International Journal of Supply and Operations Management

IJSOM

May 2017, Volume 4, Issue 2, pp. 167- 179
ISSN-Print: 2383-1359
ISSN-Online: 2383-2525
www.ijsom.com

A New Mathematical Model for Simultaneous Lot-sizing and Production Scheduling Problems Considering Earliness/Tardiness Penalties and Setup Costs

Parinaz Vaez[*, a]

[a] *Department of Industrial Engineering, Isfahan University of Technology, Isfahan, Iran*

Abstract
This paper investigated the problem of simultaneous determination of lot-sizing and production scheduling with earliness/tardiness penalties. In this problem, decisions about lot-sizing and scheduling are made so that the sum of holding, tardiness, and setup costs is minimized. There are n orders waiting to be processed on a machine. Each order has its own due date as well as tardiness and earliness cost being the same as holding cost .Each order is delivered only once. If the production is completed before or on the due date, delivery will be on the due date. Otherwise, the order will be delivered immediately after its production is completed. In spite of its wide applications, this problem has not yet been reported in the literature. A mathematical model was presented as solution methods for the problem. Two meta-heuristics, namely, Simulated Annealing and Ant Colony System meta-heuristic algorithms are presented for solving the problem. Also, lower bounds are obtained from solving the problem relaxation, and they are compared with the optimal solutions to estimate the goodness of two meta-heuristic algorithms. They are difficult benchmarks, widely used to measure the efficiency of metaheuristics with respect to both the quality of the solutions and the central. The results show that the Simulated Annealing recorded a lower solution time and average percentage deviation than did the Ant Colony System algorithm. The presented SA is capable to solve large instances that are mostly compatible with the real-world problems.

Keywords: Scheduling; Lot-sizing; Earliness/Tardiness; Simulated annealing; Ant colony system.

1. Introduction

Scheduling and lot sizing are a part of factor production planning and control. Most of the time, decisions about these two problems are made hierarchically (Sereshti and Bijari 2013). Smooth and cost-efficient operation of a factory often depends on managerial ability in the selection of lot sizes and processing times. Managers must decide which products to make during which periods and the exact production sequence and production quantities to minimize the cost function (Gupta and Magnusson, 2005). Fleischmann and Meyr (1997) consider lot-sizing and scheduling problems simultaneously due to their interdependency, the problem that was named the general lot-sizing and scheduling problem (GLSP). Recent research on scheduling problems use the just-in-time (JIT) philosophy. In JIT production, it is not desirable for customers to receive their orders later or sooner than the stated delivery time. JIT scheduling is based on the production of items only for the required quantities and at the right time (Monden, 1998). The strategy of JIT delivery imposes extra costs on producers despite their limited capacity, because, if the product is completed earlier than the delivery time, it occupies storehouse space until the delivery time is reached, which obviously imposes additional costs. This is termed the earliness/tardiness (E/T) scheduling problem. Simultaneous scheduling and lot-sizing decisions are not considered in the general E/T problem. The purpose of the model developed in this paper is to integrate the GLSP with the idea of minimization of earliness/tardiness costs, earliness cost being the same as holding cost.

The rest of the paper is organized as follows. In Section 2, a literature review is presented. Problem definition is presented in Section 3. In Section 4, a mathematical model is developed. Section 5 devoted to calculation of lower bound. The metaheuristic algorithms for the solution to the problem including the Simulated Annealing (SA) and the Ant Colony

Corresponding author email address: vaez123123@yahoo.com

System (ACS) algorithms are presented in Section 6. In Section 7, numerical results obtained from the comparison of the two algorithms are presented. Finally, conclusions are presented in Section 8.

2. Literature Review

For modelling the GLSP problem two distinct approaches might be adopted. In the first approach, there are two kinds of time buckets: small buckets, and large ones (Fleischmann and Meyr 1997). Small bucket problems break the planning horizon in small time periods which limits the number of products manufactured in a single period. Several papers on simultaneous lot-sizing and scheduling problems have developed models and methods to solve this problem (Clark and Clark (2000); Mahdieh, Bijari and Clark (2011); Mohammadi (2010); Mohammadi and Jafari (2011);Meyr (2013); Seeanner et al. (2013))

The second approach is based on the capacitated lot-sizing problem (CLSP) with sequence-dependent setup times (CLSPSD). This problem is related to the travelling salesman problem (TSP) or the vehicle routing problem (VRP). Setup costs in GLSP are like the distance cost in TSP. Because TSP is an NP-hard problem, CLSP with sequence-dependent setup is also NP-hard. Gupta and Magnusson (2005) used this approach for modelling simultaneous lot-sizing and scheduling. Almada-Lobo et al. (2008) demonstrated that Gupta's model does not avoid disconnected sub-tours. Almada-Lobo et al. (2007) presented a model more efficient than those based on Meyr's (2000) approach for the scheduling problem. Almada-Lobo and James (2010) solved the single-machine CLSD (CLSD-SM) by integrating a big-bucket lot-sizing and scheduling model and a batching scheduling model to develop two neighborhood-based search algorithms. James and Almada-Lobo (2011) studied the parallel-machine capacitated lot-sizing and scheduling problem with sequence-dependent setups (CLSD-PM) of which the single-machine problem (CLSD-SM) is a special case. Sereshti and Bijari (2013) studied profit maximization in simultaneous lot-sizing and scheduling problem (PGLSP) and represented a two mathematical models for it. Luis, Klabjan, and Bernardo (2014) proposed a two-dimensional framework for reviewing and classifying the different modelling approaches adopted for sequencing decisions in lot-sizing and scheduling problems. They reviewed the most relevant models in each class to identify their main features and differences, especially their underlying assumptions. From this study emerged an important contribution which is a new polynomial formulation of the problem using commodity flow based sub-tour elimination constraints. The authors also performed extensive computational experiments to compare the different formulations in terms of their ability to yield quality solutions under running time constraints.

Cheng-Hsiang study analyzes two variants of the discrete lot-sizing and scheduling problem (DLSP): first a bi-objective DLSP in which renewable energy is considered and earliness tardiness and CO_2 emissions are minimized simultaneously; second a DLSP in which renewable energy is considered and earliness tardiness is minimized, subject to a constraint on the CO_2 emissions. He represented non-dominated solutions for the bi-objective DLSP are subsequently derived using the lexicographic weighted Tchebycheff (LWT) method. Mehdizadeh,Hajipour, and Mohammadizadeh, studied a multi-item capacitated lot-sizing problem (MICLSP) with setup times, safety stock deficit costs, demand shortage costs – both backorder and lost sale states – and different manners of production. They develop a bi-objective mathematical programming model with two conflicting objectives. They propose two novel Pareto-based multi-objective meta-heuristic algorithms called multi-objective vibration damping optimization and the non-dominated ranking genetic algorithm. Wang et al., an integrated lot sizing and scheduling problem for a production–distribution environment with arbitrary job volumes and distinct due dates considerations. A mixed integer formulation is proposed for this problem, and then a genetic algorithm is developed to solve it.

The lot-sizing decision is not considered in the general earliness/tardiness problem. Supithak et al., (2010) applied the concept of lot-sizing to the E/T problem and investigated it within the framework of the Discrete Lot-sizing and Scheduling Problem (DLSP) along with the chemical batch scheduling which is employed to construct this model. Mirabi (2011) considered the assumption of tardiness penalty in the model for the simultaneous lot-sizing and scheduling problem and presented a mathematical model for it. The major assumption of his problem is that jobs must be produced exactly once in the planning horizon.

3. Problem definition

The problem is must determine production lot sizes and their schedules to minimize the sum of sequence-dependent setup costs, earliness costs (holding cost) and tardiness costs. In GLSPET, there are n orders waiting to be processed on a machine. Each order has its own due date as well as tardiness and earliness costs. Each order is delivered only once. If the production is completed before or on the due date, delivery will be on the due date. Otherwise, the order will be delivered immediately after its production is completed.

3.1 Assumptions

Major assumptions for problems simultaneous scheduling and lot-sizing with earliness/tardiness penalties (called SLET) are as follows:

(1) Each order contains one product.
(2) The planning horizon is finite and contains T periods.
(3) Setup times and costs are sequence-dependent. The triangular inequality holds between setup times.
(4) Holding cost is considered per unit order for unit time and the tardy cost is calculated for the total order.
(5) Machine is setup for the first order at the beginning of the planning horizon.
(6) The model is in the setup preservation mode, which means that the setup state is preserved over idle periods.
(7) If two orders are placed for one product type, sequence-dependent setup costs and setup times must be set to zero.

This model is an extension of Almada-Lobo et al.'s (2007) model in which product sequencing in each period is modelled like a TSP.

3.2. Mathematical Modelling

3.2.1. Decision variables and parameters

We consider *i* and *j* as orders, which are labelled by 1 to *J*, and *t* denotes time periods ranging from 1 to *T*. The parameters of the SLET model will be as follows:

sc_{ij} : Setup cost for transition from order i to order j
st_{ij} : Setup time for transition from order i to order j
β_i : Tardy cost for total order per unit time
h_i : Holding cost for one unit of order i
a_i : The processing time of one unit of order i
d_i : Due date of order i
o_i : Demand of order i
c_t : Available capacity in each period

The following decision variables are used:

x_{ijt} : Binary variable which is 1 when setup occurs from order i to order j in period t
y_{it} : An auxiliary variable that assigns order i in period t
u_{it} : Equals one if the machine is set up for order i at the beginning of period t
T_i : Delay of order i
I_{it} : The inventory level of order i at the end of period t
w_{it} : Binary variable which is 1 when order i is delivered in period t
q_{it} : Quantity of order i produced in period t
co_i : Delivery time of order i
de_{it} : Quantity of order i delivery in period t

3.2.2. The Model

The model is proposed as follows:

$$\text{Min } z = \sum_{i=1}^{J}\sum_{j=1}^{J}\sum_{t=1}^{T} sc_{ij}x_{ijt} + \sum_{i=1}^{J}\beta_i T_i + \sum_{t=1}^{T}\sum_{i=1}^{J} h_i I_{it} \quad (1)$$

S .T.

$$\sum_{i=1}^{J} a_i q_{it} + \sum_{i=1}^{J}\sum_{j=1}^{J} st_{ij}x_{ijt} \le c_t, \quad \forall t = 1, \dots, T \quad (2)$$

$$q_{jt} \le \frac{c_t}{a_j}\left(\sum_{\substack{i=1 \\ i\neq j}}^{J} x_{ijt} + u_{jt}\right), \quad \forall j = 1, \dots, J \quad \forall t = 1, \dots, T \quad (3)$$

$$\sum_{i=1}^{J} x_{ijt} + u_{jt} = u_{j\,(t+1)} + \sum_{i=1}^{J} x_{jit}\,, \quad \forall j = 1, \dots, J \quad \forall t = 1, \dots, T \tag{4}$$

$$y_{it} + Jx_{ijt} - (J-1) - Ju_{i(t)} \leq y_{jt}\,, \quad \forall j,i = 1, \dots, J \;\; i \neq j \quad \forall t = 1, \dots, T \tag{5}$$

$$\sum_{i=1}^{J} u_{it} = 1\,, \quad \forall t = 1, \dots, T \tag{6}$$

$$T_i = co_i - d_i, \quad \forall i = 1, \dots, J \tag{7}$$

$$I_{it} = I_{it-1} + q_{it} - de_{it}, \quad \forall i = 1, \dots, J \quad \forall t = 1, \dots, T \tag{8}$$

$$de_{it} = w_{it} \times o_i, \quad \forall i = 1, \dots, J \quad \forall t = 1, \dots, T \tag{9}$$

$$o_i \times w_{it} \leq \sum_{t'=1}^{t} q_{it}\,, \quad \forall i = 1, \dots, J \quad \forall t = d_i, \dots, T \tag{10}$$

$$w_{it} = 0, \quad \forall i = 1, \dots, J \quad \forall t = 1, \dots, d_i - 1 \tag{11}$$

$$co_i = \sum_{t=1}^{T} w_{it}.t\,, \quad \forall i = 1, \dots, J \tag{12}$$

$$q_{it}, de_{it}, co_i, I_{it}, T_i, u_{it}, y_{it} \geq 0\,, \quad \forall i = 1, \dots, J \quad \forall t = 1, \dots, T \tag{13}$$

$$w_{it}, x_{ijt} \in \{0,1\}, \quad \forall j,i = 1, \dots, J \quad \forall t = 1, \dots, T \tag{14}$$

In this model, Equation (1) represents the objective function which minimizes the sum of setup, tardiness, and inventory holding costs. Inequality (2) ensures that production and setups do not exceed the available capacity. Constraint (3) guarantees that an order is produced if the machine has been set up for it whilst Constraint (4) guarantees that the setup state network is a connected network, and Constraint (5) avoids disconnected sub-tours. Constraint (6) ensures that at the beginning of each period, there is only one setup state. In Equation (7), the amount of delay is calculated for each order. Constraint (8) balances production and inventories with demand. Relation (9) is a definition of de_{it}. Constraint (10) guarantees that an order will not be delivered unless its production is complete for delivery on the due date or after that; thus, it is delivered when the whole order is produced. Constraint (11) ensures that orders are not delivered before their due dates, even their production is complete. Equation (12) calculates the delivery time for each order. Finally, Equations (13) and (14) define the different kinds of variables in the model.

3.2.3. Complexity

The SLET model is a combination of GLSP and the earliness/tardiness scheduling problems. Since the GLSP model is Np-hard (Fleischmann and Meyr, 1997), also according to the research done by Hall, Kubiak, and Sethi (1991), single-machine problem with Earliness/Tardiness criterion is also NP-hard. Proposed problem as a combination of two Np-hard models with no new assumption, is also Np-hard. This justify to developing a meta-heuristic. Two meta-heuristics, namely the Simulated Annealing and the Ant Colony System algorithms, are proposed.

4. Lower bounding relaxation

As seen from the previous subsection, SLET model comprises a large numbers of constraints and variables. It is obvious to achieve optimal solution for model is time consuming and in the real-world situation is impossible. Hence developing meta-heuristic is unavoidable. It is also necessary to develop a lower bound to test the accuracy of the heuristic. To find lower bound we use one type of relaxation and eliminates constraint (14) (constraint relaxation). This elimination causes each order is delivered to several pieces in a separate period after the due date, if the production isn't completed before or on the due date. Therefore holding cost after the due date is zero. These eliminations make the solution be less than

the optimal but the differences are not significant.

5. Meta-heuristic approach

5.1. Ant Colony System Algorithm

This section describes the ACS algorithm. The algorithm contains two phases. In the first phase, periods are selected in a non-ascending order of due dates and the orders are then selected by Equation (15). Equation (15) is used to determine the probability that an order is selected for production in a certain period, in which q is a random variable uniformly distributed over [0,1], q_0 ($0 \leq q_0 \leq 1$) is a parameter, and i is a random variable selected according to the probability distribution given by Equation (16), where η_{it} is a heuristic value calculated by Equation (17). In this Equation, γ and γ' are parameters, β is the parameter to control the influence of the heuristic information, and τ_{ti} is the rate of pheromone if the order i is produced in period t. Also, N^k is the set of candidate orders (incomplete orders) of the ant k.

$$\mathrm{i} = \begin{cases} \mathrm{argmax}_{\mathrm{l}\in \mathrm{N}_\mathrm{i}^\mathrm{k}}\{\tau_{\mathrm{ti}}[\eta_{\mathrm{ti}}]^\beta\} & \text{if } \mathrm{q} \leq \mathrm{q}_0 \\ \mathrm{I} & \text{otherwise} \end{cases} \tag{15}$$

$$\mathrm{p}_{\mathrm{ti}}^{\mathrm{k}} = \frac{[\tau_{\mathrm{ti}}][\eta_{\mathrm{ti}}]^\beta}{\sum_{\mathrm{l}\in \mathrm{N}_\mathrm{i}^\mathrm{k}}[\tau_{\mathrm{ti}}][\eta_{\mathrm{ti}}]^\beta} \quad \text{if } \mathrm{j} \in \mathrm{N}^{\mathrm{k}} \tag{16}$$

$$\eta_{\mathrm{ti}} = \begin{cases} \gamma \mathrm{d}_\mathrm{i} = \mathrm{t} \\ \gamma' \mathrm{d}_\mathrm{i} \neq \mathrm{t} \end{cases} \tag{17}$$

If all the orders are complete in the first phase, there is no need for the second phase of the algorithm; otherwise, the algorithm starts its second phase, in which, unlike the first phase, orders are selected in a non-ascending order of the total cost (including holding, setup, and tardiness costs) of the best solution, and the periods are then selected. If the order is completed, the next order is then selected.

- Initialization of pheromones

Let the initial pheromone trail be $\tau_0 = \frac{1}{\mathrm{z}_{\mathrm{first}}}$, where $\mathrm{z}_{\mathrm{first}}$ is the objective function value of the feasible solution.

- Local and global updating of pheromones

Global updating of pheromones in ACS is implemented by Equation (18):

$$\tau_{\mathrm{ti}} = (1-\rho)\tau_{\mathrm{ti}} + \rho\Delta\tau_{\mathrm{ti}}^{\mathrm{bs}}, \quad \left(\mathrm{T}^{\mathrm{bs}}\forall(\mathrm{t,i})\middle|\mathrm{q}_{\mathrm{it}} > 0\right) \tag{18}$$

where, $0 < \rho < 1$ is the evaporation rate, $\Delta\tau_{\mathrm{ti}} = {}^{1}/_{\mathrm{z}^{\mathrm{bs}}}$ in which z^{bs} is the objective function value of the best-so-far. T^{bs} represents the sets of periods in which the orders are produced.

In ACS, the pheromone rule is locally updated and immediately applied after each feasible solution. The local updating of pheromone is implemented by Equation (19):

$$\tau_{\mathrm{ti}} = (1-\xi)\tau_{\mathrm{ti}} + \xi\tau_0 \tag{19}$$

where, ξ is a parameter in this equation whose value is between zero and one. τ_{ti} is the rate of the pheromone if order i is produced in period t.

5.2. Simulated Annealing Algorithm

Simulated annealing is a local search algorithm (meta-heuristic) capable of escaping from local optima. Its ease of implementation, convergence properties and its use of hill-climbing moves to escape local optima have made it a popular technique. Generally speaking, a search algorithm in combinatorial problems first generates neighborhood solutions from

a current solution, and then, continues to move to one of them until prespecified conditions are met. SA uses this repetitive improvement approach, but in particular it probabilistically allows deteriorating movements so that a neighborhood solution out of a local optimum can be tried.

The simulated annealing (SA) heuristic we propose is as follows:

GENERATE a feasible schedule X with cost Z

set the initial temperature T
set the temperature length L
set the cooling ratio R
set the maximum iterations number, mrun
set the iterations counter, crun
while crun < mrun do
    do i=1, L
        SEARCH and select a neighbour X' of X
        Let Z' = the cost of X'
        Let $\delta = Z' - Z$
        Let $\delta \leq 0$
            Set X= X'
        If $\delta > 0$
            Set X= X' with probability $e^{-\frac{\delta}{T}}$
    Continue
    Set T=R*T
    read crun
Continue
end

5.2.1. Initial solution

In theory, simulated annealing yields a near optimal solution independent of the initial solution. Although this factor has been tested in past simulated annealing literature, empirical evidence linking the initial solution to the final solution is inconclusive. The procedure for the initial solution runs as follows: Select the orders in a non-decreasing order of due dates. Then, determine production lot sizes on the left hand side of the ordered due date or one closest to it. If there is no period available, insert the lot size of the order in the available period closest to the due date on the right hand side.

5.2.2. Generation of neighborhood solutions

Neighborhood solutions are generated according to the following two methods.

- **Lot interchange**

This method selects and exchanges two lots. The first lot to exchange is selected based on the ratio of tardiness and earliness cost. The other lot to exchange is randomly selected.

- **Lot insert**

One lot is inserted at the end of another lot. Lots are selected as described in the lot interchange section.

5.2.3. Temperature

The temperature begins at a high level and is cooled until an equilibrium is reached. The way in which the algorithm avoids being trapped in a local minimum is that it generates and accepts random solutions in which the performance evaluation function has a greater value, i.e., the solution has a higher energy. When the temperature is high, the algorithm will be likely to accept a higher energy solution, while at a very low temperature the algorithm will almost only accept solutions of lower energy. However, it is necessary to control initial temperature more efficiently because very high initial temperature could consume too much time in preparation stage of our problems. We used objective function from initial solution in this paper.

5.2.4. Cooling ratio

We use common ratio such as the geometric ratio, which use the temperature of *k*th exterior loop T_k: $T_k = \rho\ T_{k\_1}$;

5.3. Sequencing orders within periods and Termination criteria

To sequence orders within period *t*, one directed graph is extracted at first. Hence one directed graph (called *D*) is constructed by double (*N*, *E*), where *N* is the set of nodes of *D*. Each node indicates one order. *E* is the subset of $\{ij \mid i,j \in V \text{ and } i \neq j\}$ called the set of edges of *D*. Each edge indicates feasible sequence between two orders. Therefore if order *i* cannot be processed immediately after order *j*, there is no edge from node *j* to node *i*.

To constructing *D*, one additional node numbered *0* is shown in *D*. Each edge *0i* ($i \in N$) in *D* indicates sequence *0i* and it means order *i* is the first order in sequence within period *t* (based on assumption (6)). If node *i* in *D* has various exiting edges and one of these edges ends to one degree node (node with only one entering edge), say node *j*, then it means order *j*, and no other orders, must be processed on the machine after order *i*. Therefore all ik arcs, $k \neq j$, indicate infeasible sequences. Obviously edge *ji* (if exists) represent infeasible sequence because if order *j* must be processed after order *i*, order *i* cannot be processed after order *j*. So all of redundant edges must be eliminated from *D*. Select sequence with the least summation setup costs. We set the termination criterion as N_{Al} iterations without an improvement.

5.4. Local search procedure

Local search is used in the two algorithms to improve the feasible solutions obtained. The local search contains the following three sections:

(1) In order to reduce the tardiness cost, the orders placed in periods after their due dates are moved to periods before their due dates.
(2) For any order, the holding cost is reduced by moving the lot sizes of periods other than their delivery periods to their delivery periods or to a neighbouring one.
(3) The setup cost is reduced by changing the sequence of orders inside periods using the pair-wise exchange.

6. Numerical experiments

In this section, we report the numerical experiments carried out for the SA and ACS algorithms.

6.1. Problem instance generation

The method proposed by James and Almada-Lobo (2011) is used to generate data sets for SLET with some modifications as needed.

The method runs as follows. The production capacity in each period is calculated as in Equation (20), where C_t is the capacity available in period t, O_i is the demand of order *i*, and the value of *S* is calculated using Equation (21). In Equation (20), the parameter ρ adjusts the capacity. Based on our investigations, the values 0.05, 0.25, 0.5, 0.75, and 1 were considered for this parameter.

$$C_t = \frac{\sum_{i=1}^{J} o_i}{T} \text{Rand}[1,1.2] + \rho \times S \tag{20}$$

$$S = \sum_{i=1}^{J} \max_j (st_{ij}) - \min_i \{\max_j (st_{ij})\} \tag{21}$$

The setup times (st_{ij}) have a discrete uniform distribution over the range [5,10]. Setup costs are generated using Equation (22). In this equation, θ is a coefficient for adjusting setup costs. The value for this parameter will be equal to 50 or 100 for different scenarios, holding costs are between 2 and 9 penalty units per period, processing time for one unit is equal to one unit of time, and demand is between 40 and 59 units per period (James and Almada-Lobo, 2011).

$$sc_{ij} = st_{ij} \times \theta \tag{22}$$

The tardiness cost for each order is calculated using Equation (23), in which α is a coefficient for adjusting the tardiness cost. The values for this parameter will be equal to 1.2, 1.5, and 2 for different scenarios.

$$\beta_i = h_i \times o_i \times \alpha \tag{23}$$

A total number of 30 (2×3×5) problem sets are generated based on the different coefficients considered for the parameters α, ρ, and, θ. In order to investigate the efficiency of the proposed solution methods, 12 problem groups with different

sizes were generated for each of the 30 sets. The row titled 'Small' in Table 1 shows the numbers of orders and periods. In other words, to test the performance of the algorithms, 30 problem sets each with 12 problems were generated, making a total number of problems 360 (12×30) problem instances.

6.2. Parameter setting for the two algorithms

In order to set the parameters in the two algorithms, several values are considered for each parameter at as shown in Table 2. The impacts of these parameters and their interactions on every algorithm are investigated using the full factorial design with 24 blocks selected from different problems. Tukey's (1977) multiple range test is used to compare all pairs of means for each factor to identify the best value for each parameter.

6.2.1. Parameter setting for the Ant Colony System algorithm

Initial tests revealed that a k value of 10 proposed by Dorigo and Stutzle (2004) would be the best for the number of ants. Also, the values of 1 and 1.8 suggested in Almeder (2010) were considered for γ and γ' parameters, respectively. Bold numbers in Table 2 show the values selected for the parameters.

6.3. Numerical results

In this section, we investigate the quality of the proposed meta heuristics through comparisons. This evaluation is accomplished in separated parts using small-scale and large sized instances. To solve the proposed mathematical model, the GAMS 23/4 software and CPLEX solver was used. The algorithms were coded in Visual C#.NET 2010 and solved on a computer with a 2.67 GHZ processor. The models were run in a time limit of 3600 s.

The problems without optimal solutions within this time limit were omitted. Some of the problems presented in sets 9, 10, 11, and 12 with ρ=0.05 had no optimal solutions. Computed lower bound (LB) was applied for all instances to show their accuracy compared with the mathematical model in small-size problems. To evaluate the performance of the algorithms, the solution for each problem obtained from each algorithm was compared with that obtained from the other model. To do this, each algorithm was run five times for each problem. The average percentage deviations of the five runs were reported as the percentage deviation from each algorithm for each problem. Then, the average percentage deviation was calculated for the 30 sets and presented as percentage deviation for each set. With regard to time limit, the number of problems with optimal solutions in each problem set was also presented. The results of comparisons are presented in Table 3. Clearly, optimal solutions were obtained for 333 out of the 360 problems within the limit of 3600 seconds. Table 3 also shows the percentage deviation for each problem set. It is also clear from this Table that the models exhibited differences in their performance for different problem sets. In other words, none of the algorithms could be considered the best.

Hence, the differences between two methods must be analyzed whether they are significant or not. To do this, hypothesis $\mu_1 - \mu_2 = 0$ against the hypothesis $\mu_1 - \mu_2 > 0$ with unknown and unequal variance was used. It is needed to perform a t test analysis. The Sample size is equal to 12, sample average for ASC and SA are $\bar{X}_1 = 1.77$ and $\bar{X}_2 = 1.72$ and sample standard deviation for ASC and SA are $s_1 = 0.68$ and$s_2 = 0.68$, respectively. Since $t = 0.58 < t_{0.05,24}$=1.711, it means that the difference isn't statistically significant.

In the case of solution times, similar to the case of percentage deviation, the average value of the solution times of the five runs was reported as the average solution time (Table).Clearly, the average solution time of the SA algorithm for all the problem sets, except for sets 1, is less than that of ACS. Also, the average solution time for all the sets by SA is equal to nearly one fourth of that by ACS.

To study the effects of different parameters on the performance of the algorithms, average percentage deviations were calculated for the different input parameters. For this purpose, the problems with identical parameters were grouped into one set and the average percentage deviation was calculated for each new set. The column titled 'Small-scale Instances' in Table shows the effect of coefficientsα, ρ, and θ on the performance of the algorithms. It is clear from the first part of Table that when α is equal to 1.2, the average percentage deviation is greater than the other two values for both algorithms. The second part of the Table shows the effect of θ. Clearly, the average percentage deviation of instances with θ=50 is lower than that of instances with θ=100. This indicates that problems with higher setup costs are more difficult to solve. The third part of Table shows the effect of ρ. When this coefficient is equal to 0.05, we have the tightest capacity. Generally speaking, differences are observed between the two algorithms with regard to their different parameters.

In order to compare the performances of the algorithms for large scale instances, 9 problem groups with different sizes were generated for all the 30 sets. The row titled 'Large' in Table 1shows the numbers of orders and periods. Similar to

the previous case, each problem was run five times using each algorithm. The percentage deviation value of each problem for each run was calculated as follows:

$$\text{RPD} = \frac{\text{F}_{\text{Algorithm}} - \text{LB}}{\text{LB}} \times 100 \quad (24)$$

where, LB is lower bound and $F_{Algorithm}$is the objective function value of the algorithms. Average percentage deviations and average solution times are calculated as they were for the small scale instances (Table 7). Clearly, the algorithms exhibit the different performance with respect to the different groups of problems. As in the case of small scale instances, the average solution time for the SA algorithm is less than that for the ACS in all the problem sets.

To study the effects of different parameters on the performance of the algorithms, average percentage deviation was calculated for each input parameter. The column titled 'Large Scale' in Table presents the effects of α, ρ, and θ on the performance of the algorithms.

7. Conclusions and future research

The problem investigated in this paper is one of simultaneous lot-sizing and scheduling with earliness/tardiness penalties (SLET) along with the sequence-dependent setup time and cost. The problem objective is to determine the production lot-sizes and their schedules in order to minimize the sum of the total setup, holding, and tardiness cost. A mathematical model, and two meta-heuristics, namely SA and ACS, were presented as solution methods for the problem. Also, lower bounds are obtained from solving the problem relaxation. A benchmark was established to evaluate the performance of the algorithms. The algorithms were initially evaluated for their efficiencies in obtaining optimal solutions for the set of small size instances by solving the mathematical model within a time limit of 3600 seconds. The algorithms were also compared, with lower bound, for their performance with a set of large size instances. The effects of different parameters on the performance of the algorithms were evaluated for both small and large size problem instances. As shown by the experimental results, our proposed SA algorithm outperformed the other meta-heuristics algorithm. Also the solution time of the SA algorithm was less than that of ACS.

For future research, the proposed model can be extended to the flow shop environment, assembly line. Also, considering other functions used in the sequencing and scheduling such as maximum completion time in the flow shop, will be an interesting area for future research. Other meta-heuristic methods can be developed to solve the problem**.** Finally, order selection and order acceptance or rejection are suggested as topics for future research.

Representing exact methods which try to find an optimal solution based on the model structure like branch and bound algorithm is a good aspect for future research. Efficient heuristic and met- heuristic algorithms are also appropriate methods to solve SLET.

References

Almada-Lobo, Bernardo, and Ross J. W. James. (2010). Neighbourhood search meta-heuristics for capacitated lot-sizing with sequence-dependent setups. *International Journal of Production Research*. Vol. 48 (3), pp. 861-878.

Almada-Lobo, Bernardo, Diego Klabjan, Maria Antóniacarravilla, and José F. Oliveira. (2007). Single machine multi-product capacitated lot-sizing with sequence-dependent setups. *International Journal of Production Research*. Vol. 45 (20), pp.4873-4894.

Almada-Lobo, Bernardo, José F. Oliveira, and Maria Antóniacarravilla. (2008). A note on " the capacitated lot-sizing and scheduling problem with sequence-dependent setup costs and setup times. *Computers & Operations Research*, Vol. 35 (4), pp. 1374-1376.

Almeder, Christian. (2010). A hybrid optimization approach for multi-level capacitated lot-sizing problems. *European Journal of Operational Research*, Vol. 200 (2), pp. 599-606.

Clark, Alistair R., and Simon J. Clark. (2000). Rolling-horizon lot-sizing when set-up times are sequence-dependent. *International Journal of Production Research*, Vol. 38 (10), pp. 2287-2307.

Dorigo, Marco, and Thomas Stutzle. (2004). *Ant colony Optimization*: MIT Press,Cambridge.

Fleischmann, Bernhard, and Herbert Meyr. (1997). the general lot-sizing and scheduling problem.*OR Spectrum*, Vol. 19, pp. 11-21.

Gupta, Diwakar, and Thorkell Magnusson. (2005). The capacitated lot-sizing and scheduling problem with sequence-dependent setup costs and setup times. *Computers & Operations Research*, Vol. 32 (4), pp. 727-747.

Hall, Nicholas G., WieslawKubiak, and Suresh P. Sethi. (1991). Earliness-tardiness scheduling problems, II: Deviation of completion times about a restrictive common due date. *Operations Research*, Vol. 39 (5), pp. 847-856.

James, Ross J. W., and Bernardo Almada-Lobo. (2011). Single and parallel machine capacitated lot-sizing and scheduling: New iterative MIP-based neighborhood search heuristics. *Computers & Operations Research*, Vol. 38 (12), pp.1816-1825.

Luis, Guimarães, Diego Klabjan, and Almada-Lobo Bernardo. (2014). Modeling lotsizing and scheduling problems with sequence dependent setups. *European Journal of Operational Research*, pp. 644-662. (DOI: 10.1016/j.ejor.2014.05.018)

Liu, Cheng-Hsiang. (2016). Discrete lot-sizing and scheduling problems considering renewable energy and CO2 emissions. *Production Engineering*, Vol. 10(6), pp. 607-614.

Mahdieh, Masoumeh, Mehdi Bijari, and Alistair Clark. (2011). Simultaneous Lot-Sizing and Scheduling in a Flexible Flow Line. *Journal of Industrial and Systems Engineering*, Vol. 5 (2), pp.107-119.

Mehdizadeh, E., Hajipour, V. and Mohammadizadeh, M. R. (2015). A bi-objective multi-item capacitated lot-sizing model: Two Pareto-based meta-heuristic algorithms. *International Journal of Management Science and Engineering Management*, pp. 1-15. (DOI: 10.1080/17509653.2015.1086965)

Mirabi, Mohammad. (2011). A hybrid simulated annealing for the single-machine capacitated lot-sizing and scheduling problem with sequence-dependent setup times and costs and dynamic release of jobs. *The International Journal of Advanced Manufacturing Technology*, Vol. 54 (9-12), pp. 1109-1119.

Mohammadi, Mohammad. (2010). Integrating lot-sizing, loading, and scheduling decisions in flexible flow shops. *The International Journal of Advanced Manufacturing Technology*, Vol. 50 (9-12), pp.1165-1174.

Mohammadi, Mohammad, and NiloofarJafari. (2011). A new mathematical model for integrating lot-sizing, loading, and scheduling decisions in flexible flow shops. *The International Journal of Advanced Manufacturing Technology*, Vol. 55 (5-8), pp.709-721.

Monden, Yasuhiro. (1998). *Toyota production System: An Integrated Approach to Just-In-Time*. Fourth ed. Norcross: Productivity Press.

Seeanner, F., Bernardo Almada-Lobo, and H. Meyr. (2013). Combining the principles of variable neighbourhood decomposition search and the fix & optimize heuristic to solve multi-level lot-sizing and scheduling problems. *Computers & Operations Research*, Vol. 40 (1), pp. 303-317.

Seeanner, Florian, and Herbert Meyr. (2013). Multi-stage simultaneous lot-sizing and scheduling for flow line production. *OR Spectrum*, Vol. 35 (1), pp. 33-73.

Sereshti, Narges, and Mehdi Bijari. (2013). Profit maximization in simultaneous lot-sizing and scheduling problem." *Applied Mathematical Modelling*, Vol. 37 (23), pp. 9516-9523.

Supithak, Wisut, Surya D. Liman, and Elliot J. Montes. (2010). Lot-sizing and scheduling problem with earliness tardiness and setup penalties. *Computers & Industrial Engineering*, Vol. 58 (3), pp. 363-372.

Tukey, John Wilder. (1977). *Exploratory Data Analysis*: Addison-Wesley,Boston.

Wang, D., Grunder, O. and Moudni, A. E. (2014). Using genetic algorithm for lot sizing and scheduling problem with arbitrary job volumes and distinct job due date considerations. *International Journal of Systems Science*, Vol. 45(8), pp. 1694-1707.

Appendix

Table 1. Numbers of orders and periods

Scale Problem	problem group	T×J	Problem group	T×J	Problem group	T×J
Small	1	5×5	5	10×10	9	15×15
	2	5×10	6	10×15	10	15×17
	3	10×5	7	15×10	11	15×20
	4	10×7	8	15×12	12	17×15
Large	1	20×15	4	25×20	7	30×25
	2	20×20	5	25×25	8	30×30
	3	20×25	6	25×30	9	30×35

Table 2. Values suggested for the parameters of the SA and ACS algorithms

Algorithm	Parameter	Description	Values suggested
SA	N_{Al}	stopping criteria	30 40 **50**
	ρ	Cooling ratio coefficient	0. 5 0. 7 **0. 9** 0. 95
	L	temperature length	10 20 **30** 40
ACS	N_{Al}	stopping criteria	30 **40** 50
	ψ	The rate of influence of the heuristic information	**3** 4 5
	ξ	rate of local evaporation	0.04 **0.07** 0.1
	φ	rate of global evaporation	0.1 0.2 **0.3**
	q_o	A parameter that chooses a value between zero and one	**0.4** 0.5 0.6 0.7

Table 3. Average percentage deviations of the solutions and the number of problems with optimal solutions obtained by the algorithms and lower bound

Problem groups	T×J	Number of problems with optimal solutions				Average percentage deviation		
		Mathematical model	LB	SA	ACS	SA	ACS	LB
1	5×5	30	24	27	25	0/15	0/23	0.13
2	5×10	30	14	16	19	2/16	2/63	3.01
3	10×5	30	15	19	12	1/55	1/67	1.89

Table 3. Continued

Problem groups	T×J	Number of problems with optimal solutions				Average percentage deviation		
		Mathematical model	LB	SA	ACS	SA	ACS	LB
4	10×7	30	15	12	15	2/04	1/83	2.43
5	10×10	30	22	20	21	1/59	1/41	1.33
6	10×15	30	25	24	21	1/08	0/82	1.23
7	15×10	30	12	16	14	1/59	1/79	1.02
8	15×12	29	12	12	16	2/43	1/85	2.45
9	15×15	24	10	13	16	2/05	2/31	2.46
10	15×17	24	13	17	17	1/62	2/14	2.78
11	15×20	24	14	19	14	1/53	2/18	2.36
12	17×15	22	8	12	6	2/85	2/32	3.32
Total number		333	184	207	196	-	-	-
Average		-	*0.55*	0/62	0/59	1/72	1/76	2.03

Table 4. Average solution time of the algorithms and optimal solution time (seconds)

Problem groups	T×J	Optimal time	LB	SA	ACS
1	5×5	0/61	0/51	1/01	0/31
2	5×10	1/08	1/03	0/98	1/61
3	10×5	1/80	1/14	1/11	1/49
4	10×7	13/37	10/57	1/45	7/54
5	10×10	8/64	7/54	3/04	4/06
6	10×15	60/48	54/47	4/12	14/05
7	15×10	80/09	65/08	1/54	8/89
8	15×12	332/64	321/45	4/02	23/22
9	15×15	115/31	112/05	2/12	10/40
10	15×17	381/89	289/08	5/08	19/13
11	15×20	248/33	203/23	7/25	22/97
12	17×15	714/03	552/12	13/14	42/38
Average solution time		216/17	147/07	3/74	13

Table 5. Coefficient effects on the performance of the algorithms

Parameter	Value	Small scale		Large scale	
		SA	**ACS**	**SA**	**ACS**
α	½	2/12	2/16	0/63	0/49
	1/5	1/89	1/65	0/54	0/45
	2	1/13	1/50	0/50	0/40
θ	50	1/12	1/29	0/48	0/45
	100	2/35	2/25	0/65	0/46
ρ	0/05	2/89	2/62	1/05	1/37
	0/25	1/98	2/25	0/62	0/78
	0/5	1/38	1/61	0/43	0/47
	0/75	1/43	1/40	0/32	0/43
	1	1/00	1/07	0/29	0/22

Table 6. Average percentage deviation and average solution time in large scale instances

Group problem	Average percentage deviation		Average solution time (second)	
	SA	**ACS**	**SA**	**ACS**
1	0/41	0/75	7/05	115/98
2	0/45	0/83	40/82	139/96
3	0/83	0/60	45/04	193/54
4	0/69	0/72	132/05	337/55
5	1/04	1/36	74/02	181/60
6	0/41	0/80	78/45	377/39
7	1/19	0/77	101/78	344/60
8	0/98	0/75	65/45	438/00
9	1/03	1/65	191/81	956/24
Total average	0/78	0/91	81/83	342/76