applied sciences

MDPI

Article

Lagrange Relaxation for the Capacitated Multi-Item Lot-Sizing Problem

Zhen Gao [1,2,*], Danning Li [1,2], Danni Wang [1,2] and Zengcai Yu [1,2]

[1] National Frontiers Science Center for Industrial Intelligence and Systems Optimization, Northeastern University, Shenyang 110819, China; danning1019@163.com (D.L.); danni_present@foxmail.com (D.W.); yu_zengcai@163.com (Z.Y.)
[2] Key Laboratory of Data Analytics and Optimization for Smart Industry (Northeastern University), Ministry of Education, Shenyang 110819, China
* Correspondence: gaozhen@ise.neu.edu.cn

Abstract: The capacitated multi-item lot-sizing problem, referred to as the CLSP, is to determine the lot sizes of products in each period in a given planning horizon of finite periods, meeting the product demands and resource limits in each period, and to minimize the total cost, consisting of the production, inventory holding, and setup costs. CLSPs are often encountered in industry production settings and they are considered NP-hard. In this paper, we propose a Lagrange relaxation (LR) approach for their solution. This approach relaxes the capacity constraints to the objective function and thus decomposes the CLSP into several uncapacitated single-item problems, each of which can be easily solved by dynamic programming. Feasible solutions are achieved by solving the resulting transportation problems and a fixup heuristic. The Lagrange multipliers in the relaxed problem are updated by using subgradient optimization. The experimental results show that the LR approach explores high-quality solutions and has better applicability compared with other commonly used solution approaches in the literature.

Keywords: Lagrange relaxation; lot-sizing; CLSP; subgradient optimization

Citation: Gao, Z.; Li, D.; Wang, D.; Yu, Z. Lagrange Relaxation for the Capacitated Multi-Item Lot-Sizing Problem. *Appl. Sci.* **2024**, *14*, 6517. https://doi.org/10.3390/app14156517

Academic Editors: Jose Machado and Paolo Renna

Received: 11 June 2024
Revised: 12 July 2024
Accepted: 23 July 2024
Published: 25 July 2024

1. Introduction

The capacitated multi-item lot-sizing problem, referred to as the CLSP, is often encountered in industry production settings, where multiple products are produced and a resource bottleneck exists [1–3]. The problem is to determine the production lot size of each product and its timing over a finite number of periods, so that the total cost, consisting of the production costs, setup costs and inventory costs, is minimal and the production demand of each product in each period is met. Mathematically, the CLSP can be formulated as follows.

(CLSP)

$$\text{minimize } z = \sum_{t=1}^{T}\sum_{i=1}^{I}(p_{it}x_{it} + h_{it}I_{it} + s_{it}y_{it}) \tag{1}$$

s.t.

$$I_{it-1} + x_{it} - I_{it} = d_{it}, i \in I, t \in T \tag{2}$$

$$\sum_{i\in I} a_i x_{it} \le C_t, t \in T \tag{3}$$

$$x_{it} \le My_{it}, i \in I, t \in T \tag{4}$$

$$x_{it,} I_{it} \ge 0, i \in I, t \in T \tag{5}$$

$$I_{i0}, I_{iT} = 0, i \in I \tag{6}$$

$$y_{it} \in \{0,1\}, i \in I, t \in T \tag{7}$$

where the meanings of the notations are as follows.

Indices and sets:

i—product;
t—period;
I—set of products;
T—set of periods.

Parameters:

p_{it}—unit linear production cost for product i in period t;
h_{it}—unit inventory holding cost for one unit of product i between periods t and $t + 1$;
s_{it}—setup cost for product i in period t;
d_{it}—demand for product i in period t;
a_i—unit capacity requirement for product i;
C_t—available capacity in period t;
M—a large positive number.

Variables:

x_{it}—amount of product i produced in period t;
I_{it}—inventory of product i at the end of period t;
y_{it}—binary variable that = 1 if $x_{it} > 0$ and = 0 if $x_{it} = 0$.

In the CLSP model, the objective function (1) minimizes the total production, inventory holding and setup costs. Constraint (2) represents the inventory balance; for any product i, the production in period t plus the remaining inventory at the end of period $t - 1$ minus the demand in period t is equal to the inventory at the end of period t. Constraint (3) is the capacity constraint for the resource bottleneck, which means that the capacity consumption of all products in each period t cannot exceed the available amount in that period. Constraint (4) is the associated relation of variables x and y, where a product i is produced in a period t if and only if it has been set up in that period. Constraints (5)–(7) define the decision variables. Note that inventories are non-negative, meaning that no backlogs are allowed. Note also that when the production costs do not change over time, the first cost in the objective function is constant and can be removed.

CLSPs are the core of production planning and scheduling. In the enterprise production environment, the production capacity of the key equipment is often limited, and many products must share the key equipment resource, such as a key machine tool or a processing center—for example, in the manufacturing industry, these include tires, bearings, chemical products and pharmaceuticals. In these industries, the organization of production is often in the form of an MRP; in this process, the main production schedule (MPS) is devised according to the bill of materials (BOM), used to calculate the material demand. Often, the product demand is beyond the key equipment capacity in many periods; in addition, the key equipment capacity is not fully utilized in many periods, resulting in stock shortages and cost increases. Therefore, it is important to consider the production capacity limit in the process of production planning. The CLSP is useful to solve this kind of problem and its main aim is to determine the lot sizes. However, the CLSP problem is NP-hard [4]; thus, the search for an effective solution to the CLSP has been of interest in both industry [5,6] and academic communities [7–9]. Many publications on the subject of "lot-sizing or lot sizes" can be found [10,11] in the manufacturing environment. The most recent studies include Daryna and Christian [12], who propose a two-step construct heuristic for the CLSP. The algorithm firstly sorts the customer orders and then iteratively adds them to a preliminary production plan in the second step. The algorithm can solve various variants of the CLSP easily and rapidly, and the performance surpasses that of the well-known Dixon and Silver heuristic [13] and the ABC heuristic [14]. Seiringer et al. [15] apply a simheuristic to minimize the overall costs of an MRP production system, where an important issue is the lot-sizing problem. This method combines different simulation and optimization schemes for a planned MRP system to determine the optimal parameters, like the lot size, safety stock and lead time. Cyril et al. [16] consider a lot-sizing and scheduling problem in the tire

industry. This problem includes 170 products, 70 machines and 42 periods. A mixed integer programming formulation is built and a problem-based metaheuristic is developed to solve it. Gurkan and Tunc [17] use a fix-and-optimize heuristic for the capacitated multi-item stochastic lot-sizing problem. Different item and period sets are partitioned to become many subproblems and they set binary variables to be optimized.

2. Literature Review

The study of lot-sizing problems can be traced back to the EOQ problem [18], i.e., the economic ordering quantity model. The EOQ is essentially a trade-off among the ordering/setup cost and inventory holding cost, seeking to achieve the optimal economic lot sizes. The EOQ supposes that the problem is for a single stationary product, an invariable demand and an unlimited production capacity. The EOQ can be solved with differential methods. While some complex characteristics are combined in the "lot sizes" problem, this problem is very difficult to solve. For example, when the product demand changes over time, more products are produced. However, one of the most essential changes for problem-solving is the existence of production capacity constraints. Consequently, the complexity of a lot-sizing problem can be characterized in terms of the length of the horizon, number of items and capacity constraints. Based on these characteristics, lot-sizing problems can be classified into the uncapacitated single-item lot-sizing problem (SILSP), the capacitated single-item lot-sizing problem (CSILSP), the uncapacitated multi-item lot-sizing problem (ULSP) and the capacitated single-level multi-item lot-sizing problem (CLSP). Among them, the CLSP is the most common and difficult problem and has become the core of lot-sizing problem research. Meanwhile, the SILSP is the simplest lot sizing problem, specific to a single product, with a finite planning horizon of T periods and no capacity constraints. SILSPs are solvable in $O(T^2)$, except for some variants with some essential extensions, such as the setup time and (or) cost, backlogs, time windows, etc., which are NP-hard. ULSPs are an extension of SILSPs with multiple items and thus are also solvable in $O(T^2)$. CSILSPs are an extension of the SILSPs with capacity constraints and are NP-hard [4]. The CLSP is the extension of the CSILSP with multiple items and is thus NP-hard. Because of the difficulties of the CLSP, most of the existing solution approaches are heuristic and the exact solution approaches are only suitable for small-sized problems. In general, the solution approaches to the CLSP can be divided into three classes: common heuristics, mathematic programming-based and meta-heuristics approaches.

Heuristics is sometimes the only available choice when the capacity constraints are too tight in a CLSP. The well-known common heuristics include lot-for-lot, least unit cost (LUC) [19], part-period balancing (PPB) [20,21], Silver–Meal (SM) [22], Lambrecht–Vanderveken (LV) [23] and Dixon–Silver (DS) [13]. The lot-for-lot heuristic directly converts the demand matrix into a lot-size matrix to obtain an initial solution. This approach maximizes the setup costs and minimizes the inventory holding costs. It is suitable for problems with looser capacity constraints and can quickly obtain an initial feasible solution. The LUC heuristic tracks the lowest cost per unit for production, and it can be expressed mathematically as

$$\underset{t}{\operatorname{argmin}}\, k(t) = \frac{s + h\sum_{\tau=1}^{t}(\tau-1)d_\tau}{\sum_{\tau=1}^{t} d_\tau}, t = 1, \ldots, T \tag{8}$$

Hereafter, s indicates the setup cost and h is the unit inventory holding cost. The PPB heuristic aims at seeking the maximum production over the demand of several periods when the total holding cost does not exceed a single setup cost. It is expressed by the formula

$$\underset{t}{\operatorname{argmax}}\, h\sum_{\tau=1}^{t}(\tau-1)d_\tau - s \le 0, t = 1, \ldots, T \tag{9}$$

The SM heuristic attempts to minimize the average cost per unit time for each lot. Mathematically, it can be expressed as

$$\underset{t}{\operatorname{argmin}}\, k(t) = \frac{s + h\sum_{\tau=1}^{t}(\tau-1)d_\tau}{t}, t = 1, \ldots, T \tag{10}$$

The LV heuristic is an extended Eisenhut heuristic [24]. LV's idea is analogous to that of Eisenhut's heuristic, a period-by-period heuristic. Because we always suppose that the current period is 1, the LV then computes the maximum order up to the period

$$\min_{t} B_t = \sum_{\tau=2}^{t}\left(\sum_{i=1}^{I} d_{i\tau} - C_\tau\right) > 0,\ t = 2, \ldots, T \tag{11}$$

Suppose that $B_{t^*} > 0$; this means that, from period 2 to period t^*, the cumulative demands exceed the cumulative capacity by B_{t^*} units, which must be backward-shifted to period 1. We repeat the above process until the current period is T and a feasible solution is obtained. The DS heuristic can be regarded as an extension of the SM heuristic for multiple items, because the single-item SM heuristic for the multiple-item lot-sizing problems is not necessarily optimal or even feasible if capacity limitations exist. Consequently, for a multiple-item problem, the DS would seek to increase the items, which results in the largest decrease in the average costs per unit time per unit capacity.

$$\underset{u_i}{\operatorname{argmax}}\ \ u_i = \frac{AC_i(T_i) - AC_i(T_i+1)}{d_{i,T_i+1}} > 0,\ i = 1, \ldots, I \tag{12}$$

where, $AC_i(T_i) = (s_i + h_i\sum_{\tau=1}^{T_i}(\tau-1)d_{i\tau})/T_i, T_i = 1, \ldots, T$ denotes the average cost per unit time of a lot of item i that will satisfy T_i periods' requirements. Therefore, u_i indicates the marginal decrease in the average costs per unit of capacity absorbed when increasing the production of item i from T_i to $T_i + 1$.

Mathematic programming-based approaches are a broad class of methods, suited for lot-sizing problems with complex constraints and multiple items. The well-known W-W [25] dynamic programming for the uncapacitated single-item lot-sizing problem is representative. It solves the USLP effectively in time $O(T^2)$. Moreover, Barany et al. [26] proposed a linear programming (LP)-based approach for the CLSP. This approach embeds valid inequalities into a branch-and-bound framework to solve the linear programming relaxation of the CLSP. Up to 20-item × 13-period CLSPs are solved optimally with this approach. Lasdon and Terjung [27] used column generation and a generalized upper bound to solve the CLSP. Up to 393-item × 6-period CLSPs are solved with good approximation. In addition, Bahl [28] proposed a column generation heuristic for the CLSP, where the time-consuming W-W dynamic programming algorithm is replaced by the PPB heuristic in computing shadow prices, thus reducing the CPU time, with a slight loss in optimality. Hindi [29] proposed a variable redefinition approach for the CLSP. He uses computationally efficient approaches, such as the shortest-path and minimum-cost network flow, to obtain lower bounds and upper bounds in a branch-and-bound framework. Thizy and Wassenhove [30] proposed a Lagrangian relaxation approach for the CLSP. The Lagrangian relaxation problem consists of $|I|$ uncapacitated single-item problems and each of them is solved with W-W dynamic programming in $O(T^2)$. Feasible solutions are obtained from the resulting transportation problems. The Lagrange multipliers are updated by a subgradient procedure [31]. Trigeiro et al. [32] also proposed Lagrangian relaxation for the CLSP and considered the setup times. They found feasible solutions with a heuristic called the smooth procedure. Diaby et al. [33] proposed Lagrangian relaxation for the CLSP, in which Lagrangian relaxation is used within a branch-and-bound framework to generate bounds. However, the initial upper bounds are generated by three heuristics: (1) the DS

heuristic, (2) linear programming-based relaxation and (3) an extension of the solution procedure of Thizy and Van Wassenhove [30] with setup times and multiple resources.

Meta-heuristics have also been used to solve the CLSP. Xie and Dong [34] proposed a heuristic genetic algorithm (GA) for the CLSP. They designed a domain-specific encoding scheme for lot sizes and provided a heuristic shifting procedure for decoding for feasibility. The computational results show that the algorithm converges a solution within 200 generations for most of the small-scale examples ($N \leq 10$) and within 500 generations for most of the modest examples ($N \leq 20$). Gaafar [35] applied a genetic algorithm (GA) to solve a lot-sizing problem and considered batch ordering and backorders. This approach uses a "012" coding scheme specific to the problem. The computational results show that the proposed genetic algorithm outperforms the modified SM heuristic in terms of the optimal rate and the percentage deviation. Moreover, Gaafar et al. [36] also proposed a simulated annealing (SA) algorithm for a lot-sizing problem. In their implementation, the "012" coding scheme and 16 mutation-based moves are used to solve the dynamic lot-sizing problem, and the computational results show that the performance of the SA algorithm surpasses that of the compared GA and the modified SM heuristic. Hindi [37] proposed a tabu search (TS) heuristic for the CLSP. In his algorithm, a reformulation of the original problem is first executed by variable redefinition. The new problem has tighter bounds but more variables and thus is solved by column generation. The resulting feasible solution is further improved by solving a minimum-cost network flow problem, and then the improved solution is used as a starting point for a tabu search procedure. More studies for CLSPs with metaheuristics can be found [38].

Summarizing the above methods, we find that, for complex lot-sizing problems such as the CLSP, the Lagrange relaxation method has more advantages, which are as follows:

(1) It is a method based on mathematical optimization, which can obtain the optimal solution;
(2) It can deal with complex constraint relations, differing from the heuristic method, especially if the problem comes from a complex industrial production environment, most of which have complex industrial constraints and coupling relations;
(3) The Lagrange relaxation method has a rich theoretical background and is easy to implement.

Therefore, in this paper, we propose a Lagrange relaxation approach for the CLSP, and the work is closely related to that of [30]. However, the difference lies in the methods of finding the lower bounds and upper bounds. The contributions of this paper are as follows:

(1) A Lagrange relaxation approach for the CLSP is implemented;
(2) A stepping-stone algorithm is developed to solve the resulting transportation problem;
(3) A fix-up heuristic is proposed to obtain feasible solutions;
(4) A local neighborhood search heuristic is used to further search for high-quality solutions, which increases the likelihood of finding the optimal solution.

3. Solution Approach

3.1. Lagrange Relaxation

We propose a Lagrange relaxation approach for the CLSP [39,40]. Let u_t, $t = 1, \ldots, T$, be nonnegative Lagrange multipliers for capacity constraints (3); then, a Lagrange relaxation problem ($LR\ (u)$) can be obtained as follows.

$$
\begin{aligned}
LR\ (u): \quad z_u &= min \sum_{t=1}^{T} \sum_{i=1}^{I} (p_{it}x_{it} + h_{it}I_{it} + s_i y_{it}) + \sum_{t \in T} u_t \Big(\sum_i a_i x_{it} - C_t \Big) \\
&= \sum_{t \in T} \sum_{i \in I} ((p_{it} + u_t a_i)x_{it} + h_{it}I_{it} + s_i y_{it}) - \sum_{t \in T} u_t C_t \\
&s.t. \\
&(2), (4) - (6),\ u_t \geq 0, t \in T.
\end{aligned}
$$

The $LR\ (u)$ problem can be then decomposed into $|I|$ independent uncapacitated single-item lot-sizing problems, each of which is solvable in $O(T^2)$ [25], and any of the previously mentioned single-item problem methods—for example, lot-for-lot, LUC, W-W dynamic programming, etc.—can be used to this aim. In this paper, W-W dynamic

programming is chosen. Note that, for any given u, the solution of the $LR(u)$ will give a lower bound z_{LB} for the original problem (CLSP).

3.2. Obtaining Feasible Solutions

Let (y_u, x_u, I_u) be the optimal solution of z_u for a given Lagrange multiplier u; note that when the production periods are specified by fixing y_{it} at 0 or 1 for all $i = 1, \ldots, I$, $t = 1, \ldots, T$, the problem can be transformed into a transportation problem with $|T|$ origins and $|T| \times |I|$ destinations. Note also that some of the arcs of the transportation problem are forbidden due to either the prevention of backlogging or production not starting in that period.

The transportation problem (TP) is formulated as follows.

$$TP(y): \quad z_y = min \sum_{i \in I} \sum_{t \in T} \sum_{\tau \in T} \delta_{it\tau} \xi_{it\tau} \tag{13}$$

s.t.

$$\sum_{t \in T} \xi_{it\tau} \geq a_i d_{i\tau}, \; i \in I, t \in T \tag{14}$$

$$\sum_{i \in I} \sum_{\tau \in T} \xi_{it\tau} \leq C_t, \; t \in T \tag{15}$$

$$\xi_{it\tau} = 0, \text{ if } y_{it} = 0 \text{ or } \; t > \tau; \; i \in I, t \in T, \tau \in T \tag{16}$$

$$\xi_{it\tau} \geq 0, \; i \in I, t \in T, t \in T \tag{17}$$

$$\delta_{it\tau} = \begin{cases} (p_{it} + \sum_{q=t}^{\tau-1} h_{iq})/a_i, t \leq \tau \\ +\infty, \; otherwise \end{cases}, i \in I, t \in T, \tau \in T \tag{18}$$

$\xi_{it\tau}$: the amount of product i produced in period t to satisfy the demand in the future period τ.

To solve the resulting transportation problem TP (y), we develop a stepping-stone algorithm [41,42], which is shown in Algorithm 1.

Algorithm 1: Stepping-stone algorithm.

Input: matrix X, where x_{ij} is the number of units shipped from supply point i to demand point j
Output: the optimal basic feasible solution (*bfs*) X

1: Determine an initial basic feasible solution (*bfs*) with the matrix minimum rule.
2: Calculate dual variables u_i, $i = 1, \ldots, m$, v_j, $j = 1, \ldots, n$, for the current basis solution.
3: Find a non-basic variable with a negative reduced cost, min $\lambda = u_i + v_j - c_{ij} < 0$, which will enter the basis to improve the current basic solution; if such a variable x_{ij} is not found, then the optimal solution has been obtained; output the current *bfs* and stop. Otherwise, go to step 4.
4: Find the closed loop that includes the newly found non-basic variable x_{ij}.
5: Update the values of the elements along the found closed loop and obtain a new *bfs*.
6: Return to step 2.

Note that the solution to the transportation problem will give an upper bound z_{UB} for the CLSP if it is feasible. As an illustrative example from the literature [30], here, a 3-item $\times$ 4-period CLSP with a specified y is considered. The problem and data are shown in Tables 1–3. The iteration process of the algorithm is shown in Tables 4–8.

In Table 1, a number with a square indicates that production occurs in this period. In Table 3, the resulting transportation problem is formed with the y specified in Table 1. The marks X and X indicate that the corresponding arcs are forbidden, either because $y_{it} = 0$ or because $\tau < t$, respectively. The initial basis is listed in Table 4. The superscripts are the basis variable labels. The initial objective value is 385.3 (no setups included). The optimal solution is obtained with four iterations and the objective values are gradually

reduced to 285.3, 193.3, 162.3 and 146 (optimal). The four closed loops found are (14-1-0), (10-12-13-1-0-14-15), (13-1-0-14-15-10-4) and (13-1-0), which the components marked with the green background.

Table 1. Demand and capacity.

Item	Period				
	1	**2**	**3**	**4**	
1	20	30	40	10	demands
2	20	10	10	10	
3	25	30	30	30	
Capacity	450	400	450	300	

Table 2. Other data.

Item	p_i	a_i	s_i	h_i
1	-	5	70	3
2	-	4	90	4
3	-	6	200	5

Table 3. The transportation problem.

Reduction	Demand													$\|T\|$
	100	**150**	**200**	**50**	**80**	**40**	**40**	**40**	**150**	**180**	**180**	**180**	**210**	t
450	0	0.6	1.2	1.8	0	1	2	3	0	5/6	5/3	2.5	0	1
400	X	0	0.6	1.2	X	X	X	X	X	0	5/6	5/3	0	2
450	X	X	0	0.6	X	X	0	1	X	X	0	5/6	0	3
300	X	X	X	X	X	X	X	X	X	X	X	0	0	4
$\|T\| \times \|I\| + 1$		$\tau = 1, 2, 3, 4; i = 1$				$\tau = 1, 2, 3, 4; i = 2$					$\tau = 1, 2, 3, 4; i = 3$		dummy	

Table 4. The initial basic feasible solution and first closed loop.

100^{9}				80^{8}	40^{11}		40^{15}	150^{7}			40^{14}	X	450	1
	150^{6}		20^{12}						180^{5}		50^{13}		400	2
		200^{4}	30^{10}			40^{3}				180^{2}			450	3
											90^{1}	210^{0}	300	4
100	150	200	50	80	40	40	40	150	180	180	180	210		

Table 5. The second basic feasible solution and closed loop.

100^{9}				80^{8}	40^{11}		40^{15}	150^{7}				40^{14}	450	1
	150^{6}		20^{12}						180^{5}		50^{13}		400	2
		200^{4}	30^{10}			40^{3}	X			180^{2}			450	3
											130^{1}	170^{0}	300	4
100	150	200	50	80	40	40	40	150	180	180	180	210		

Table 6. The third basic feasible solution and closed loop.

100^{9}				80^{8}	40^{11}		10^{15}	150^{7}				70^{14}	450	1
	150^{6}	X	50^{12}						180^{5}		20^{13}		400	2
		200^{4}				40^{3}	30^{10}			180^{2}			450	3
											160^{1}	140^{0}	300	4
100	150	200	50	80	40	40	40	150	180	180	180	210		

Table 7. The fourth basic feasible solution and closed loop.

100^{9}				80^{8}	40^{11}			150^{7}				80^{14}	450	1
	150^{6}	10^{15}	50^{12}						180^{5}		10^{13}	X	400	2
		190^{4}				40^{3}	40^{10}			180^{2}			450	3
											170^{1}	130^{0}	300	4
100	150	200	50	80	40	40	40	150	180	180	180	210		

Table 8. The optimal basic solution.

100^{9}				80^{8}	40^{11}			150^{7}				80^{14}	450	1
	150^{6}	10^{15}	50^{12}						180^{5}			10^{13}	400	2
		190^{4}				40^{3}	40^{10}			180^{2}			450	3
											180^{1}	120^{0}	300	4
100	150	200	50	80	40	40	40	150	180	180	180	210		

3.3. Fix-Up Heuristic

For a given y_u, from the $LR(u)$, the resulting transportation problem TP(y) often is infeasible. To obtain a feasible solution, we introduce a fix-up heuristic. The fix-up heuristic shifts the lots among periods using lot techniques. The fix-up heuristic is shown in Algorithm 2.

Algorithm 2: Fix-up heuristic.

1. Check infeasible period cp, compute the remaining capacity RC_j, $RC_j = \sum_{i \in I} a_i d_{ij} - L_j$, for each period $j = 1, \ldots, T$.
2. If ($cp < T$), go to forward pass; otherwise, go to backward pass.
3. Forward pass: for each product, check inventory period by period and move product lots forward into periods $cp + 1$, $cp + 2$, …, T, in a cost-effective fashion, until no sufficient capacity is available. Go to 4.
4. Backward pass: if, in the following periods, no excessive capacities exist, move product lots backward into periods $cp - 1$, $cp - 2$, …, in a cost-effective fashion.
5. Iteratively execute 3 and 4 until a feasible solution is obtained.

3.4. Local Neighborhood Search Algorithm

To find more high-quality solutions, we proposed a local neighborhood search algorithm for the current feasible solution. The neighborhood structures are defined as follows.

(1) The neighborhood of the lot move

The first neighborhood structure is for the shifting of lots between two periods for a single item. Assume that there exists an item i, and, for any two periods t_1 and t_2, there are two lots $x_{it_1} > 0$,$x_{it_2} > 0$, in the corresponding period. Again, assume that there are no positive lots among t_1 and t_2, i.e., the two periods t_1 and t_2 can be discontinuous, but there must be no production occurring between them. The neighborhood of the lot move is then defined for all possible moves between t_1 and t_2 as stated above. As an illustrative example, let $\Delta > 0$ be the lot from period t_1 to period t_2; then, we have

$$\begin{cases} x_{it_1} = x_{it_1} - \Delta \\ x_{it_2} = x_{it_2} + \Delta' \end{cases} \tag{19}$$

Note that the lot move may be bidirectional. A forward move reduces the inventories, while a backward move may reduce the setups. The cost change can be computed from Formulas (20) and (21).

$$\text{cost} = -\lambda h_i \cdot \Delta - s_i, \text{ forforwardmoves} \tag{20}$$

$$\text{cost} = \lambda h_i \cdot \Delta - s_i, \text{ forbackwardmoves} \quad (21)$$

When the inventory cost h_i changes from period to period, the formula needs to be adjusted correspondingly. The cardinal of the lot-move neighborhood is $O(n^3)$.

(2) The neighborhood of the 2-opt lot exchanges

This neighborhood is for the 2-opt lot exchanges, which change the lots for two items in two continuous periods. Assume that there are two items i_1, i_2 and two continuous periods, t_1 and t_2, and the related lots are $x_{i_1t_1}, x_{i_1t_2}, x_{i_2t_1}, x_{i_2t_2}$, respectively. Now, we swap two lots for two items i_1 and i_2 in two periods t_1 and t_2, and assume that $\Delta_1 \leq \min(x_{i_1t_2}, x_{i_2t_1})$, $\Delta_2 \leq \min(x_{i_1t_1}, x_{i_2t_2})$. Then, we have

$$\begin{cases} x_{i_1t_1} = x_{i_1t_1} + \Delta_1 \\ x_{i_1t_2} = x_{i_1t_2} - \Delta_1 \\ x_{i_2t_1} = x_{i_2t_1} - \Delta_1 \\ x_{i_2t_2} = x_{i_2t_2} + \Delta_1 \end{cases}, \text{ for forward exchange, } \begin{cases} x_{i_1t_1} = x_{i_1t_1} - \Delta_2 \\ x_{i_1t_2} = x_{i_1t_2} + \Delta_2 \\ x_{i_2t_1} = x_{i_2t_1} + \Delta_2 \\ x_{i_2t_2} = x_{i_2t_2} - \Delta_2 \end{cases}, \text{ for backward exchange.}$$

The cardinal of the neighborhood is $O(n^4)$.

As an illustrative example, below, a 3-item × 4-period lot-sizing problem is considered [15]. The data are referred to in the literature. The 2-opt lot exchange local neighborhood search results are shown in Figure 1.

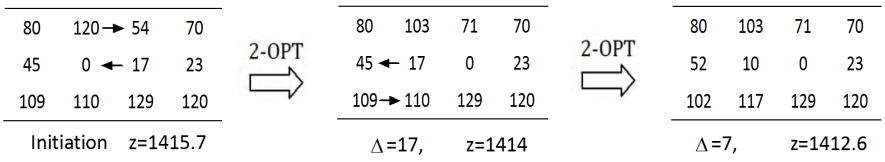

Figure 1. A 2-opt lot exchange for solution improvement.

An initial solution is obtained from the DS heuristic, and the objective function value is 1415.7. Then, the two 2-opt lot exchanges are performed and two improved solutions are obtained with values of 1414 and 1412.6, which are optimal.

3.5. Subgradient Algorithm

For any given u, the solution of the LR(u) gives a lower bound z_u for the original CLSP. To obtain the maximum lower bound, a subgradient algorithm [31] is proposed in this paper and, according to subgradient theory, $\sum_{i \in I} a_i x_{it} - C_t$ is a subgradient of function z_u at point u. The Lagrange multipliers u can be updated with the formula

$$u_t^{k+1} = \max\left\{0, u_t^k + s^k\left(\sum_{i \in I} a_i x_{it}^k - C_t\right)\right\}, s^k = \frac{\eta^k\left(z^{best} - z_u^k\right)}{\sum_{t \in T}\left(\sum_{i \in I} a_i x_{it}^k - C_t\right)^2} \quad (22)$$

where x_{it}^k is the solution of *LR* (u^k), and z^{best} is the best upper bound found so far, $\eta^0 = 2$. The subgradient algorithm is shown in Algorithm 3.

Algorithm 3: Subgradient algorithm.

1: given a Lagrangian multiplier u^k, $k = 0$;
2: loop
3: Arbitrarily select a subgradient from $\partial(z_u)$;
If any of the termination criteria are met, then stop; / /see note 2
else
$u^{k+1} = \max\{0, u^k + \theta_k s^k\}$; //see note 1
4: $k = k + 1$;
5: if $k > N$ then stop;
6: end loop

Notes.

1. Selection of θ_k:
   (1) $\sum_{k=1}^{\infty} \theta_k = \infty, \theta \to 0, k \to \infty$;
   (2) $\theta_k = \theta_0 \rho^k$, 0 < ρ<1;
   (3) $\theta_k = \frac{z_{UB}^k - z_{LB}^k}{\left\| s^k \right\|} \eta_k$.
2. Termination criteria:
   (1) Iterations: N,
   (2) $s^k = 0$ or $\left\| s^k \right\| \leq \varepsilon$,
   (3) $| {z_{UB}}^k - {z_{LB}}^k | < \varepsilon$
   (4) u^k or $z(u^k)$ does not change within a given number of iterations (e.g., 7).

3.6. Lagrange Relaxation Algorithm

Based on the above description of the components of the Lagrange relaxation algorithm, now, we can give the complete Lagrange relaxation algorithm shown in Algorithm 4.

Algorithm 4: Lagrange relaxation algorithm.

```
//    Initialization
1:    k = 0
2:    LB = −1 × 10^10          //a sufficiently small lower bound
3:    UB = 1 × 10^10           //a sufficiently large upper bound
4:    Gap = 1 × 10^−2          //a dual gap percentage
5:    K = 5000                 //the maximum number of iterations
6:    η_k = 2.0                //step size
7:    u_k = 0                  //Lagrangian multipliers
8:  loop                       //main cycle beginning
9:    Solve the Lagrangian relaxation LR (u_k) by W-W dynamic programming, and
      calculate the current lower bound Z_LB (u_k)
10:   Solve the resulting transportation problem with the stepping-stone algorithm for the y
from the solution of the current LR to obtain a Z_UB (u_k) if it is feasible
      else
      {
      Execute the fix-up heuristic
      Solve the corresponding transportation problem using the stepping-stone algorithm
      }
11:   Execute the local neighborhood search algorithm to increase the solution quality
12:   if Z_LB (u_k) ≥ Z_LB, Z_LB = Z_LB (u_k)        //updating low bounds
13:   if Z_UB (u_k) ≤ Z_UB, Z_UB = Z_UB (u_k)        //updating upper bounds
14:   if (Z_UB^k − Z_LB^k)/Z_UB^k ≤ Gap, then stop
15:   k++, if k ≥ K then stop;
16:   Update the Lagrangian multipliers with the subgradient algorithm
17:   η_k = 1/2*η_k;           //update the step size
18:   continue;
19:  end loop                  //main cycle ending
```

4. Computational Results

4.1. Comparison Study

The Lagrange relaxation algorithm proposed in this paper has been implemented on a personal computer using the *VC++* 6.0 programming language. Computational experiments for the benchmarks and randomly generated large-sized CLSPs are performed. The benchmarks are from the literature [23]. The data are listed in Table 9. The computational results are listed in Table 10.

Table 9. Data for problems TVW1-4.

Item	h_i	s_i	Period								TBO ①
			1	2	3	4	5	6	7	8	
1	1	100	-	70	50	100	20	80	-	100	2
2	1	200	20	40	50	10	30	-	40	50	3.65
3	1	200	40	50	-	100	40	80	90	160	2.33
4	1	300	-	100	100	150	160	90	100	100	2.5
5	1	400	50	-	20	40	10	10	20	10	6.32
6	1	250	70	40	40	40	100	20	40	50	3.16
7	1	500	-	20	50	10	20	60	40	40	5.77
8	1	300	10	20	-	-	10	10	20	30	6.93
Available demand			190	340	310	450	390	350	350	540	Total demand 2920
TVW1			350	350	350	400	400	400	400	500	3150 (93%)
TVW2			400	400	400	400	400	400	400	400	3200 (91%)
TVW3			500	500	500	500	500	500	500	500	4000 (73%)
TVW4			600	600	600	600	600	600	600	600	4800 (61%)

Notes: ① TBO = $(2s/hd)^{1/2}$, time between orders; $a_i = 1$, for all $i = 1, \ldots, I$, hence not appeared in the table.

Table 10. Computational results for benchmarks.

Problem ID	LP Relaxation	Optimal Solution	LR	LRFN	DS	LV
TVW1	7996.67	8430	8710	**8520**	8710	8970
TVW2	7722.27	7910	7930	**7910**	7930	8800
TVW3	7534.17	7610	7610	**7610**	7970	7970
TVW4	7446.17	7520	7520	**7520**	8000	8000

The four approaches, LRFN (the proposed approach), LR (without the local neighborhood search), DS and LV, are compared, wherein the results of DS and LV are from the literature [23]. The LP relaxation and optimal solutions are from CPLEX 7.0. It can be seen that the results of the LRFN proposed in this paper are better compared to the others. For problems with looser capacity constraints, like TVW3 and TVW4, their capacity utilization ratios are below 73%, and the LRFN obtains the optimal solution. For problems with tighter capacity constraints, such as TVW2, the capacity utilization ratio is 91%. The LRFN obtains the optimal solution, and the others obtain near-optimal solutions. LV has a larger error, about 11.3%. For the tightest capacity problem, TVW1, its capacity utilization ratio reaches 93%, the LRFN obtains the best near solution, and the error is below 1%, while the errors of DS and LV are 3.3% and 6.4%, respectively. The computational results show that the LRFN has the best performance. The performance comparison is shown in Figure 2.

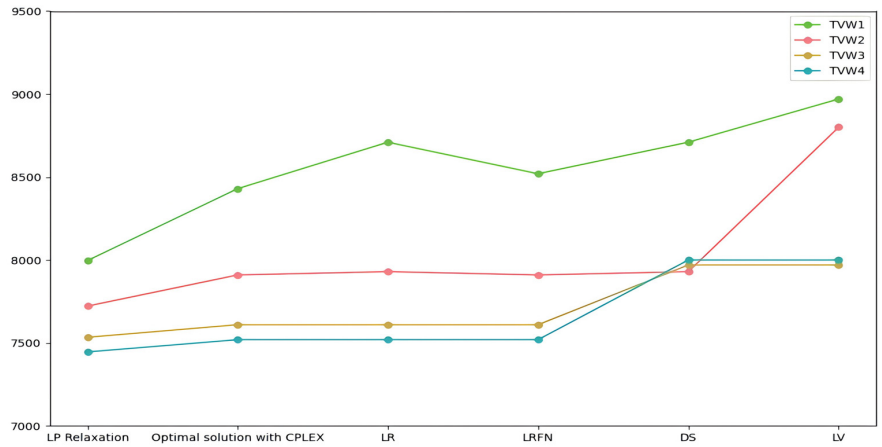

Figure 2. Computational results comparison.

4.2. More Computational Examples

The benchmarks that we previously computed have been regarded as traditional problems in lot-sizing publications. However, the difficulty of lot-sizing problems can be evaluated with the following criteria.

(1) The number of time periods $|T|$ and the number of items $|I|$.

(2) The tightness of the capacity utilization: $\frac{\sum_{t=1}^{T}\sum_{i=1}^{I} d_{it}}{\sum_{t=1}^{T} C_t}$.

Thus, we compute larger problems at different difficulty levels. The data of these problems are randomly generated and adhere to the following limitations:

(1) the maximal number of items is set to 512;
(2) the number of periods is set to 48;
(3) the maximal utilization ratio is set to 93%.

The other problem-specific data are randomly generated—for example, C_t: rand (800, 2000), d_{it}: rand (0,100), s_i: rand (100, 2000), p_i: (1,8), h_i: (1,4) and a_i = 1, and so on. The computational results for these problems are listed in Table 11.

Table 11. Computational results of large-sized problems.

Problem No.	Problem Size (Items × Periods)	LB	Optimal Solution	Iteration Found	Gap (%)	CPU Time (s)
1	5 × 36	82,131.54	85,374	201	3.95	68.73
2	5 × 48	111,368.25	115,136	215	3.38	215
3	10 × 10	84,904.29	88,094	2	3.76	5.33
4	10 × 15	67,167.7	67,906	49	1.1	1.5
5	10 × 15	71,694.02	71,705	8	0.02	0.03
6	10 × 20	151,859.54	157,470	87	3.69	40.75
7	10 × 20	85,426.11	85,619	1	0.23	2.47
8	15 × 20	160,088.16	161,128	14	0.65	7.4
9	10 × 24	103,383.29	103,866	10	0.47	82.97
10	15 × 36	57,341.8	58,184	40	1.47	66.08
11	10 × 36	220,486.05	224,047	214	1.62	380.94
12	10 × 30	224,231.93	226,571	7	1.04	161.01
13	20 × 20	261,226.71	267,422	2	2.37	16.65
14	20 × 30	379,853.7	383,533	3	0.97	50.51
15	10 × 48	358,551	392,750	1	9.54	1147.91

In Table 11, the column "Iteration Found" indicates that the best solution is found in a certain iteration. The meanings of the other columns are clear. As shown in Table 11, the average dual gap is within 2% (exactly 1.76), with the exceptions of some extreme examples, like problem No. 15, where the dual gap is 9.54. The reason for this result may be the excessively tight capacity utilization. Usually, the looser the capacity utilization ratio, the smaller the dual gap is, and vice versa.

5. Conclusions

In this paper, we propose a Lagrange relaxation approach to solve the capacitated multi-item lot-sizing problem, the CLSP. The CLSP is often encountered in industry production settings and the study of its solution approaches is significant. Our approach represents a deliberate attempt to solve such industrial problems. The main work includes (1) developing a stepping-stone algorithm to solve the resulting transportation problem to obtain the upper bounds; (2) developing a fix-up procedure to obtain feasible solutions; (3) developing a local search heuristic to obtain high-quality solutions; and (4) providing a complete Lagrange relaxation algorithm framework. The computational experiments on the benchmarks and randomly generated lot-sizing problems verify the effectiveness and efficiency of the proposed solution approach. There are some limitations to our method:

(1) Parameter setting in the subgradient algorithm;
(2) A feasible solution fix-up policy;
(3) Subgradient algorithm substitution, such as multiplier adjustment methods.

These also constitute the directions of our ongoing research.

Author Contributions: Conceptualization, Z.G. and D.L.; data curation, D.L.; formal analysis, D.W.; methodology, Z.G.; experimental design and testing, D.L.; software, D.W. and Z.Y.; supervision, Z.G.; validation, Z.Y.; writing—original draft, Z.G.; writing—review and editing, D.W. All authors have read and agreed to the published version of the manuscript.

Funding: This study was partly supported by the Major Program of the National Natural Science Foundation of China (72192830, 72192831) and the 111 Project (B16009).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Drexl, A.; Kimms, A. Lot sizing and scheduling-survey and extensions. *Eur. J. Oper. Res.* **1997**, *99*, 228–249. [CrossRef]
2. Maes, J.; Van Wassenhove, L.N. Multi-item single-level capacitated dynamic lot-sizing heuristics: A general review. *J. Oper. Res. Soc.* **1988**, *39*, 991–1004. [CrossRef]
3. Karimi, B.; Fatemi Ghomi, S.M.T.; Wilson, J.M. The capacitated lot sizing problem: A review of models and algorithms. *Omega* **2003**, *31*, 365–378. [CrossRef]
4. Florian, M.; Lenstra, J.K.; Rinnooy Kan, A.H.G. Deterministic production planning algorithms and complexity. *Manag. Sci.* **1980**, *26*, 669–679. [CrossRef]
5. Tang, L.; Meng, Y.; Liu, J. An improved Lagrangean relaxation algorithm for the dynamic batching decision problem: Lot sizing and scheduling: New models and solution approaches to address industrial extensions. *Int. J. Prod. Res.* **2011**, *49*, 2501–2517. [CrossRef]
6. Gao, Z.; Tang, L.; Jin, H.; Xu, N. An Optimization Model for the Production Planning of Overall Refinery. *Chin. J. Chem. Eng.* **2008**, *6*, 67–70. [CrossRef]
7. Zhang, X.F.; Boutat, D.; Liu, D.Y. Applications of fractional operator in image processing and stability of control systems. *Fractal Fract.* **2023**, *7*, 359. [CrossRef]
8. Zhang, X.F.; Lin, C.; Chen, Y.Q.; Boutat, D. A unified framework of stability theorems for LTI fractional order systems with $0 < \alpha < 2$. *IEEE Trans. Circuit Syst. II Express Briefs* **2020**, *67*, 3237–3241.
9. Zhang, X.F.; Chen, S.N.; Zhang, J.X. Adaptive sliding mode consensus control based on neural network for singular fractional order multi-agent systems. *Appl. Math. Comput.* **2022**, *336*, 127442. [CrossRef]
10. Jans, R.; Degraeve, Z. Modeling industrial lot sizing problems: A review. *Int. J. Prod. Res.* **2008**, *46*, 1619–1643. [CrossRef]
11. Buschkuhl, L.; Sahling, F.; Helber, S.; Tempelmeier, H. Dynamic capacitated lot-sizing problems: A classification and review of solution approaches. *OR Spectr.* **2010**, *32*, 231–261. [CrossRef]
12. Daryna, D.; Christian, A. New construction heuristic for capacitated lot sizing problems. *Eur. J. Oper. Res.* **2023**, *311*, 906–920.
13. Dixon, P.S.; Silver, E.A. A heuristics solution procedure for multi-item, single-level, limited capacity, lot-sizing problem. *J. Oper. Manag.* **1981**, *2*, 23–39. [CrossRef]
14. Maes, J.; Van Wassenhove, L.N. A simple heuristic for the multi-item single level capacitated lot sizing problem. *Oper. Res. Lett.* **1986**, *4*, 265–274. [CrossRef]
15. Seiringer, W.; Castaneda, J.; Altendorfer, K.; Panadero, J.; Juan, A.A. Applying Simheuristics to Minimize Overall Costs of an MRP Planned Production System. *Algorithms* **2022**, *15*, 40. [CrossRef] [PubMed]
16. Cyril, K.; Taha, A.; Yassine, O.; Farouk, Y.; Humbert, D.B.; Nicolas, J.; Antoine, D.W. A metaheuristic approach for solving a simultaneous lot sizing and scheduling problem with client prioritization in tire industry. *Comput. Ind. Eng.* **2022**, *165*, 107932.
17. Gurkan, M.E.; Tunc, H. A fix-and-optimize heuristic for the capacitated multi-item stochastic lot-sizing problem. *Int. J. Optim. Control Theor. Appl.* **2021**, *11*, 41–51. [CrossRef]
18. Harris, F.W. How many parts to make at once. *Fact.—Mag. Manag.* **1913**, *10*, 135–136, 152. [CrossRef]
19. Gorham, T. Dynamic order quantities. *Prod. Inventory Manag. J.* **1968**, *9*, 75–79.
20. DeMatteis, J.J. An economic lot-sizing technique. I: The part-period algorithm. *IBM Syst. J.* **1968**, *7*, 30–38. [CrossRef]
21. Mendoza, A.G. An economic lot-sizing technique. II: Mathematical analysis of the part period algorithm. *IBM Syst. J.* **1968**, *7*, 39–46. [CrossRef]

22. Silver, E.A.; Meal, H.C. A heuristic for selecting lot size quantities for the case of a deterministic time varying demand rate and discrete opportunities for replenishment. *Prod. Inventory Manag.* **1973**, *14*, 64–74.
23. Lambrecht, M.R.; Vanderveken, H. Heuristics procedures for the single operation, multi-item loading problem. *AIIE Trans.* **1979**, *11*, 319–326. [CrossRef]
24. Eisenhut, P.S. A Dynamic Lot Sizing Algorithm with capacity constraints. *AIIE Trans.* **1975**, *7*, 170–176. [CrossRef]
25. Wagner, H.M.; Whitin, T.M. Dynamic version of the economic lot size model. *Manag. Sci.* **1958**, *5*, 89–96. [CrossRef]
26. Barany, I.; Van Roy, T.J.; Wolsey, L.A. Strong formulations for multi-item capacitated lot sizing. *Manag. Sci.* **1984**, *30*, 1255–1261. [CrossRef]
27. Lasdon, L.S.; Terjung, R.C. An efficient algorithm for multi-item scheduling. *Oper. Res.* **1971**, *19*, 946–969. [CrossRef]
28. Bahl, H.C. Column generation based heuristic algorithm for multi-item scheduling. *AIIE Trans.* **1983**, *15*, 136–141. [CrossRef]
29. Hindi, K.S. Computationally efficient solution of the multi-item, capacitated lot-sizing problem. *Comput. Ind. Eng.* **1995**, *28*, 709–719. [CrossRef]
30. Thizy, J.M.; Van Wassenhove, L.N. Lagrangian relaxation for the multi-item capacitated lot-sizing problem, a heuristics approach. *AIIE Trans.* **1985**, *17*, 308–313.
31. Held, M.; Wolfe, P.; Crowder, H.P. Validation of subgradient optimization. *Math. Program.* **1974**, *6*, 62–88. [CrossRef]
32. Trigeiro, W.W.; Thomas, L.J.; McLain, J.O. Capacitated lot-sizing with setup times. *Manag. Sci.* **1989**, *35*, 353–366. [CrossRef]
33. Diaby, M.; Bahl, H.C.; Karwan, H.M.; Zionts, S. Capacitated lot-sizing and scheduling by Lagrangean relaxation. *Eur. J. Oper. Res.* **1992**, *59*, 444–458. [CrossRef]
34. Xie, J.; Dong, J. Heuristic genetic algorithms for general capacitated lot-sizing problems. *Comput. Math. Appl.* **2022**, *44*, 263–276. [CrossRef]
35. Gaafar, L. Apply genetic algorithms to dynamic lot sizing with batch ordering. *Comput. Ind. Eng.* **2006**, *51*, 433–444. [CrossRef]
36. Gaafar, L.K.; Nassef, A.O.; Aly, A.I. Fixed-quantity dynamic lot sizing using simulated annealing. *Int. J. Adv. Manuf. Technol.* **2009**, *41*, 122–131. [CrossRef]
37. Hindi, K.S. Solving the CLSP by a tabu search heuristic. *J. Oper. Res. Soc.* **1996**, *47*, 151–161. [CrossRef]
38. Jans, R.; Degraeve, Z. Meta-heuristics for dynamic lot sizing: A review and comparison of solution approaches. *Eur. J. Oper. Res.* **2007**, *177*, 1855–1875. [CrossRef]
39. Fisher, M.L. The Lagrangian relaxation method for solving integer programming problems. *Manag. Sci.* **1981**, *27*, 1–18. [CrossRef]
40. Fisher, M.L. An applications oriented guide to Lagrangian relaxation. *Interfaces* **1985**, *15*, 10–21. [CrossRef]
41. Charnes, A.; Cooper, W.W. The stepping-stone method of explaining linear programming calculations in transportation problems. *Manag. Sci.* **1954**, *1*, 49–69. [CrossRef]
42. Glover, F.; Klingman, D. Locating stepping-stone paths in distribution problems via the predecessor index method. *Transp. Sci.* **1970**, *4*, 220–225. [CrossRef]