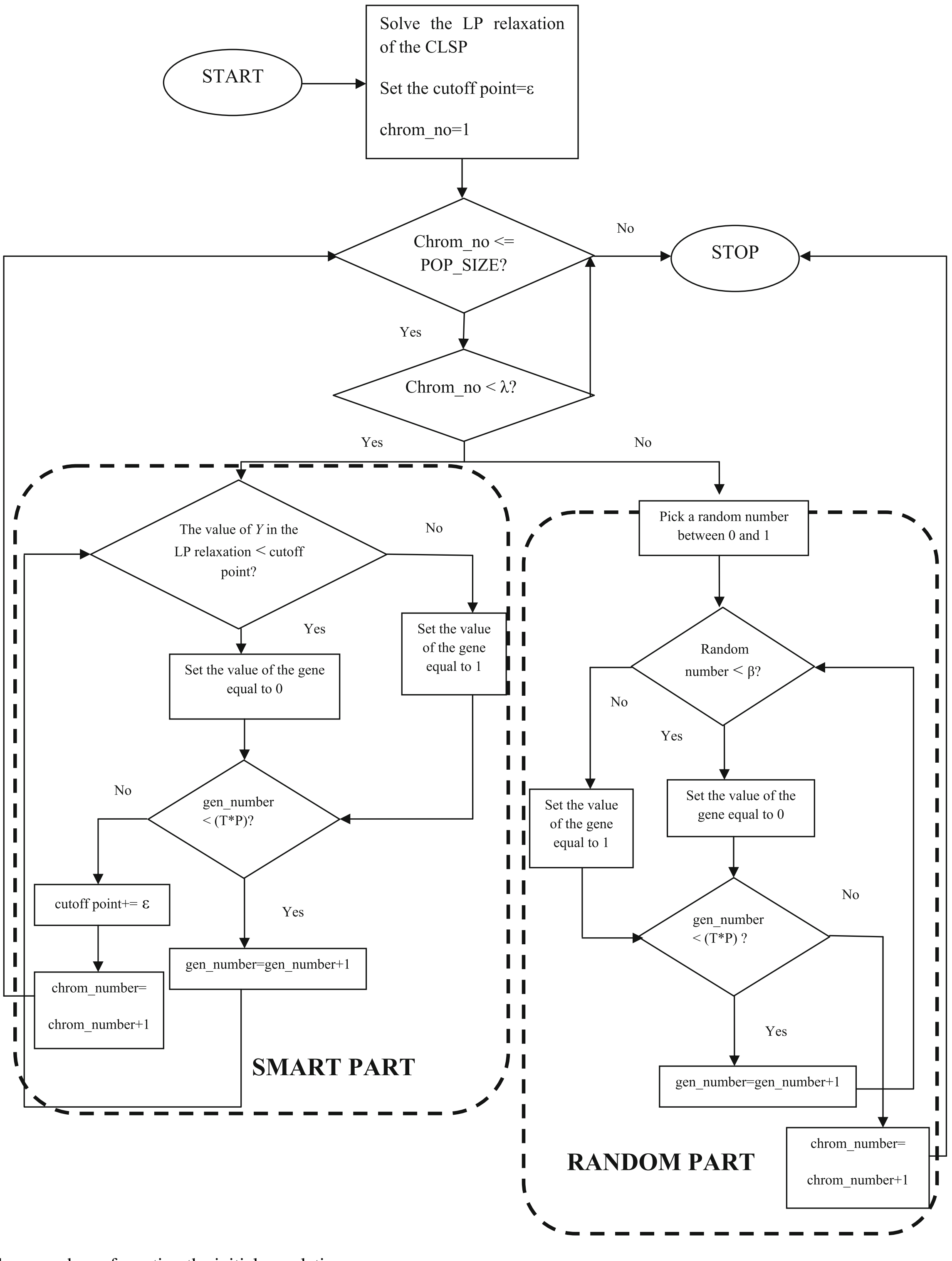

START
Solve the LP relaxation of the CLSP
Set the cutoff point=ε
chrom_no=1
Chrom_no <= POP_SIZE?
No
STOP
Yes
Chrom_no < λ?
Yes
No
The value of Y in the LP relaxation < cutoff point?
No
Yes
Set the value of the gene equal to 0
Set the value of the gene equal to 1
No
gen_number < (T*P)?
cutoff point+= ε
Yes
gen_number=gen_number+1
chrom_number= chrom_number+1
SMART PART
Pick a random number between 0 and 1
Random number < β?
No
Yes
Set the value of the gene equal to 1
Set the value of the gene equal to 0
gen_number < (T*P) ?
No
Yes
gen_number=gen_number+1
chrom_number= chrom_number+1
RANDOM PART

Fig. 2 The procedure of creating the initial population

3 Computational results

Computational experiments involve two sets of experiments. First, an extensive experimental study was carried out to decide the ratio of smart part to random part in the initial solution by employing three groups of problems (i.e. small, medium and large size). Second, using these best performing ratios for each problem size, the performance of the proposed hybrid approach was compared to that of pure GAs and the latest results reported in the literature, Suerie

Table 3 Classification of TTM-test set [6]

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

and Stadtler [6] and Gören et al. [1]. 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.

3.1 Benchmark problems

Trigeiro et al. [3] proposed an algorithm (referred to as TTM, Trigeiro-Thomas-McClain) to solve the CLSP and 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. Table 3 gives the classification of these instances [6]. It should be noted that our experimental study does not include the problem instances used in phase one since no results have been reported in the literature for these instances to carry out a comparative study.

Based on problem complexity, these nine problem classes have been placed into three groups, small, medium and large (See Table 4). Note that problem complexity is measured based on the number of binary variables (number of periods*number of products).

3.2 Experiments for the initialization strategy

The performances of initialization strategies were investigated on three groups of problems (small, medium, large).

Table 4 Problem sizes

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

Five problem instances from each problem class (see Table 3) were solved using the pure GAs with the initial population created by the proposed initialization strategies. As stated earlier, 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. 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. It should be noted that only feasible solutions are taken into consideration while calculating the average gap.

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 [6].

$$Gap = 100 * \frac{(heuristic\ solution - lower\ bound)}{lower\ bound} \tag{14}$$

The results of computational experiments are summarized in Table 6. It should be noted that the computational time is not taken into consideration in comparisons since nearly the same computational time is observed across all problem sizes (i.e. small, medium, large).

In Table 6, 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.

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 and 95 infeasible solutions out of 100 runs, for small and medium size problems, respectively. Note that it is not even possible to find a feasible solution for large size problems.

As stated in Section 3, the initialization scheme employed in this study 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. The purpose is to use this information 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 6, the results of experimental studies support our expectation and when some problem specific information is added into the initial

Table 5 Experimental design for the ratio of random part to smart part

	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**

population generation scheme, GA finds feasible solutions in nearly all runs for all problem sizes. It is also quite clear from Table 6 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. When both randomness and problem specific information are included in the initial population, the ratios of random part to smart part giving minimum average gap are found as RS^{02}, RS and RS^{04} for small, medium and large size problems, respectively. Having observed the same behavior for all problem sizes we can state that keeping the portion of the smart part bigger than the portion of the random part improves the performance of the proposed initialization scheme.

Based on these results, it can be stated that the 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. Other insights gained as a result of these experimental studies are that generating some part of the initial population randomly and keeping the smart part larger than random part further improve the performance of the proposed initialization scheme.

3.3 Comparative experiments

This section presents the results of the computational studies comparing the performance of the pure GAs and proposed sequential hybrid approach 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 sequential hybrid approach and pure GAs (PGA). The

Table 6 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

Table 7 Parameter settings for each problem size

	Small	Medium	Large
Population Size/Generation number	50/200	20/500	100/100
Crossover rate	0.9	0.5	0.9
Mutation rate	0.005	0.003	0.001

experiments were repeated 10 times for every instance and the solutions with the best and average performance were recorded. We also carried out some preliminary experiments and ANOVA to determine the most appopriate GAs parameters (see Table 7).

To obtain compatible test results for each problem class, the computational times for the proposed hybrid approach were set to the average computational time (i.e. the time calculated based on the computational times of 10 runs) of the PGA. For instance, the proposed hybrid approach was run for 58.75 seconds for each run to solve the problem instances in Class 1 (see Table 8). 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. Lastly, based on the experimental studies presented in earlier section, the ratio of random part to small part in the initial population of GAs was set to RS^{02}, RS and RS^{04}, for small, medium and large size problems, respectively.

When compared to the PGA, the solution quality of the proposed sequential hybrid approach which uses the Fix-and-Optimize heuristic after GAs is quite remarkable for every problem class. As seen in Table 8, the average gaps obtained by the hybrid approach are much smaller than those of the PGA. Table 8 also presents the average gaps reported in Gören et al. [1] and Suerie and Stadtler [6]. Considering the average of ten runs the proposed sequential hybrid approach outperforms the time decomposition heuristic of Suerie and Stadtler (2003) for the problem instances in Classes 7, 8, and 9. Since problem classes 8 and 9 include large size problem instances we might state that the performance of proposed hybrid approach improves as problem size increases. However, regarding the computational time, the proposed hybrid approach requires much more computational effort than the time decomposition heuristic.

When compared to the results obtained in Gören et al. [1], the solution quality of the proposed approach both with respect to average of ten runs and the best of ten runs shows a better performance in almost every problem class. This could be attributed to the two factors. As mentioned earlier, while in Gören et al. [1] Fix-and-Optimize heuristic is embedded into the GA, in this study, the GAs and Fix-and-Optimize heuristic are run sequentially. Besides employing a new hybridization scheme, in this study, we carried out a comprehensive experimental analysis to determine the best performing ratios of random to small part in generation of the initial population. Hence, unlike Gören et al. [1] in which half of the initial population was created randomly while the other half was created smartly, in this study, these best performing ratios were used when generating the initial population. 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

Table 8 Computational results

Class	Suerie and Stadtler (2003)		Goren et al. (2012)			PGA			Proposed sequential approach	
			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. Gap
1	2.53 %	5.2 s	3.09 %	65.94 s	2.45 %	8.39 %	58.75 s	7.19 %	**3.02 %**	2.68 %
2	2.32 %	8.8 s	2.97 %	44.83 s	2.22 %	8.20 %	72.88 s	7.24 %	**2.78 %**	2.64 %
3	0.95 %	6.2 s	1.78 %	59.87 s	1.24 %	5.31 %	72.65 s	4.71 %	1.87 %	**0.96 %**
4	0.72 %	11.8 s	1.67 %	77.13 s	1.19 %	5.13 %	77.13 s	4.66 %	**1.56 %**	**1.15 %**
5	0.54 %	9.2 s	1.22 %	98.70 s	0.80 %	3.05 %	98.70 s	2.6 %	1.65 %	1.07 %
6	0.30 %	13.2 s	2.19 %	114.1 s	1.65 %	2.95 %	144.21 s	2.79 %	**2.10 %**	**1.50 %**
7	3.14 %	9.2 s	2.25 %	68.66 s	1.73 %	4.87 %	48.91 s	4.12 %	**2.10 %**	**1.71 %**
8	2.73 %	12.7 s	1.76 %	100.2 s	1.38 %	2.81 %	99.94 s	2.53 %	**1.58 %**	**1.25 %**
9	2.21 %	18.0 s	1.61 %	121.45 s	1.37%	1.95 %	127.74 s	1.75 %	**1.54 %**	**1.29 %**

MIP-based heuristic in a sequential way significantly improves its performance.

4 Conclusion and future research directions

Lot sizing problems determine when and how many items (i.e. lot size) of a particular product to produce for a given horizon. Lot sizing problem is one of the most challenging production planning problems and solving this problem optimally has an important effect on the efficiency of the production and inventory systems.

In this study, we proposed a sequential hybrid approach for solving the CLSP with the extension of the setup carryover, CLSPC. The proposed approach combines Fix-and-optimize heuristic with GAs and the hybridization scheme is a sequential scheme where Fix-and-Optimize heuristic approach is executed after GAs. To evaluate the performances of pure GAs and proposed hybrid approach, 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 and the solution quality of the proposed approach is superior compared to the recent results in the literature.

As a future research direction, the proposed hybrid approach can be extended to solve the multi-level capacitated lot sizing problems. This study focuses on a deterministic lot sizing problem where demand is assumed to be known. Another future research direction can be to solve the stochastic version of this problem using the proposed hybrid method. The recent trend in lot sizing field is to build multi objective problems which integrate lot sizing, distribution and scheduling decisions. Solving these multiobjective problems using evolutionary approaches such as GAs can be another interesting future research direction.

References

1. Gören Güner H, Tunali S, Jans R (2012) A hybrid approach for the capacitated lot sizing problem with setup carryover. Int J of Produc Res 50(6):1582–1597
2. Florian M, Lenstra JK, Rinnooy Kan AHG (1980) Deterministic production planning: Algorithms and complexity. Manag Sci 26:669–679
3. Trigeiro W, Thomas LJ, McClain JO (1989) Capacitated Lot Sizing with setup times. Manag Sci 35:353–366
4. Dillenberger C, Escudero LF, Wollensak A, Zhang W (1993) Operations research in production planning and control. In: Fandel G, Gulledge T, Jones A (eds) On solving a large-scale resource allocation problem in production planning. Springer, Heidelberg, pp 105–119
5. Sox CR, Gao Y (1999) The capacitated lot-sizing problem with set-up carryover. IIE IIE Transac 31:173–181
6. Suerie C, Stadtler H (2003) The capacitated lot-sizing problem with linked lot sizes. Manag Sci 49(8):1039–1054
7. 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. Computer and Op Res 36(9):2546–2553
8. Helber S, Sahling F (2010) A fix-and-optimize approach for the multi-level capacitated lot sizing problem. Int J of Prod Econ 123:247–256
9. Lang JC, Shen Z-JM (2011) Fix-and-optimize heuristics for capacitated lot-sizing with sequence-dependent setups and substitutions. European J of Oper Res 3:214
10. Xiao J, Zhang C, Zheng L, Gupta JND (2013) MIP-based fix-and-optimise algorithms for the parallel machine capacitated lot-sizing and scheduling problem. Int J of Produc Res 51(16):5011–5028
11. Gopalakrishnan M, Ding K, Bourjolly J-M, Mohan S. (2001) A Tabu search heuristic for the capacitated lot-sizing problem with set-up carryover. Manag Sci 47(6):851–863
12. Karimi B, Ghomi FSMT, Wilson JM (2006) A tabu search heuristic for the CLSP with backlogging and set-up carryover. J of the Opl Res Soc 2006 57:140–157
13. Nascimento MC, Toledo FMB (2008) A hybrid heuristic for the multi-plant capacitated lot sizing problem with setup carry-over. J of the Braz Computer Soc 14(4):7–15
14. Haase K (1994) Lot sizing and scheduling for Production Planning. In: Lecture Notes in Economics and Mathematical Systems. Springer, Berlin, p 408
15. Haase K (1998) Capacitated lot-sizing with linked production quantities of adjacent periods. In: Drexl A, Kimms A (eds) Beyond Manufacturing Resource Planning (MRP II). Advanced Models and Methods for Production Planning. Springer, Berlin, pp 127-146
16. Karimi B, Ghomi F (2002) A new heuristic for the CLSP problem with backlogging and set-up carryover. Int J of Adv Manuf Syst 5(2):66–77
17. Gupta D, Magnusson T (2005) The capacitated lot sizing and scheduling problem with sequence-dependent set-up costs and set-up times. Comp Op Res 32:727–747
18. Seanner F, Almada-Lobo B, Meyr H (2013) Combining the principles of variable neighborhood decomposition and the fix-and-optimize heuristic to solve the multi-level lot-sizing and scheduling problems. Comp Op Res 40:303–317
19. Gen M, Cheng R (1997) Genetic Algorithms and Engineering Design. John Wiley and Sons, New York
20. Kang MH, Choi HR, Kim HS, Park BJ (2012) Development of a maritime transportation planning support system for car carriers based on genetic algorithm. Appl Intell 36(3):585–604
21. Ombuki B, Ross BJ, Hanshar F (2006) Multi-objective genetic algorithms for vehicle routing problem with time windows. Appl Intell 24(1):17–30
22. Korkmaz EE (2010) Multi-objective genetic algorithms for grouping problems. Appl Intell 33(2):179–192
23. Mohammadi M, Ghomi FSMT (2011) Genetic algorithm-based heuristic for capacitated lot sizing problem in flow shops with sequence-dependent setups. Expert Syst Appl 38(6):7201–7207

Hacer Güner Gören is an Assistant Professor at the Department of Industrial Engineering at Pamukkale University, Turkey. She received her BS degree in Industrial Engineering from Dokuz Eylul University, Turkey in 2002, her MS degree in Industrial Engineering from Pamukkale University, Turkey in 2005 and her PhD degree in Industrial Engineering from Dokuz Eylul University, Turkey in 2011. Her main research interests are modelling and optimisation algorithms for production planning problems and multi criteria decision making.

Semra Tunalı received her BS degree in Industrial Engineering in 1978, her MS degree in Applied Statistics in 1980 from Ege University, Turkey, her MBA degree from University of Missouri, USA in 1985, and her PhD degree in Computer Science from Ege University, Turkey in 1991. Before joining Izmir University of Economics in 2010, she worked at department of Industrial Engineering, Dokuz Eylul University, Turkey, and she has been as a Visiting Professor at GINTIC Institute of Manufacturing Technology in Singapore, Northwestern University in Chicago, USA, and Loyola University in Chicago, USA. Her main research interest is performance improvement of manufacturing systems using simulation modeling and meta-heuristics.