Natthanan Praseeratasang et al.

Article

Adaptive large neighborhood search to solve multi-level scheduling and assignment problems in broiler farms

Journal of Open Innovation: Technology, Market, and Complexity

Provided in Cooperation with:
Society of Open Innovation: Technology, Market, and Complexity (SOItmC)

Suggested Citation: Natthanan Praseeratasang et al. (2019) : Adaptive large neighborhood search to solve multi-level scheduling and assignment problems in broiler farms, Journal of Open Innovation: Technology, Market, and Complexity, ISSN 2199-8531, MDPI, Basel, Vol. 5, Iss. 3, pp. 1-20, https://doi.org/10.3390/joitmc5030037

Journal of Open Innovation: Technology, Market, and Complexity

Article

Adaptive Large Neighborhood Search to Solve Multi-Level Scheduling and Assignment Problems in Broiler Farms

Natthanan Praseeratasang [1], Rapeepan Pitakaso [1,*], Kanchana Sethanan [2], Monika Kosacka-Olejnik [3], Sasitorn kaewman [4] and Chalermchat Theeraviriya [5]

1 Faculty of Engineering, Ubon Ratchathani University, Ubon Ratchathani 34190, Thailand
2 Department of Industrial Engineering, Faculty of Engineering, Khon Kaen University, Khon Kaen 40002, Thailand
3 Faculty of Engineering Management, Poznan University of Technology, Poznań 61-704, Poland
4 Department of Computer Science, Faculty of Informatics, Mahasarakham University, Maha Sarakham 44150, Thailand
5 Faculty of Engineering, Nakhon Phanom University, Nakhon Phanom 48000, Thailand
* Correspondence: rapeepan.p@ubu.ac.th; Tel.: +6685-921-0826

Received: 30 May 2019; Accepted: 24 June 2019; Published: 26 June 2019

Abstract: This research aimed to present a solution to the problem of production scheduling and assignment in broiler farms, which thus enabled the farms to achieve maximum profit. In the operation of farms, there are many factors that affect profits, such as the number of broilers being consistent with the demand of production plants, including profits from the sales and transportation costs. Therefore, we formulated a mathematical model and tested it while using three problem groups through the Lingo v.11 program. The results indicated that this mathematical model could find a suitable solution. However, finding the best solution had time constraints, which resulted in various other problems that prevented a search for an optimal solution due to time consumption exceeding 72 h. We developed an algorithm using the Adaptive Large Neighborhood Search (ALNS) method in order to find another possible solution using a shorter time period, which consisted of ALNS1, ALNS2, and ALNS3. These algorithms are based on a combination of the method of destruction solutions and methods accepting different solutions. We aimed to effectively solve the problems and ensure that they are appropriate for the case study, a broiler farm in Buriram. When comparing the algorithm efficiency with the Lingo v.11 program, it was found that the ALNS1 algorithm was the most suitable for finding the optimal solution in the shortest time, which resulted in a 5.74% increase in operating profits.

Keywords: adaptive large neighborhood search; broiler farm management; scheduling and assignment problems; Adaptive Large Neighborhood Search (ALNS)

1. Introduction

Open innovation is formulated to drive the success of a company, while using outside information as well as inside information. A company has better opportunities to track the new technologies that can be used to enhance the competitiveness of the company in the real-world economy once open innovation is adapted. Optimization software, tools, and methods comprise the knowledge that all of the companies can use in order to enable increased profit, which is the primary goal of every private organization. This research aims in developing a suitable methodology to solve multi-level scheduling and assignment problems while using the case study of a broiler production plant.

At present, Thailand has the largest amount of broiler production in the association of Southeast Asian nations (ASEAN), which is composed of Indonesia, Malaysia, the Philippines, Singapore, Thailand, Brunei Darussalam, Cambodia, Laos, Myanmar (Burma), and Vietnam [1]. The broiler industry is an industry that is important to the national economy, with an amount of export volume comprising more than 85% of all livestock products, because Thailand has long been associated with raising chickens. At present, the broiler industry in Thailand is considered to be an opportunity to export to the ASEAN market because of the integration of the ASEAN Economic Community (AEC), which resulted in purchasing power and increasing demand for broilers among ASEAN member countries. At the same time, when Thailand has the ASEAN market to support its market expansion, a lower average production cost per unit will result. Therefore, the operators will be able to diversify their investments in broiler production, which ranged from the planting of raw materials to animal feed, broiler farm business, and processing business to other ASEAN countries according to the distinctive characteristics of the country. This is an advantage in terms of increasing the competitiveness of operators, as well as production efficiency, production factors, production standards, and production technology.

Buriram Province is one of the provinces in the northeast or Isan region of Thailand. It has a higher number of broiler farmers when compared to other provinces in the region, because it is a province with an area and physical characteristics that are suitable for raising broilers. It is also close to the source of raw materials that are used for broiler-raising, meaning that the cost of raising them is lower than for other provinces in the region [2]. As a result, some of the farmers in the province have turned their attention to broiler farms. At present, there are 177 broiler farms in total, which have been registered with the provincial livestock office. The total amount of broilers is 18,906,670. Most of the broilers for each farm are raised for sale to production plants. There are nine standard production plants that can support the amount of broilers that come from the farms in the region. The common criteria used in broiler-raising for each farm and for each batch is that all of the chickens that come into the farm must go out at the same time. When each batch of broilers is out, the farm will have to rest the coop for 35–42 days before taking the next batch of broilers into the coop. This helps to prevent disease, because each broiler farm has a different amount of stock. Additionally, each production plant has different needs. Therefore, each farm needs to allocate an amount of broilers that suits the needs of the production plant. Each farm must be responsible for all of its costs, including those that are incurred from delivering broilers to the production plants. As a result, each farm must be appropriately and efficiently managed by scheduling the production and assignment of tasks in the farm, both for the allocation of chicken quantity and for the selection of the size of the trucks used to effectively and promptly deliver the broilers to gain the highest possible operating profit. Figure 1 shows the framework of the proposed problem. Therefore, a suitable model for solving the multi-stage problems was determined in this research. The first step was to assign tasks to each farm to raise each batch of broilers. The next step was to allocate the number of broilers that are needed to meet the needs of the production plant, while the final step was to deliver the broilers to the production plant.

In Figure 1, the maximum profit is the aim of decision-makers. The production plants determine the chicken demand in each planning period. Decision-makers must then determine during which period farms (each farm has a different capacity) have to produce a certain amount of chickens to supply the production plant. A truck will be assigned to deliver the chickens to the production plants when the chickens are ready to be produced in the production plants. The transportation cost was different when we assigned different trucks to a farm because the trucks have different sizes and fuel consumption rates. Different trucks have different rental costs and fuel consumption rates depending on the size and the number of years that the truck has been used. Therefore, the truck that the farm will select must generate the lowest cost for the farm. Several production plants are available, so the decision-maker must assign the right farm to the right factory in order to gain maximum profit.

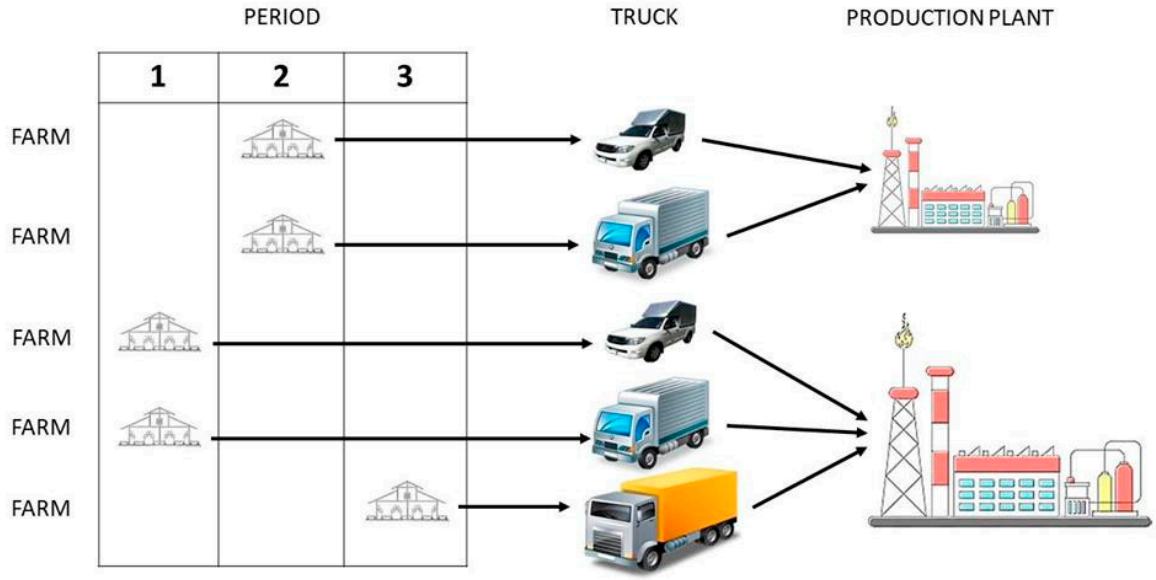

Figure 1. Supply chain of farms and production plants.

As we previously mentioned, the problem is a combination of production scheduling and assignment. Srivarapongse and Pijitbanjong have previously discussed he assignment problem in the agricultural industry [2]. They presented a special case of the generalized assignment problem (S-GAP) of assigning the driver to the truck. Subsequently, the truck was assigned to harvest sugar cane. The objective of the study was to generate the maximum area that can be used for harvesting in a single day. Our article will assign the ideal amount of chickens to the most suitable farm and assign the most suitable truck to deliver the chickens to the broiler factory. Each time period has different demands, which behooves the decision-maker to assign the right amount to fulfill all of the demands. The study by Srivarapongse and Pijitbanjong did not discuss this [2]. Moreover, in this article, a new solving method based on an algorithm called Adaptive Large Neighborhood Search (ALNS) will be presented. Generally, ALNS is comprised of three steps: (1) generate an initial solution, (2) perform a destroy operation, and (3) perform a repair operation. We present a new design for a destroy operator, which does not appear in any article published so far in the literature, which is called the freeze zone destroy operator.

This study uses the ALNS algorithm to solve the scheduling and assignment problems for broiler farms in Buriram Province. We have developed three solutions. We review and present some of the literature on production scheduling and assignment in Section 2. Section 3 gives an explanation of the problems and mathematical patterns, while the ALNS algorithm is presented in Section 4. Results and discussion regarding the ALNS algorithms are provided in Section 5, and Section 6 presents a summary.

2. Literature Review

A chicken farm is sometimes called a poultry or broiler farm. Significant research has been done with regard to deaths, food, and growing periods, which can increase the production rate or reduce the loss from death and lost weight [3–5]. The production planning of chicken farms, including egg or meat farms, has been studied by some researchers, such as Mohaddes [6], who tried to reduce the production cost of growing chickens by finding the most suitable level of food supply and the number of chickens in each farm. Demircan et al. [7] tried to generate maximum profit by determining the optimal farm size, so that feed consumption, production cost, and profitability per chicken could be optimized. In this study, we consider the optimal farm size and the parameters obtained from previous literature as ideal for production planning. Therefore, the optimal production plans to obtain the best chickens will be retained. The production plants determine the chicken demand in each period. The chickens that are produced from the farms fulfill the demand. The farms are differently sized.

Therefore, a good assignment must be made, so that the chicken demand is satisfied, because there are a limited number of farms available. The maximum possible income will be obtained when all of the demand is satisfied. The profit of the production planning is the income minus the cost, which is assumed to be the operating cost and the transportation cost of the farm. All of the farms are assumed to have the same operating cost, but the transportation cost is different because each farm is located in a different location. The transportation cost depends on several factors: (1) the amount of chickens that it carries; (2) the fuel consumption rate of the truck, which depends on the size and the number of years it has been in use; and, (3) the distance between the farm and the production plant. However, the maximum profit can be increased by placing the right amount of chickens on the right farm and delivering the chickens to the right production plant.

The heuristic and meta-heuristic methods that are available in a variety of ways are widely used to meet various challenges, such as the generalized assignment problem [8], scheduling shipments [9], problems in assigning general work [10], problems concerning assignments and job allocations [11], and problems concerning multi-stage and multi-purpose assignments [12]. Ross [13] first presented problems that are related to general assignments (GAPs), who found that they were difficult problems with NP-hard levels [14]. GAPs are often solved with the exact method. Small problems, or problems with 200 jobs and 20 employees, are the biggest problems that can be solved by the exact method [2]. Therefore, the heuristics method has been used to solve GAPs [8]. In economics, the use of problems that are related to assignment is intended to seek the lowest cost for assignment [15]. For multiple resources in GAPs [16], multi-level GAPs [17] are extended versions of GAPs, where the assignment of a task to a machine is considered. Multiple resources need to be taken care of, and the multi-level needs to be executed in the assigning phase, and this includes assigning the driver to the truck and assigning the truck that collects the product from customers. Solving problems that are related to assignments will mainly focus on optimal solutions while using various methods, such as the Hungarian method [18], or simplex methods, such as the branch-and-bound method [19], an improved simple depth-first Lagrangian branch-and-bound method [20], and simulated annealing [21].

Metaheuristics has been used to solve many combinatorial optimization challenges, which include the job shop scheduling problem [22], fuzzy job-shop scheduling problems [23], the capacitated vehicle routing problem with soft time windows [24], the fleet size and mix vehicle routing problem [25,26], a combination of assignment and transportation problems [27], bicriterion transportation problems [28], transportation for handicapped persons [29], the robust capacitated vehicle routing problem [30], distributed reentrant permutation flow shop scheduling [31], cumulative capacitated vehicles [32], multi-stage logistic chain networks [33], supply chain management [34], the dynamic technician routing and scheduling problem [35], a combination of vehicle routing and scheduling problems with time window constraints [36,37], cyclic scheduling of a hoist with time window constraints [38], and generalized assignment problems [10]. Metaheuristics involves the methods of finding a good solution within a reasonable computational time. Such methods include adaptive memetic algorithms [39], variable neighborhood search [40,41], genetic algorithms [42], particle swam optimization [43], parallel route building algorithms [44], and ALNS [45]. Metaheuristics or the exact method that is addressed above can be used individually or in combination with several methods [46], so the efficiency of the solving method is improved. Blum et al. (2011) mention that there are many informative review articles about metaheuristics [47].

ALNS has been applied to solve many combinatorial optimizations, which include a selective and periodic inventory routing problem [48], a resource-constrained project scheduling problem [49], the pickup and delivery problem with transfers [50], and the pickup and delivery problem with time windows and scheduled lines [51]. ALNS is simple but effective. Generally, it is composed of four steps: (1) generate an initial solution; (2) perform a destroy method; (3) perform a repair method; and, (4) repeat Steps (2) and (3). The ALNS method searches for a wider variety of solutions (a larger search space) if compared with other types of research that use local search heuristics to improve solutions, With one iteration, ALNS can increase the solutions of interest up to 30–40% [51]. ALNS outperformed

many methods due to its use of fewer parameters, which means that it can be integrated or combined with other exact methods or heuristics [52], and it allows for the integration of special characteristics with methods, such as the special destroy and repair methods, as well as self-adaptive behavior [53].

From the literature mentioned above, we found that the solution of the ALNS algorithm was effective. It is important to consider the ALNS algorithms for solving scheduling and assignment problems. Although algorithms with search methods for side-by-side adjustments are not yet widely applied to solve such problems as much as they should be, many studies have shown good results in finding the best solutions and they have shown an ability to quickly solve problems. Consequently, the ALNS algorithm was selected to solve the scheduling and assignment problems for a broiler farm in Buriram Province.

3. Mathematical Formulation

A solution to multi-level scheduling and assignment problems for broiler farms can be formulated, as follows:

Indices

i	farm $i = 1, 2, \ldots, I$
j	production plant $j = 1, 2, \ldots, J$
t	the period of broiler-raising $t = 1, 2, \ldots, T$
k	truck capacity for broiler delivery $k = 1, 2, 3$

Parameters

I	number of farms
J	number of production plants
T	number of production periods
O	operational lead time
M	great number
A_j	profit from selling broilers to production plants (j) (Baht)
D_{ij}	distance from farm (i) to production plants (j) (km)
E_{jt}	maximum demand of plant (j) at time (t) (unit/week)
L_i	capacity of the farm i (unit/farm)
G_k	capacity of truck k (units)
N_{ik}	capacity of truck k to deliver the chickens from the farm (i)
B_k	fuel cost of vehicle k
H_k	rental cost of truck k (Baht/round)
F_{ik}	truck k is used to transport chickens from farm (i)
V_{ik}	transportation cost of a farm (i) using truck k

Decision Variables

X_{ijt}	amount of chickens delivered from farm i to plant j at time t
F_{ik}	$\begin{cases} 1 & if\ truck\ k\ is\ assigned\ to\ deliver\ chicken\ from\ i \\ 0 & therwise \end{cases}$
R_{ij}	number of rounds of truck that transport from farm i to production plant j
Y_{ijt}	$\begin{cases} 1 & if\ there\ is\ the\ delivery\ at\ tim\ t\ from\ farm\ i\ to\ plan\ j \\ 0 & therwise \end{cases}$

Objective Function

$$Miz\ Z = \sum_{t=1}^{T}\sum_{j=1}^{J}\sum_{i=1}^{I} A_j X_{ijt} - \sum_{t=1}^{T}\sum_{j=1}^{J}\sum_{i=1}^{I}\sum_{k=1}^{k} D_{ij} B_k F_{ik} Y_{ijt} R_{ij} - \sum_{k=1}^{K}\sum_{j=1}^{J}\sum_{i=1}^{I} H_k R_{ij} F_{ik} \quad (1)$$

Constraints

$$\sum_{i=1}^{I} X_{ijt} \leq E_{jt+O}, \forall j \in J, \forall t \in T \quad (2)$$

$$\sum_{j=1}^{J} \sum_{t=1}^{T} Y_{ijt} = 1, \forall i \in I \quad (3)$$

$$X_{ijt} \leq MY_{ijt}, \forall i \in I, \forall j \in J, \forall t \in T \quad (4)$$

$$\sum_{j=1}^{J} X_{ijt} \leq L_i, \forall i \in I, \forall t \in T \quad (5)$$

$$\frac{X_{ijt}}{N_{ik}} \leq R_{ij} F_{ik}, \forall k \in K, \forall i \in I, \forall j \in J, \forall t \in T \quad (6)$$

$$\sum_{k=1}^{K} F_{ik} = 1, \forall i \in I \quad (7)$$

The objectives of this model are as follows: Constraint (1) is the objective function to achieve the maximum operating profit, which is the total sales volume multiplied by the selling price of chickens to specified boiler production plants. Constraint (2) controls exist in every period; the farm cannot deliver more product than the production demand of plant j in the period $t + O$ when O is the lead time for producing the product. Constraint (3) is that each farm can have only one delivery to a production plant at any period of time. Constraint (4) confirms that the number of deliveries will be positive only when farm j is operated to deliver from farm i to production plant j at time t. Constraint (5) shows that farm i must produce less than its capacity. Therefore, the number of units does not exceed its capacity. Constraint (6) is the number of rounds of transportation in any period, which is equal to the number of chickens and have to be delivered from farm i to plant j, divided by the capacity of the assigned truck to serve farm i. Constraint (7) ensures that, for each farm i, only one truck can be assigned to it.

The mathematical model was tested with the exact method in the Lingo program, v.11. After that, the Lingo program test results were compared with the ALNS algorithm.

4. Adaptive Large Neighborhood Search Algorithm (ALNS)

Multi-level scheduling and assignment problems in order to plan production scheduling and assignments can be explained, as follows.

Generally, ALNS is composed of four steps: (1) generate an initial solution; (2) perform the destroy procedure; (3) perform the repair procedure; and, (4) update the required information. This can be explained in detail, as follows.

4.1. Generate Initial Solution

We firstly construct an initial solution while using Algorithm 1.

An example to construct a feasible solution and an initial solution while using the algorithm that is shown in Algorithm 1 is as follows: If we have seven farms and five periods of horizon planning, and then the production lead time (O) is two periods of time. There are two production plants that have a constant demand of 300 and 400 chicken units every period. The capacity of farms 1, 2, 3, 4, 5, 6, and 7 is 300, 200, 500, 500, 200, 400, and 450, respectively. The distance from Farms 1 to 7 to Plant 1 is 30, 50, 25, 50, 45, 60, and 90 km, respectively, while that to Plant 2 is 60, 60, 90, 40, 40, 30, 20, 40, and 30 km, respectively. If there are 12 trucks, then Trucks 1 to 12 have a capacity of 50, 25, 40, 50, 50, 20, 30, 50, 40, 40, 30, and 50, respectively. Assume that the order of the trucks is 12, 10, 5, 3, 1, 2, 8, 6, 9, 4, 7, and 11.

Algorithm 1: Creating the initial solution

Input: L_i, E_{jt},O = lead time
Step 1: Generate a vector to represent the problem; the vector has size I, where I is the number of farms.
Let this vector be W_i
Step 2: Sort W_i according to increasing order
Step 3 Set $t = 1$
Step 4: While (t<=T-O)
While $\sum_{i=1}^{I} X_{ijt} <= E_{jt+O}$
Assign farm i to produce chickens, and assign the farm according to the order of W_i

- $Li\ if\ \sum_{i=1}^{I} X_{ijt} + L_i \leq E_{jt+O}$
- $E_{jt+L} - \sum_{i=1}^{I} X_{ijt}\ if\ \sum_{i=1}^{I} X_{ijt} + L_i > E_{jt+O}$
- j is the closest production plant to the assigned farm. When the closet production plant is full, the next closest production plant is selected instead.

End $t = t + 1$;
End
Step 5: Randomly generate a vector to represent vehicle k. Let this vector be A_k.
Step 6: Sort vector A_k
Step 7: Assign the truck to the farm according to the order in A_k
Step 8: Calculate the total cost

If we generate a vector for a farm that has the size of seven positions, we will obtain {0.4, 0.3, 0.6, 0.9, 0.25, 0.57, 0.5} for farms 1, 2, 3, 4, 5, 6, and 7, respectively. After we sort the vector, we will obtain the order of the farm, as follows: {5(0.25), 2(0.3), 1(0.4), 7(0.5), 6(0.57), 3(0.6), and 4(0.9)}. We start to produce at Time 1 in Farm 5, which has a capacity of 200 chicken units, while Farm 5 will serve Plant 2, which has a demand of 400 units. Therefore, we can operate more farms in Period 1, which has to be Farm 2. Farm 2 has a capacity of 200 units and it needs to deliver to Plant 2 (which is the cooperating farm of Farm 5). The same mechanism will apply to all T-O periods (starting from Period 1). Table 1 shows the result of the assignment using Algorithm 1.

Table 1. Production planning and plant and truck assignment results while using Algorithm 1.

Period	1	2	3	4	5	Cap of Farm
Demand of Plant 1			300	300	300	
Demand of Plant 2			400	400	400	
Farm 1						300
Plant 1	300					
Plant 2						
Truck	12					
Farm 2						200
Plant 1						
Plant 2	200					
truck	10					
Farm 3						500
Plant 1		300				
Plant 2						
Truck		3				
Farm 4						500
Plant 1						
Plant 2			400			
Truck			8			

Table 1. *Cont.*

Period	1	2	3	4	5	Cap of Farm
Farm 5						200
Plant 1						
Plant 2	200					
Truck	5					
Farm 6						400
Plant 1		300				
Plant 2						
Truck		1				
Farm 7						450
Plant 1						
Plant 2		400				
Truck		2				

4.2. Destroy Operator

In this section, we will discuss the methods of the destroy operator, which is used to destroy the complete solution, such that it becomes incomplete. This method is used so that the solution moves to other search areas, and a new solution is thus obtained. Here, we present four methods: (1) random removal, (2) worst removal, (3) related removal, and (4) freeze zone. There are principles and steps of destruction in each method, as follows.

4.2.1. Random Removal

The principle of random removal destroy is shown in Algorithm 2.

Algorithm 2: Random removal

1. $B = I$; I = {Sequence of all farms}
2. L = {}
3. Q = Number to delete from 2 models;
4. While $|L| < Q$ do
   - 4.1 Random i from set B
   - 4.2 $L \leftarrow L \cup \{I\}$
5. Return to L

From Algorithm 2, we firstly make a list I, which is the list of all farms (B), and the Q farms will then be randomly selected to be deleted from list I.

4.2.2. Worst Removal

This method of destruction is carried out to remove the worst value from the set of the solution. Algorithm 3 shows the algorithm for worst removal.

The worse removal is more or less the same as random removal, but the difference is that list I is sorted according to profit in ascending order. The remaining step is the same as random removal.

Algorithm 3: Worst Removal

1. $B = I$; I = {Sequence of all farms}
2. L = {}
3. Q = Number to delete from 2 models;
4. $A = p_1, p_2, p_3, \ldots \ldots \ldots, p_n$
5. O^A = The order of farms according to profits in A (Sorted in ascending order)
6. While $|L| < Q$ do
   $p = |A| \, x^{\beta}$
   $i = O^A \, p$
   $L \leftarrow L \cup \{I\}$
7. Return to L

4.2.3. Related Removal

This method of destruction will destroy the value of the solution based on the location of the adjacent point that is related to the location of the point that has been removed. Algorithm 4 shows the algorithm.

Algorithm 4: Related Removal

$B = I$; I = {Sequence of all farms}
Q = Number to delete from 2 models;
While $|L| < Q$ do

a. Random farm i in L
b. O^B is order list of farm

Based on equation $R(c,i)$, $i \in$ B

c. Random x from 0 to 1
d. $p = |A| \, x^{\beta}$ and $r = O^B \, p$
e. Remove r from B and

$L \leftarrow L \cup \{I\}$
2. Return to L

The related removal is more or less the same as the random removal, but the difference is that list I is sorted according to $R(c,i)$. The remaining step is the same as the random removal.

4.2.4. Freeze Zone

The freeze zone is inspired by the idea that the current solution moves in the right direction to obtain the optimal solution (best solution). The solution should only remove some parts. Therefore, the random walk search does not occur in the current solution. This method is the destruction of the solution by considering the location of the point that needs to be destroyed within the scope of the area that has been defined. Figure 2 shows the characteristics of destruction.

Figure 2 represents the characteristics of the freeze zone. We determined the procedure for the freeze zone destruction solution, as follows.

Step 1: Divide the farm into four zones (half divided by the latitude and longitude of the farm).

Step 2: Randomly select the value of d, which has a value from 1 to 3 (the number of zones that are not in the freeze zone).

Step 3: Randomly select the d zone from the four zones that were established in Step 1.

Step 4: Randomly select the destroy method for use in removing the entities in the selected zone (random removal/worst removal/related removal).

Step 5: Perform the selected removal method in the zone for the selected zone obtained in Step 3.

We divided all of the farms in Figure 2 into four zones. Therefore, we have Zones 1, 2, 3, and 4. In Step 2, the number of zones that are not frozen will be selected if we randomly selected d as being equal to two zones. For Step 3, two out of the four zones will be selected if we select Zones 1 and 3 as the zones that are not frozen, after which Steps 4 and 5 will be executed with the farm in Zones 1 and 3. Steps 4 and 5 will be executed while using the procedure explained in Section 4.3.1, Section 4.3.2, and Section 4.3.3

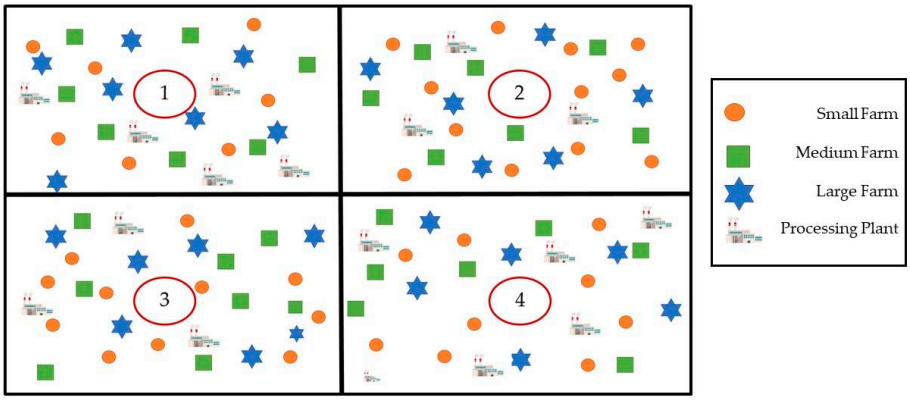

Figure 2. Characteristics of the freeze zone.

4.3. Repairing Operation

After we destroy the complete solution into an incomplete solution while using the destroy operator, the repair operator is designed to obtain the complete solution again. Here, we present three repair operators: (1) greedy insertion, (2) random insertion, and (3) swap. The principles that were used to select the repair method are as follows.

4.3.1. Greedy Insertion

Greedy insertion is a method that is used to repair the solution by finding the lowest cost variant and re-inserting the best point, which will provide a new solution. Algorithm 5 shows the greedy insertion algorithm.

Algorithm 5: Greedy Insertion

1. $B = L$
2. While $|B| > 0$ do
   - 2.1 Sort the items in B, by profit from descending farm i, which can insert the best feasible plants in the solution
   - 2.2 Random x from 0 to 1
   - 2.3 $p = | O^B | x^\beta$ and $i = O^B p$
   - 2.4 Insert farm i at the best solution plant

4.3.2. Random Insertion

This method is easy and uncomplicated. The principle of random insertion is to repair the relationship of variables that will be inserted into any position, regardless of cost. Algorithm 6 shows the random insertion algorithm.

Algorithm 6: Random Insertion

1. $B = L$
2. While $|B| > 0$ do

    2.1 Random farm i in set B
    2.2 Random time period T
    2.3 Random plant j in time period T
    2.4 If farm feasible to add to plant j

Remove farm i in set B
Add farm i to plant j

4.3.3. Swap

The swap is easy and uncomplicated. The principle of the swap method that we used in this work involved selecting the solution in which the selected is swapped. The steps are as follows:

Step 1: Randomly select 2–3 farms.

Step 2: Mark the location of 2–3 farms (marking is to specify the location of the farm randomly).

Step 3: Swap the position of the farm from the point that has been marked.

Based on the two steps to repair the solutions, swap the solution to obtain a better solution.

4.4. Solution Acceptance

Solution acceptance is used in the case of a new generated solution when we finish the destroy and repair methods. This solution is worse than the original solution. We call it the solution acceptance criteria. We use three equations in this study, as follows.

4.4.1. Simulated Annealing (SA)

Equation (8) is motivated by simulated annealing (a popular metaheuristic method). The acceptance formula using information from the current best solution and the current solution to derive the formula is shown in Equation (8):

$$\mathrm{p} = \exp^{-\frac{(\mathrm{Z}(\mathrm{V}\prime)-\mathrm{Z}(\mathrm{V}))}{\mathrm{T}\times\mathrm{K}}} \tag{8}$$

When *Z(V)* is the best current solution and *Z(V')* is the new generated solution. *T* and *K* are the predefined parameters.

4.4.2. Linear Function

We believe that the current iteration should play an important role in accepting or rejecting the solution. Normally, we will allow the algorithm to search the whole area of the search space (diversification). The algorithm should have a way to escape from the local optimal when the it is trapped there. Therefore, information regarding the current number of iterations has been added to Equation (8) to obtain Equation (9).

$$\mathrm{p} = 1 - \exp^{-\left[\left(\frac{\mathrm{Z}(\mathrm{V})-\mathrm{Z}(\mathrm{V}\prime)}{\mathrm{Z}(\mathrm{V})}\right)^2 + \left(It-\frac{MaxIt}{2}\right)^2\right]} \tag{9}$$

where *MaxIT* is the maximum number of iterations and *It* is the number of current iterations.

4.4.3. Exponential Function of the Current Number of Iteration

Equation (10) is used to test, when only using the current number of iterations as the given information, how it will change the performance of the algorithm.

$$p = 1 - \exp^{-\left[\left(It - \frac{MaxIt}{2}\right)^2\right]} \quad (10)$$

The adaptive large neighborhood search used to solve the proposed problem can be concluded in Algorithm 7.

Algorithm 7: Adaptive Large Neighborhood

Begin
Construct the initial solution
While the number of iterations ≤ maximum number of iterations
Randomly select the destroy method
(freeze zone/related removal/worst removal/random removal)
Perform the selected destroy method
Randomly select the repair method
(greedy insertion/random insertion/swap)
Perform the selected repair method
Update heuristic information
(solution acceptance/simulated annealing/linear function/exponential function of the current number of iterations)
End (while)
End (Begin)

From Algorithm 7, the proposed method starts from generating the initial solution, after which the iterative search is executed. The search can be performed while using the destroy and repair operators. During the search, heuristics information, such as the current solution and the weight to select the destroy and repair method, will be updated.

5. Computation Framework and Results

In this research, we designed three groups of problems to test the instances. Problem Group 1 is a small problem that requires 20–30 farms and 2–3 production plants. Problem Group 2 is a medium problem that requires 60–80 farms and 3–5 production plants. Problem Group 3 is a large problem that requires 90–100 farms and 6–9 production plants. The framework of the research is shown in Table 2, as below.

Three ALNS algorithms were tested: (1) ALNS-1, which is ALNS using Equation 8 to determine the probability of obtaining the worse solution to use for the next iteration; (2) ALNS-2 (using Equation (9)); and, (3) ALNS-3 (using Equation (10)). All of the proposed methods were executed five times. Table 3 depicts the best solution among five runs. In each simulation, the number of iterations, 1000, was set to be the stopping criteria. The small size of the test instances for the proposed methods was compared with the optimal solution that is generated by Lingo v.11, while the medium and large size test instances for the proposed algorithms were compared with the bound that was generated by Lingo v.11 within 72 h (or 4320 min).

Table 2. Details of the test instances.

Size of Problem	Number of Farms	Number of Production Plants
S1	20	1
S2	20	2
S3	25	2
S4	30	3
S5	30	3
M1	60	3
M2	60	4
M3	70	4
M4	80	4
M5	80	5
L1	90	6
L2	90	7
L3	100	7
L4	100	8
L5	100	9

We further analyzed the data given in Table 2 by calculating the percentage difference in the results that were obtained from the optimal or bound solution of Lingo v. 11 and the proposed methods. The percentage difference was calculated while using Equation (11):

$$\%\, diff = \frac{S^L - S^A}{S^L} \times 100\% \tag{11}$$

where S^L is the solution that was obtained from Lingo v.11, S^A is the solution gained from the proposed methods, and is the percentage difference of the two solution approaches. Table 4 shows the percentage difference.

Table 3. Lingo sample test results with the Adaptive Large Neighborhood Search (ALNS) method as compared to the Lingo program.

Size of Problems	Lingo v.11			ALNS1		ALNS2		ALNS3	
	Status	Profit (Baht)	Time Minutes	Profit (Baht)	Time Minutes	Profit (Baht)	Time Minutes	Profit (Baht)	Time
S1	Best Obj	14,798,366.5	7.30	**14,798,366.5**	5.01	**14,798,366.5**	5.40	**14,798,366.5**	4.13
S2	Best Obj	14,819,077.5	7.58	**14,819,077.5**	5.10	**14,819,077.5**	5.16	**14,819,077.5**	5.22
S3	Best Obj	17,745,366.5	8.54	**17,745,366.5**	6.20	**17,745,366.5**	6.53	**17,745,366.5**	6.40
S4	Best Obj	18,380,441.9	4320.00	**18,380,441.9**	5.30	**18,380,441.9**	6.14	18,379,827.8	6.07
S5	Best Obj	18,388,258.0	4320.00	**18,388,258**	6.10	**18,388,258**	7.15	18,387,644	7.45
M1	Obj Bound	48,164,507.9	4320.00	**47,197,076.9**	8.20	**47,207,076.9**	89.00	47,206,755	72.00
M2	Obj Bound	49,643,809.4	4320.00	48,746,050	85.10	**48,747,477.8**	85.20	48,741,329.9	89.00
M3	Obj Bound	54,758,312.0	4320.00	53,492,055.1	94.40	**53,497,814.6**	107.40	53,489,316.5	108.00
M4	Obj Bound	55,214,823.6	4320.00	54,459,997.8	111.20	**54,464,756.7**	117.80	54,428,783.2	124.00
M5	Obj Bound	56,509,277.4	4320.00	54,739,416.6	136.30	**54,749,040.7**	292.00	54,743,931.2	161.00
L1	Obj Bound	56,135,242.7	4320.00	54,636,399.3	300.10	**54,652,447.3**	324.50	54,646,561.6	321.00
L2	Obj Bound	55,993,528.0	4320.00	54,936,334.1	312.40	**54,959,383.3**	340.24	54,938,904.7	334.00
L3	Obj Bound	57,188,015.2	4320.00	55,817,146.2	379.20	**55,900,470.9**	405.20	55,897,309.9	382.00
L4	Obj Bound	58,558,560.1	4320.00	56,555,984.8	420.00	**56,565,124.3**	440.20	56,541,695.8	429.00
L5	Obj Bound	58,056,520.6	4320.00	56,898,278.7	432.40	**56,898,446.7**	462.10	56,888,534.3	474.00

Table 4. The percentage difference of Lingo v.11 and the proposed methods.

#Instance Number	Lingo v.11 Result	ALNS-1	ALNS-2	ALNS-3
S1	14,798,366.53	**0**	**0**	**0**
S2	14,819,077.51	**0**	**0**	**0**
S3	17,745,366.48	**0**	**0**	**0**
S4	18,380,441.85	**0**	**0**	0.003341
S5	18,388,257.98	**0**	**0**	0.003339
Small Size Instance Average		0		0
M1	47,197,076.9	2.01	**1.99**	1.99
M2	48,746,050	1.81	**1.81**	1.82
M3	53,492,055.1	2.31	**2.30**	2.32
M4	54,459,997.8	1.37	**1.36**	1.42
M5	54,739,416.6	3.13	**3.11**	3.12
Medium Size Instance Average		2.13		2.11
L1	54,636,399.3	2.67	**2.64**	2.65
L2	54,936,334.1	1.89	**1.85**	1.88
L3	55,817,146.2	2.40	**2.25**	2.26
L4	56,555,984.8	3.42	**3.40**	3.44
L5	56,898,278.7	2.00	**1.99**	2.01
Large Size Instance Average		2.48	**2.43**	2.45
Overall Average		1.52	**1.51**	1.53

From the results shown in Table 4, the proposed methods have a 0.00–2.43% average difference from that of the optimal solution, which is quite close. The performance of the proposed methods is still good, even though the problem size is higher. The statistical test was verified while using a two-tailed Wilcoxon sign rank test [54]. We wanted to conclude that the results that were obtained from Lingo v.11 were different from those of the proposed methods. Table 5 shows the results.

Table 5. *p*-value of the Wilcoxon sign rank significance test.

Algorithm	ALNS-1	ALNS-2	ALNS-3
Lingo v.11	0.00512	**0.00222**	0.00512
ALNS-1		**0.0256**	0.46728
ALNS-2			**0.00222**

From Table 5, we can conclude that all of the proposed heuristics are statistically and significantly worse than the optimal solution or upper bound obtained from Lingo v.11. Lingo v.11 can find a better solution when compared with the proposed methods, but it requires 93% higher computational time. The most feasible solution is not guaranteed for the upper bound that was obtained from Lingo v.11 in the medium and large size test instances. Therefore, the proposed methods can find an average that is 1.51–1.53% worse than that of the optimization software, while using 93% less computational time, which is suitable for real-world applications.

When comparing between all the proposed methods, ALNS-2, while using Equation (9) as the acceptance criteria, performs significantly better than the other proposed methods. Equation (9) is the combination of Equations (8) and (10). The advantages of Equations (8) and (10) make it the best proposed method. Figure 3 presents a graph that reports the best known solutions from ALNS-1 to ALNS-3.

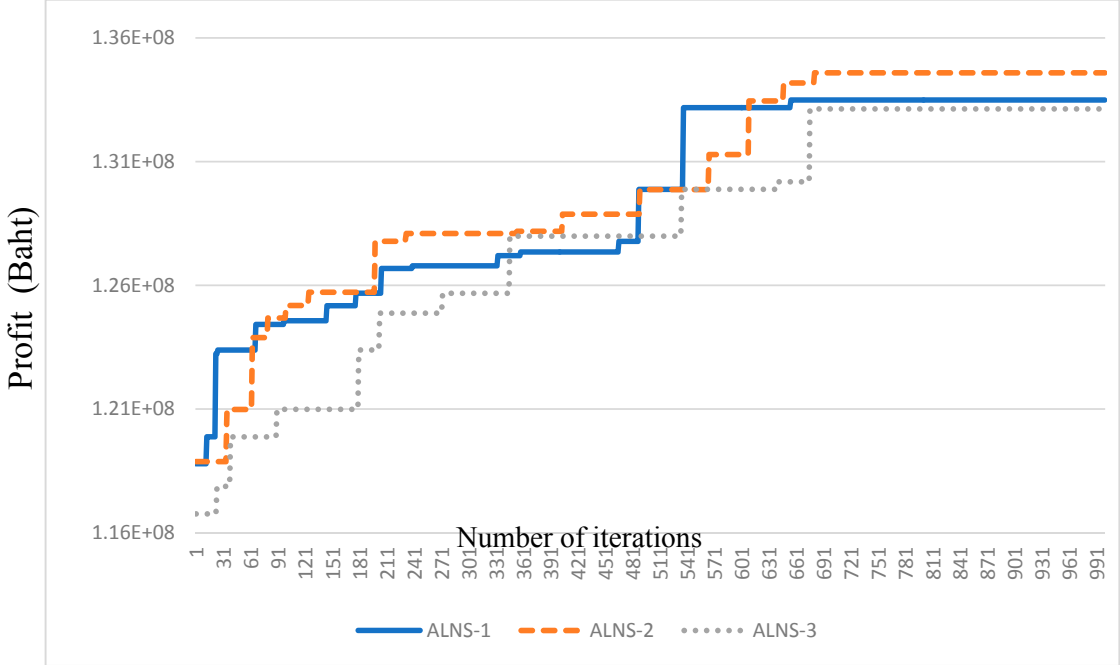

Figure 3. The best known solution plot during the simulation run of the case study.

From Figure 3, we can conclude that ALNS-1 can quite often find the best known solution when the X-axis represents the current number of iterations that the simulation is executing (from 1 to 1000) and when the Y-axis shows the current best solution found (profit) during the simulation run. However, the solution is not moved and it is harder to find the new best solution when it is trapped on the local optimal. Meanwhile, ALNS-3 uses acceptance criteria depending only on the number of iterations. The solution only changes in the first part of the simulation and it is quite quickly trapped on the local optimal. ALNS-2 is the combination of these two formulas, so it finds the new best solution often and it is not trapped on the local optimal. This algorithm is still a better solution, even in the last part of the simulation, until it is in the mature stage, after which the search is terminated.

The case study comprises nine production plants and 177 broiler farms under the time frame used for the delivery of up to 6 h per delivery. Table 6 shows the computational results, as below.

Table 6. Results of solving problems for the case study using the ALNS algorithm.

Period	Broiler Quantity	Type of Car			Income (Baht)	Contracting Cost (Baht)	Transportation Cost (Baht)
		4 Wheels	6 Wheels	10 Wheels			
1	3,065,900	0	10	308	21,857,325	315,000	126,727.27
2	3,143,300	0	2	317	22,379,775	3,185,000	140,364.46
3	3,194,000	0	11	321	22,722,000	3,292,500	118,319.98
4	3,126,380	0	15	308	22,265,285	3,192,500	130,880.26
5	3,194,610	0	15	316	22,725,743	3,277,500	113,011.20
6	3,182,480	0	21	312	22,643,680	3,227,500	127,111.75
Total	18,906,670	0	74	1882	134,593,808	19,375,000	756,414.91

Table 6 shows the results of solving the case study by applying the ALNS algorithm. It was found that the farms were able to efficiently allocate the amount of broilers that were delivered to all nine production plants with a total of 18,906,670 broilers from farms delivered to production plants in the scheduling and assignment for six periods of time (1 period = 1 week). Farms did not use four-wheeled trucks. Six-wheeled trucks and 1882 10-wheeled trucks were used. The total operating income was 134,593,807.50 baht. The cost of hiring the trucks was 19,375,000 baht. The total delivery costs amounted to 756,414.91 baht. This indicates an operating profit of 114,462,393.09 baht/month.

When considering the profits from traditional operations, the profit was 107,886,904.69 baht/month. It was found that the profits from the ALNS algorithm resulted in the farm's having a higher operating profit, up to 6575,488.40 baht/month, or a 5.74% increase when comparing the original profits and net profits arising from the application of the ALNS algorithm. Therefore, it can be concluded that the method of ALNS is effective and it can help farms to obtain maximum profit from their operation.

6. Conclusions and Future Research

This paper examined multi-level scheduling and assignment problems in broiler production plants. The goal was to maximize the total profit of operations by more efficiently assigning tasks in the farms, allocating chicken quantity and more effectively, and suitably selecting the size of truck that is to be used to deliver the broilers.

We formulated a mathematical model to find the solution while using the Lingo program (Lingo v.11). ALNS-1 and ALNS-2 could find a 100% optimal solution for the small problem group, while ALNS-3 could only find a 60% optimal solution because ALNS-3 used Equation (10) as acceptance criteria and led to a worse solution. The acceptance of a worse solution for operation in the next iteration only depends on the number of iterations, meaning that it is more easily trapped in the local optimal, while other acceptance also depends on the solution quality, which is significantly higher than that of ALNS-3.

For the medium and large size test instances, the proposed method could find a solution that was 2.11 to 2.48% worse than that of Lingo v.11, but this method required much less computational time (93%). This means the proposed method can generate an acceptable solution quality within an acceptable time.

When comparing between all of the proposed methods, the method that considers solution quality and the number of iterations (ALNS-2) generates a better solution than the algorithm that considers only solution quality (ALNS-1) or the number of iterations (ALNS-3) does. The proposed method was used to solve a case study and the results show that it could generate more profit than the current method of the firm by 5.74%.

Therefore, this study presents an appropriate and effective method to solve the problems of broiler farms, which are important businesses in export, and therefore Thailand's economy. The type of problem that is addressed in this paper is widespread throughout real business fields, including logistics, marketing, and production. Therefore, the proposed algorithm can be used by firms facing similar challenges.

Author Contributions: Conceptualization, R.P.; Methodology, N.P. and R.P.; Project administration, K.S. and C.T.; Validation, N.P. and R.P.; Writing—original draft, N.P.; Writing—review and editing, M.K.-O. and C.T.; Supervision, S.K.

Funding: This research received no external funding.

Acknowledgments: The authors wish to thank Ubonratchathani University for providing us with the facilities to perform the relevant simulations.

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

References

1. Wareerat, P. Frozen and Processed Chicken Industry. Thailand Industry Outlook 2017–2019. June 2017. Available online: https://www.krungsri.com/bank/.../IO_Chicken_171019_EN_EX.aspx (accessed on 13 June 2018).
2. Srivarapongse, T.; Pijitbanjong, P. Solving a Special Case of the Generalized Assignment Problem Using the Modified Differential Evolution Algorithms: A Case Study in Sugarcane Harvesting. *J. Open Innov. Technol. Mark. Complex.* **2019**, *5*, 5. [CrossRef]
3. Lokhorst, C.; Lamaker, E.J.J. An expert system for monitoring the daily production process in aviary systems for laying hens. *Comput. Electron. Agric.* **1996**, *15*, 215–231. [CrossRef]

4. Patel, V.C.; McClendon, R.W.; Goodrum, J.W. Development and evaluation of an expert system for egg sorting. *Comput. Electron. Agric.* **1998**, *20*, 97–116. [CrossRef]
5. Mertens, K.; Vaesen, I.; Löffel, J.; Ostyn, B.; Kemps, B.; Kamers, B.; Bamelis, F.; Zoons, J.; Darius, P.; Decuypere, E.; et al. Data-based design of an intelligent control chart for the daily monitoring of the average egg weight. *Comput. Electron. Agric.* **2008**, *61*, 222–232. [CrossRef]
6. Mohaddes, S.A. Productivity Analysis of Eggs Production in Khorasan Razavi Province. Iran. *Int. J. Poult. Sci.* **2009**, *8*, 1209–2113. [CrossRef]
7. Demircan, V.; Yilmaz, H.; Dernek, Z.; Bal, T.; Gul, M.; Koknaroglu, H. Economic analysis of different laying hen farm capacities in Turkey. *Agric. Econ.* **2010**, *56*, 489–497. [CrossRef]
8. Liu, L.; Mu, H.; Song, Y.; Luo, H.; Li, X.; Wu, F. The equilibrium generalized assignment problem and genetic algorithm. *Appl. Math. Comput.* **2012**, *218*, 6526–6535. [CrossRef]
9. Sethanan, K.; Pitakaso, R. Differential evolution algorithms for scheduling raw milk transportation. *Comput. Electron. Agric.* **2016**, *121*, 245–259. [CrossRef]
10. Sethanan, K.; Pitakaso, R. Improved differential evolution algorithms for solving generalized assignment problem. *Expert Syst. Appl.* **2016**, *45*, 450–459. [CrossRef]
11. Thongdee, T.; Pitakaso, R. Differential evolution algorithms solving a multi-objective, source and stage location-allocation problem. *Ind. Eng. Manag. Syst.* **2015**, *14*, 11–21. [CrossRef]
12. Lourenço, H.R.; Serra, D. *Adaptive Approach Heuristics for the Generalized Assignment Problem*; Universitat Pompeu Fabra: Barcelona, Spain, 1998.
13. Ross, G.T.; Soland, R.M. A branch and bound algorithm for the generalized assignment problem. *Math. Program. Comput.* **1975**, *8*, 91–103. [CrossRef]
14. Fisher, M.L.; Jaikumar, R. A generalized assignment heuristic for vehicle routing. *Networks* **1981**, *11*, 109–124. [CrossRef]
15. Pardalos, L.; Resende, M. A greedy randomized adaptive search procedure for the quadratic assignment problem. *Quadratic Assign. Relat. Probl. Dimacs Ser. Discret. Math. Theor. Comput. Sci.* **1994**, *16*, 237–261.
16. Osorio, M.A.; Laguna, M. Logic cuts for multilevel generalized assignment problems. *Eur. J. Oper. Res.* **2003**, *151*, 238–246. [CrossRef]
17. Nežana, M.M.; Abraham, P.P. Local search intensified: Very large-scale variable neighborhood search for the multi-resource generalized assignment problem. *Discret. Optim.* **2009**, *6*, 370–377. [CrossRef]
18. Kuhn, H.W. The Hungarian method for the assignment problem. *Nav. Res. Logist. Q.* **1955**, *2*, 83–97. [CrossRef]
19. Kuenne, R.E.; Soland, R.M. Exact and approximate solutions to the multisource Weber problem. *Math. Program.* **1972**, *3*, 193–209. [CrossRef]
20. Posta, M.; Ferland, J.A. An exact method with variable fixing for solving the generalized assignment problem. *Comput. Optim. Appl.* **2012**, *52*, 629. [CrossRef]
21. Murray, A.T.; Church, R.L. Applying simulated annealing to location-planning models. *J. Heuristics* **1996**, *2*, 31–53. [CrossRef]
22. Bhatt, N.; Chauhan, N.R. Genetic algorithm applications on Job Shop Scheduling Problem: A review. In Proceedings of the 2015 International Conference on Soft Computing Techniques and Implementations (ICSCTI), Faridabad, India, 8–10 October 2015; pp. 7–14.
23. Alwani, A.; Majid, A.N. Fuzzy job-shop scheduling problems. A review. *Inf. Sci.* **2014**, *278*, 380–407. [CrossRef]
24. Taillard, É.; Badeau, P.; Gendreau, M.; Guertin, F.; Potvin, J.-Y. A tabu search heuristic for the vehicle routing problem with soft time windows. *J. Transp. Sci.* **1997**, *31*, 170–186. [CrossRef]
25. Repoussis, P.P.; Tarantilis, C.D. Solving the fleet size and mix vehicle routing problem with time windows via adaptive memory programming. *Transp. Res. Part C Emerg. Technol.* **2010**, *18*, 695–712. [CrossRef]
26. Liu, F.-H.; Shen, S.-Y. The fleet size and mix vehicle routing problem with time windows. *Eur. J. Inf. Syst.* **1999**, *50*, 721–732.
27. Munkres, J. Algorithms for the assignment and transportation problems. *SIAM J. Optim.* **1957**, *5*, 32–38. [CrossRef]
28. Aneja, Y.P.; Nair, K.P. Bicriteria transportation problem. *Manag. Sci.* **1979**, *25*, 73–78. [CrossRef]
29. Toth, P.; Vigo, D. Heuristic algorithms for the handicapped persons transportation problem. *J. Transp. Sci.* **1997**, *31*, 60–71. [CrossRef]

30. Gounaris, C.E.; Repoussis, P.P.; Tarantilis, C.D.; Wiesemann, W.; Floudas, C.A. An adaptive memory programming framework for the robust capacitated vehicle routing problem. *J. Transp. Sci.* **2014**, *50*, 1239–1260. [CrossRef]
31. Rifai, A.P.; Nguyen, H.-T.; Dawal, S.Z.M. Multi-objective adaptive large neighborhood search for distributed reentrant permutation flow shop scheduling. *Appl. Soft Comput.* **2016**, *40*, 42–57. [CrossRef]
32. Ribeiro, G.M.; Laporte, G. An adaptive large neighborhood search heuristic for the cumulative capacitated vehicle routing problem. *Comput. Oper. Res.* **2012**, *39*, 728–735. [CrossRef]
33. Syarif, A.; Yun, Y.; Gen, M. Study on multi-stage logistic chain network: A spanning tree-based genetic algorithm approach. *Comput. Ind. Eng.* **2002**, *43*, 299–314. [CrossRef]
34. Griffis, S.E.; Bell, J.E.; Closs, D.J. Metaheuristics in logistics and supply chain management. *J. Bus. Logist.* **2012**, *33*, 90–106. [CrossRef]
35. Pillac, V.; Guéret, C.; Medaglia, A. *On the Dynamic Technician Routing and Scheduling Problem*; Ecole des Mines de Nantes: Nantes, France, 2012; pp. 509–512.
36. Solomon, M.M. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Oper. Res.* **1987**, *35*, 254–265. [CrossRef]
37. Solomon, M.M.; Desrosiers, J. Survey paper—Time window constrained routing and scheduling problems. *J. Transp. Sci.* **1988**, *22*, 1–13. [CrossRef]
38. Chen, H.; Chu, C.; Proth, J.-M. Cyclic scheduling of a hoist with time window constraints. *IEEE Trans. Robot. Autom.* **1998**, *14*, 144–152. [CrossRef]
39. Nalepa, J.; Cwiek, M.; Kawulok, M. Adaptive memetic algorithm for the job shop scheduling problem. In Proceedings of the 2015 International Joint Conference on Neural Networks (IJCNN), Killarney, Ireland, 12–17 July 2015; pp. 1–8.
40. Hansen, P.; Mladenović, N. Variable neighborhood search: Principles and applications. *Eur. J. Oper. Res.* **2001**, *130*, 449–467. [CrossRef]
41. Mladenović, N.; Hansen, P. Variable neighborhood search. *Comput. Oper. Res.* **1997**, *24*, 1097–1100. [CrossRef]
42. Altiparmak, F.; Gen, M.; Lin, L.; Paksoy, T. A genetic algorithm approach for multi-objective optimization of supply chain networks. *Comput. Ind. Eng.* **2006**, *51*, 196–215. [CrossRef]
43. Salman, A.; Ahmad, I.; Al-Madani, S. Particle swarm optimization for task assignment problem. *Microprocess. Microsyst.* **2002**, *26*, 363–371. [CrossRef]
44. Potvin, J.-Y.; Rousseau, J.-M. A parallel route building algorithm for the vehicle routing and scheduling problem with time windows. *Eur. J. Oper. Res.* **1993**, *66*, 331–340. [CrossRef]
45. Kovacs, A.A.; Parragh, S.N.; Doerner, K.F.; Hartl, R.F. Adaptive large neighborhood search for service technician routing and scheduling problems. *J. Sched.* **2012**, *15*, 579–600. [CrossRef]
46. D'Andreagiovanni, F.; Mett, F.; Nardin, A.; Pulaj, J. Integrating LP-guided variable fixing with MIP heuristics in the robust design of hybrid wired-wireless FTTx access networks. *Appl. Soft Comput.* **2017**, *61*, 1074–1087. [CrossRef]
47. Blum, C.; Puchinger, J.; Jakob, G.; Raidl, G.; Günther, A.; Roli, A. Hybrid metaheuristics in combinatorial optimization: A survey. *Appl. Soft Comput.* **2011**, *11*, 4135–4151. [CrossRef]
48. Aksen, D.; Kaya, O.; Salman, F.S.; Tüncel, Ö. An adaptive large neighborhood search algorithm for a selective and periodic inventory routing problem. *Eur. J. Oper. Res.* **2014**, *239*, 413–426. [CrossRef]
49. Muller, L.F. An adaptive large neighborhood search algorithm for the resource-constrained project scheduling problem. In Proceedings of the MIC 2009: The VIII Metaheuristics International Conference, Hamburg, Germany, 13–16 July 2009.
50. Masson, R.; Lehuédé, F.; Péton, O. An adaptive large neighborhood search for the pickup and delivery problem with transfers. *J. Transp. Sci.* **2013**, *47*, 344–355. [CrossRef]
51. Ghilas, V.; Demir, E.; Van Woensel, T. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows and scheduled lines. *Comput. Oper. Res.* **2016**, *72*, 12–30. [CrossRef]
52. Parragh, S.N.; Cordeau, J.-F. Branch-and-price and adaptive large neighborhood search for the truck and trailer routing problem with time windows. *Comput. Oper. Res.* **2017**, *83*, 28–44. [CrossRef]

53. Wang, P.; Reinelt, G.; Tan, Y. Self-adaptive large neighborhood search algorithm for parallel machine scheduling problems. *IEEE Syst. J.* **2012**, *23*, 208–215. [CrossRef]
54. Social Science Statistics. The Wilcoxon Signed-Ranks Test Calculator. 2017. Available online: https://www.socscistatistics.com/tests/signedranks/default.aspx (accessed on 13 June 2018).