Proceedings of the 2022 Winter Simulation Conference
B. Feng, G. Pedrielli, Y. Peng, S. Shashaani, E. Song, C. G. Corlu, L. H. Lee, E. P. Chew, T. Roeder and P. Lendermann, eds.

AN ADAPTIVE LARGE NEIGHBORHOOD SEARCH ALGORITHM FOR WIND FARM INSPECTION USING A TRUCK WITH A DRONE

Wenyu Tao
Xinjia Jiang
Dongqiang Zhao

College of Economics and Management
Nanjing University of Aeronautics and Astronautics
29 Jiangjun Road
Nanjing, 211106, CHINA

ABSTRACT

With the increasing demand for energy, wind power as a new energy source has been widely used and developed on a large scale. To extend the life of wind turbines, it is necessary but difficult to carry out regular inspections in wind farms located in remote areas. This paper studied the clustering and routing problem of truck-drone joint inspection of wind farms. An Adaptive Large Neighborhood Search (ALNS) algorithm is designed based on the characteristics of this problem. In addition, wind farm instances with different sizes and distributions are generated in this paper to simulate realistic scenes and evaluate ALNS. Finally, real wind farm instances are tested to demonstrate the inspection time in detail. Computational experiments show ALNS can improve significantly inspection time compared with another method.

1 INTRODUCTION

Wind power is one of the most popular forms of renewable energy and grows in popularity in recent years. It has a smaller impact on the environment than other traditional ways of generating electricity. A wind farm usually occupies hundreds of square kilometers and is distributed over remote areas. With damage to wind turbines being unavoidable, it is necessary to inspect them regularly. At present, the traditional inspecting method is non-destructive testing (NDT), which is usually done manually (Juengert 2008). The industrial climber needs to inspect the blades roping from the rotor of a wind turbine. This presents a safety concern and requires a long time for manual inspection. Therefore, a lot of costs are incurred. Traditional inspecting methods cannot adapt to the development of the times, and it is necessary to find more efficient and safe inspecting methods.

With the development of drone technology, using drones to inspect wind turbines is considered an economic and efficient way. SkySpecs, a Michigan-based startup, can use a drone to inspect a wind turbine in 15 minutes, covering 17 wind turbines per day (Baik and Valenzuela 2020).

However, a drone cannot inspect a whole wind farm due to its endurance limit. For this reason, a joint inspection system of wind farms using a truck and a drone is proposed. The truck carrying a drone can be parked at a certain location. When the truck stops, the drone flies to inspect several wind turbines and then returns to the truck. The operator drives the truck to the next parking location until all wind turbines are inspected.

The technical challenges of using drones to inspect wind turbines have been addressed in several papers. To overcome the impact of high wind gusts on the drone, Domínguez et al. (2017) proposed an effective adaptive speed control method. Wang and Zhang (2017) proposed a data-driven framework to automatically

detect wind turbines blade surface cracks based on images taken by a drone. Cao et al. (2019) proposed a mobile edge computing (MEC) driven UAV routine inspection system to inspect wind turbines blades. However, these papers only consider that a single drone inspects a single wind turbine.

The combination of a truck and a drone has been used in the logistics field. Murray and Chu (2015) were the first to introduce the flying sidekick traveling salesman problem (FSTSP) in which a truck carrying a drone delivers parcels to minimize the total operation time. They formulated a MILP model and proposed a heuristic that adopts the "truck first, drone second" idea. Many heuristic and meta-heuristic algorithms have been proposed for the FSTSP problem so far, there are simulated annealing algorithm (SA) (Ponza 2016), greedy randomized adaptive search procedure (GRASP) (Ha et al. 2018), hybrid genetic algorithm (HGA) (Ha et al. 2019), etc. Traveling salesman problem with drone (TSP-D) is similar to FSTSP and was first introduced by Agatz et al. (2018). The difference is that the drone may be launched and returned to the same location. The two vehicles (truck and drone) also share the same road network. For the TSP-D and its variety, existing studies have focused on exact algorithms (Bouman et al. 2018; Poikonen et al. 2019; Roberti and Ruthmair 2021) and heuristic algorithms (Jeong et al. 2019; Marinelli et al. 2018). In most parcel delivery problems (e.g. FSTSP), the drone is allowed to visit only one node which is different from our research.

In the field of surveillance, truck and drone jointly used for tasks have also been researched. Savuran and Karakaya (2015) considered that given a route of a carrier, a drone is launched from the carrier to visit the targets and received at the other point aimed at minimizing the total inspecting time. Luo et al. (2017) proposed the two-echelon vehicle cooperative problem in which the truck could not serve the customers and worked only as a mobile satellite for the drone. Baik and Valenzuela (2020) provided a routing optimization model to reduce the total operation time for inspecting a wind farm. They provided a method in which wind turbines were clustered in k-means clustering algorithm and the drone routing was optimized in each cluster by solving the TSP. Then they optimized the truck routing by solving the equality generalized traveling salesman problem (G-TSP) using an integer linear programming model.

From the theoretical perspective, similar research has been conducted to solve the truck and trailer routing problem (TTRP). In TTRP, vehicle composed of a truck with a detachable trailer serves the demand of a set of customers reachable by truck and trailer or only by the truck. A "matheuristic" approach (Villegas et al. 2011) and a branch-and-cut algorithm (Drexl et al. 2014) have been proposed to solve the problem. The difference between TTRP and this study is that all points can be accessed by the truck or the drone, while the truck with a trailer cannot serve customers reachable only by the truck in TTRP.

This study studies the optimization problem of the truck-drone joint inspection system and the simulation of wind farms. We design an adaptive large neighborhood search (ALNS) algorithm to solve the truck-drone joint inspection problem. Moreover, we test ALNS on the generated instance and compared it with the method proposed by Baik and Valenzuela (2020). The generated instances consider the distribution and size to simulate the realistic scene. Three real wind farms are also tested to show the detailed results of ALNS.

2 PROBLEM DESCRIPTION

The truck-drone joint inspection system includes a truck and a drone. The truck with a drone departs from a depot at a constant speed V_{truck} and finally returns to the depot when all inspection tasks are over. When the truck arrives at a stop location, the operator remotely controls a drone to conduct an inspection task. The drone can fly at a constant speed V_{drone} to inspect wind turbines around the stop location and finally return to where it is launched within the endurance D_{max}. The inspection time for each wind turbine lasts I minutes. Here the stop location is where a wind turbine locates. The time for pre-and post-flight procedures is P minutes. The goal is to minimize the return time of the truck after the drone inspects all wind turbines. Figure 1 shows the joint inspection system. According to the problem description, the truck-drone joint inspection system has the following features:

- If a drone is launched from point i, it must return to the same point i after its inspection.

- The drone cannot return to the truck multiple times when the truck stops.
- The truck shares the same road network as the drone.
- The drone can inspect only the wind turbine *i* and return to the truck without inspecting other wind turbines when the truck stops at *i*.

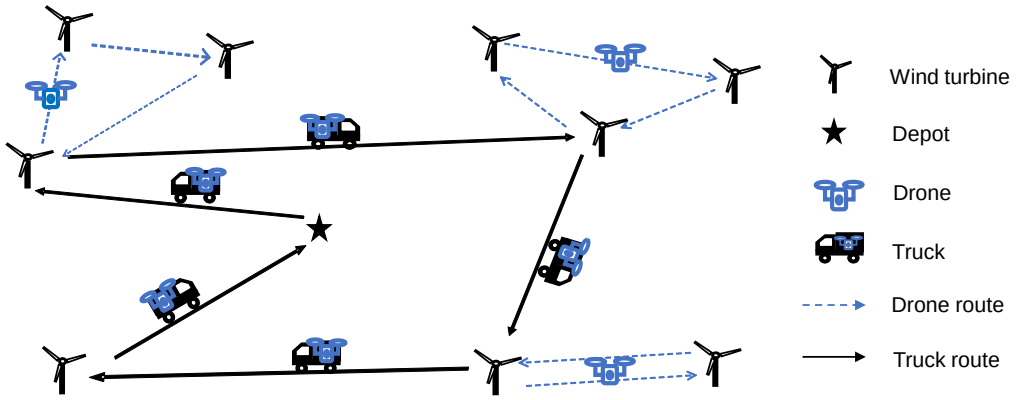

Figure 1: The Truck-drone Joint Inspection System.

Model notations are summarized in Table 1:

Table 1: Model notations.

Sets and Indices	
N	Set of wind turbines, $N=\{1,2,...n\}$
N^+	Set of wind turbines and depot, $N^+=N\bigcup\{0\}$
K	Set of clusters, $K=\{1,2,...k\}$
Parameters	
d_{ij}	The distance between node i to j
C_{ij}^t	The time for the truck to travel from i to j, $C_{ij}^t=d_{ij}/V_{truck}$
C_{ij}^d	The time for the drone to travel from i to j, $C_{ij}^d=d_{ij}/V_{drone}$
P	Time for conducting pre-and post-flight procedures
I	Time for inspecting one wind turbine
D_{max}	Drone endurance
M	A large positive number
Decision Variables	
x_{ij}^k	Binary. x_{ij}^k equals 1 if the drone in cluster k travels from i to j
y_{ij}	Binary. y_{ij} equals 1 if the truck travels from i to j
z_i^k	Binary. z_i^k quals 1 if node i in cluster k is selected as a truck stop

In order to describe the problem, we can formulate the MIP model as follow:

$$\min\ T_{total}=\sum_{i\in N}\sum_{j\in N,i\neq j}C_{ij}^d\cdot\sum_{k\in K}x_{ij}^k+\sum_{i\in N}\sum_{j\in N,i\neq j}C_{ij}^t\cdot y_{ij}+P\cdot\sum_{k\in K}\sum_{i\in N}z_i^k+I\cdot n \tag{1}$$

Subject to:

$$\sum_{i\in N}\sum_{k\in K}x_{ij}^k=1,\forall j\in N \tag{2}$$

$$\sum_{i\in N,i\neq p}x_{ip}^k-\sum_{j\in N,j\neq p}x_{pj}^k=0,\forall p\in N,\forall k\in K \tag{3}$$

$$\sum_{i \in N} x_{ij}^k \geq z_j^k, \forall j \in N, \forall k \in K \tag{4}$$

$$\sum_{j \in N} z_j^k \leq 1, \forall k \in K \tag{5}$$

$$\sum_{i \in N} \sum_{j \in N, i \neq j} (C_{ij}^d \cdot x_{ij}^k + I \cdot x_{ij}^k) \leq D_{max}, \forall k \in K \tag{6}$$

$$u_i - u_j + M \cdot \sum_{k \in K} x_{ij}^k \leq M - 1 + M \cdot \sum_{k \in K} z_j^k, \forall i, j \in N \tag{7}$$

$$\sum_{i \in N} y_{0i} = \sum_{i \in N} y_{i0} = 1 \tag{8}$$

$$\sum_{j \in N^+, i \neq j} y_{ij} = \sum_{k \in K} z_i^k, \forall i \in N^+ \tag{9}$$

$$\sum_{i \in N^+, i \neq p} y_{ip} - \sum_{j \in N^+, j \neq p} y_{pj} = 0, \forall p \in N \tag{10}$$

$$v_i - v_j + (n+1) \cdot y_{ij} \leq n, \forall i, j \in N, i \neq j \tag{11}$$

$$0 \leq u_i \leq n+1, u_i \in Z, \forall i \in N \tag{12}$$

$$0 \leq v_i \leq n+1, v_i \in Z, \forall i \in N \tag{13}$$

$$x_{ij}^k \in \{0,1\}, \forall i, j \in N, \forall k \in K \tag{14}$$

$$y_{ij} \in \{0,1\}, \forall i, j \in N \tag{15}$$

$$z_i^k \in \{0,1\}, \forall i \in N, \forall k \in K \tag{16}$$

Objective (1) is to minimize the total operation time departing from and returning to the depot. Here the total operation time (T_{total}) includes 3 parts: the total flight time for a drone (T_{drone}): $T_{drone} = \sum_{i \in N} \sum_{j \in N, i \neq j} C_{ij}^d \cdot \sum_{k \in K} x_{ij}^k + I \cdot n$; the total time for conducting pre-and post-flight procedures (T_{pre}): $T_{pre} = P \cdot \sum_{k \in K} \sum_{i \in N} z_i^k$; and the moving time for the truck: $T_{truck} = \sum_{i \in N} \sum_{j \in N, i \neq j} C_{ij}^t \cdot y_{ij}$.

Constraints (2) and (3) ensure that the drone can reach and leave the nodes exactly once. Constraint (4) indicates that if the node is selected as a parking point, the drone must arrive at the node. Constraint (5) ensures that at most one node can be selected as a parking point in one cluster. Constraint (6) enforces that the operating time of the drone in each cluster shall not exceed the endurance. Constraint (7) excludes subtours among wind turbines in each cluster. Constraint (8) ensures that the truck departs from and returns to the depot exactly once. Constraint (9) forces the relationship between y_{ij} and z_i^k. Constraint (10) is the flow conservation constraint that ensures once the truck visits a stop, it must also depart from the same stop. Constraint (11) excludes subtours among clusters and a depot. Constraints (12) and (13) indicate that v_i and u_i are integer variables. Constraints (14)-(16) indicate that x_{ij}^k, y_{ij}, and z_i^k are binary variables.

To minimize T_{total}, we need to cluster wind turbines, route the drone and the truck. In each cluster, the routing of the drone is a traveling salesman problem (TSP). Moreover, the routing of the truck in different clusters is a generalized traveling salesman problem (GTSP). The two basic problems are both NP-hard problems. It is well known that the problem is very difficult to solve in a short time. Thus, we propose an ALNS algorithm for this problem.

3 ALNS ALGORITHM

The ALNS framework presented by Ropke and Pisinger (2006) is an extension of LNS, which presents many destroy and repair methods that are statistically chosen according to the performance achieved during the search. Destroy methods can "destroy" the structure of the current solution, while repair methods rebuild the uncompleted solution. ALNS has been applied in many fields such as scheduling, vehicle routing problem, and its variant. The pseudo-code is shown as follows:

```
Input: S: Initial solution; RO: Removal set; IO: Insertion set;
```

```
      T_init: Initial temperature; T_end: Termination temperature;
S_best ←S, T ←T_init.
While stop-criterion not met do
   S′ ←S;
   Choose a destroy method io() and a repair method ro() from IO and RO;
      S′ ←io(ro(S′));
   if f(S′)<f(S_best) then
      S_best ←S ←S′;
   else if f(S′)<f(S) then
      S ←S′;
   else if random(0,1)<exp((f(S)-f(S′))/T) then
      S ←S′;
   end if
      T ←T*c;
   If T<T_end then
      T ←T_init;
      Update weight of ro() and io() by scores;
end while.
```

3.1 Initialization and Solution Representation

In this paper, a feasible solution can be represented as a two-dimensional array π_{ij}, where row π_i is a sequence of nodes, each node in the order it is visited by a drone. Then the first position π_{i0} of π_i is the launch point of the drone (truck stop location), which comprises the order visited by a truck. Note that the number of turbines is not necessarily equal in each row.

Considering the case of nine turbines and a depot. As is shown in Figure 2, the depot number is 0 and the turbine numbers are 1-9. The feasible solution consists of three one-dimensional arrays. Wind turbines 1, 2, and 3 are assigned to one cluster, wind turbines 6, 7, 5, and 4 are assigned to one cluster and wind turbines 8 and 9 are assigned to one cluster. The truck visits 1, 6, and 8 in sequence.

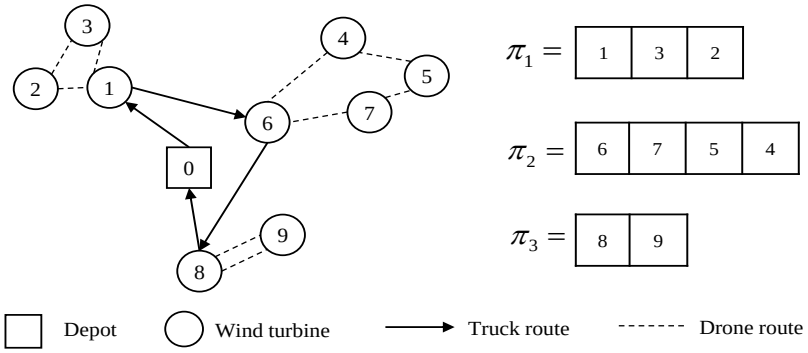

Figure 2: Initial solution and its representation.

An initial solution needs to be generated in such a way that it is feasible and more efficient in the algorithm. For this reason, we generate the initial solution as follows: all wind turbines are randomly permutated, which is used to be partitioned in multiple clusters by criteria of drone endurance. In the case of Figure 2, an initial permutation randomly generated is 1-3-2-6-7-5-4-8-9 and then three clusters are generated. The first two clusters have reached the limit of the number of wind turbines that can be accessed. Noticed that the number of clusters obtained here is not fixed and can be adjusted in ALNS.

3.2 Penalty Functions

Infeasible solutions are also possible in ALNS because of the drone endurance. Therefore, penalty terms for endurance violation need to be added to the cost function *TW*(*S*). Since a solution *S* is defined by a set of clusters ($S = \{\pi_1, \pi_2, \pi_3 ... \pi_k\}$), the overall penalty *p*(*S*) is calculated by summarizing cluster penalties for all clusters, which is given by Equation (17).

$$p(S) = \sum_{i=1}^{k} p(\pi_i) = \sum_{i=1}^{k} \lambda \cdot TW(\pi_i) \tag{17}$$

For each cluster π_i, $T_{\pi_i}^d$ represents flight time of the drone in cluster π_i. The endurance penalty for a cluster π_i is calculated by $TW(\pi_i) = \max\{T_{\pi_i}^d - D_{max}, 0\}$. λ is a weighting factor that is used to adjust the penalty term and set to 20 in this paper. ALNS uses a cost function $f(S) + p(S)$ to evaluate the solution *S*.

3.3 Destroy and Repair Method

In ALNS, the design of destroy and repair methods is the key to the effectiveness of the algorithm. In this paper, five destroy methods and three repair methods are designed which adapt to the problem. Note that the number of wind turbines need to be removed (*q*) is 3. The destroy methods are designed as follows:

1. Random removal: The random removal method simply selects *q* wind turbines at random and removes them from the solution.
2. Worst removal: We define the cost of the wind turbine *i* as cost(*i*)=$d_{i\text{-}1,j} + d_{i,j+1} - d_{i,j}$ where $d_{i,j}$ is the distance between wind turbine *i* and wind turbine *j*. This method removes the *q* wind turbines with the highest *cost*(*i*). The worst removal is commonly used and effective for many other problems.
3. Shaw removal: The method was proposed by Shaw (1997). It employs a relatedness measure *R*(*i*,*j*) to describe the similarity between two vertices. In this study, *R*(*i*,*j*) is measured in distance between two wind turbines, such that removing the wind turbines that are close to each other enables repositioning them to a better one. We first select one wind turbine *i* and calculate other wind turbines' relatedness *R*(*i*,*j*). Then *q* wind turbines with the highest *R*(*i*,*j*) are removed.
4. Maximum distance removal: For each cluster, calculate the distances between wind turbines and the truck stop and sort them in descending order. We select the *q* wind turbines to remove. The method can minimize the range of a cluster as far as possible so that the cluster can include more wind turbines.
5. Endurance limit removal: For each cluster, if the drone flying time exceeds the drone endurance, we traverse forward the cluster which is violated the constraint, and remove the wind turbine until it is feasible. Experiments show that this method is very effective in finding better solutions.

The repair methods are designed as follows:

1. Random insertion: The random insertion method is similar to the random destroy method. It simply selects the positions of permutation at random and inserts the removed wind turbines into them.
2. Closest insertion: This simple construction heuristic corresponds to the Worst removal method. Let $\Delta(i,t) = d_{i-1,t} + d_{t,i+1} - d_{i,i+1}$ denotes the change of total distance in the solution if we insert the wind turbine *t* into position *i*. For each wind turbine *t* that needs to be inserted, calculate the $\Delta(i,t)$ and insert it into position with the lowest $\Delta(i,t)$.
3. Greedy insertion: For each moved wind turbine, it traverses all positions and inserts it with the lowest reduction of total operation time $\Delta f(s)$.

3.4 Adaptive Weight Adjustment and Acceptance

In this paper, the entire search of ALNS is divided into several segments. A segment is several iterations of the ALNS. Here we reference the simulated annealing algorithm idea to finish a segment. The segment starts initially with temperature T_0 which is set to a high value, and then it is decreased at each step with the annealing rate c and ends with the termination temperature T_{end}.

The basic idea of adaptive weight is that ALNS can adjust the methods' weight in each iteration according to its performance in the last iteration. In the beginning, the initial weight ω_i^0 of every method i is set to be equal. In each iteration t, we use a roulette wheel selection principle to select a destroy method ($ro_i \in RO$)and a repair method ($io_i \in IO$). The weight of each method is calculated by $\omega_i^t / \sum_{i \in O} \omega_i^t$, where $i = ro_i, O = RO$ or $i = io_i, O = IO$. After applying the destroy method ro_i and repair method io_i, the scores for methods are updated by adding θ_i. If a global optimal solution is found, the scores for ro_i and io_i increase θ_1; When a solution that is better than the current solution is found, the scores for ro_i and io_i increase θ_2; If the solution is not better than the current one but is accepted by the simulated annealing accept criterion, the scores for ro_i and io_i increase θ_3 ($\theta_1 > \theta_2 > \theta_3$). The criterion accepts a current solution S' with probability $\exp(-(f(S') - f(S))/T_{iter})$ where T_{iter} is the temperature of the current iteration *iter*. At the end of each segment, we calculate the new weight according to the new scores. Let $w_{i,j}$ be the weight of the method i used in segment j. Then the weight for all methods to be used in segment j+1 is calculated by $w_{i,j+1} = b \times w_{i,j} + (1-b)\theta_i / \beta_i$. β_i is the number of times that method i is selected in the segment and b is the reaction factor that adjusts the $w_{i,j}$. When the maximum number of iterations is reached, ALNS terminates and outputs the optimal solution.

In this paper, we choose the ALNS parameters as follows: the maximum number of iterations (*Iter*) is 20, initial temperature (T_0) is 5000, termination temperature (T_{end}) is 1e-8, annealing rate (c) is 0.5, and reaction factor (b) is 0.5.

4 COMPUTATIONAL EXPERIMENT

In this section, we generate new instances to simulate the real wind farms and test ALNS. Then, 3 real wind farms in the United States are also introduced to evaluate the performance of ALNS. All experiments which include instance generation and ALNS are coded in C++ on a personal computer with an AMD Ryzen 7 4800H (2.90 GHz) processor with 16 GB of RAM.

4.1 Simulation of Wind Farm Inspection Cases

To simulate the real inspection scenario, we set the parameters of the truck and drone according to the actual situation. Here the flight speed of the drone (V_{drone}) is 64 km/h, the truck moving speed (V_{truck}) is 32 km/h, the endurance of the drone (D_{max}) is 50 min, the time for pre-and post-flight procedures (P) is 5 min, and the inspection time for each wind turbine (I) is 5 min.

To fit the reality, the distance between two turbines needs to be considered when generating the instances. The minimum distance between two wind turbines depends on how large the rotor diameter is. Generally, the wind turbines themselves need to be around '5 rotor diameters' apart so that they don't affect each other. The rotor diameter of a large turbine is around 50-100 meters (Tasneem et al. 2020). 80m is chosen as the turbine blade diameter and the minimum distance between two wind turbines is 0.4 km. If the minimum distance between the new point and other existing points exceeds the limit, the new point is removed otherwise it keeps. To evaluate the performance of the algorithm in different geographical data distributions, we design the random instances, clustered instances, and instances with random and clustered structures.

For all instances generated, the central depot is always located at coordinates (0,0), while the wind turbines' geographical data are generated in a grid of dimensions 2dx2d around the depot. Every instance is named as (n,m,l), where n is the number of wind turbines in the scenario, m is the dimension of the grid (m=2d) and l is the label of the scenario. Since wind farms cover an area of about 10 km^2 to 100 km^2, the number of turbines varies from tens to hundreds. For different size instances, n and m are set as in Table 2.

As an example, (50.10.c) represents 50 wind turbines are generated in the grid of dimensions 10x10 with clustered structures.

Table 2: Parameter of instances generation.

Size	*n*	*m*
small	25	5
medium	50	10
large	100	16

4.1.1 Random Instances Generation

For each instance, we generated *n* wind turbines with coordinates following a uniform distribution *U*(*-d, d*). The label of random instances is "r". Figure 3 shows the location of three random instances.

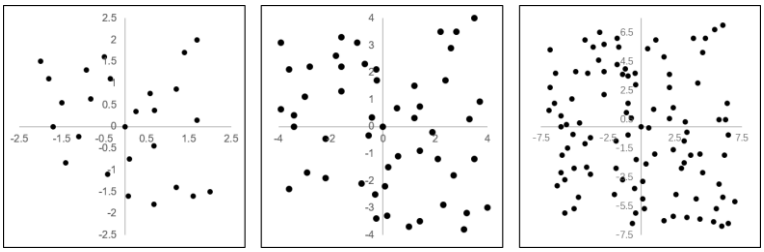

Figure 3: The picture on the left is the geographical distribution in (25.5.r), the middle is the geographical distribution in (50.10.r) and the right is (100.16.r).

4.1.2 Clustered Instances Generation

For each instance, we will generate θ focal points around the grid and the distance between two focal points over $d/4$. θ is the number of focal points which can be calculated as $\theta = \lceil n / x \rceil + rand(0, 2)$, where x is the maximum number of wind turbines the drone can visit in a cluster and $rand(0, 2)$ represents the integer generated randomly from 0 to 2. The maximum number of wind turbines in a cluster can also be estimated by $x \geq D_{max} / I$. Then each wind turbine is randomly assigned to a focal point where the wind turbine's location follows a normal distribution centered on the focal point and with a standard deviation of 0.4 km. The label of random instances is "c". Figure 4 shows the location of three clustered instances.

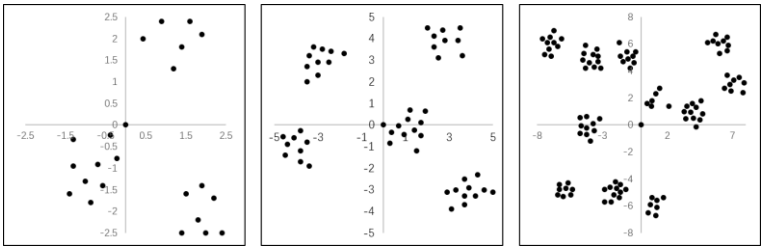

Figure 4: The picture on the left is the geographical distribution in (25.5.c), the middle is the geographical distribution in (50.10.c) and the right is (100.16.c).

4.1.3 Mixed Instances Generation

Mixed instances are generated by the way combined with the above two generating methods. For each instance, 50% of wind turbines follow the uniform distribution *U*(*-d, d*) while 50% are in the clustered

structure. The label of mixed instances is “rc”. Figure 5 shows the location of three instances in a mixed structure.

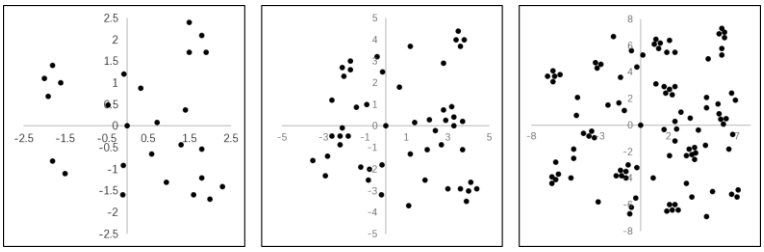

Figure 5: The picture on the left is the geographical distribution in (25.5.rc), the middle is the geographical distribution in (50.10.rc) and the right is (100.16.rc).

4.2 Experimental Results with Generated Instances

In this section, the method of "clustering first-routing second" proposed by Baik and Valenzuela (2020) is compared with ALNS. The method is named C&R. C&R was run on the same computer using C++ programming, where the TSP and G-tsp models were solved using a CPLEX 12.9.0 solver. Set the solution time of the two models to 1800 seconds. To verify the performance of ALNS and C&R, the study also uses CPLEX 12.9.0 to solve the mixed integer programming model of the problem with an 1800 seconds limit.

Table 3 shows the specific results, where C&R and ALNS were run 10 times. In the columns C&R and ALNS, *AvgObj* is the average result for 10 runs, *MinObj* is the best result of the 10 runs, and *Time* is the average execution time for the 10 runs.

Table 3: Performance of different methods on generated instances.

case	CPLEX		C&R			ALNS		
	Obj (min)	*Time* (s)	*AvgObj* (min)	*MinObj* (min)	*Time* (s)	*AvgObj* (min)	*MinObj* (min)	*Time* (s)
(25.5.r)	172.11	1800	173.45	171.23	1.63	173.59	171.22	0.24
(50.10.r)	408.34	1800	380.80	369.60	6.55	370.45	364.25	0.88
(100.16.r)	-	1800	815.62	800.28	1152.98	809.46	786.84	5.36
(25.5.c)	168.59	1800	180.64	174.14	1.31	171.78	168.88	0.25
(50.10.c)	388.66	1800	389.69	374.52	497.37	371.78	366.59	1.02
(100.16.c)	-	1800	773.35	758.51	1800.00	750.06	729.63	7.62
(25.5.rc)	171.66	1800	184.09	174.62	2.54	175.34	172.84	0.24
(50.10.rc)	394.01	1800	379.95	371.34	3.80	362.76	360.01	0.86
(100.16.rc)	-	1800	798.24	773.29	1120.09	780.83	764.11	8.81

As is shown in Table 3, ALNS significantly outperforms CPLEX and C&R in terms of execution time. The time consumed by C&R increases significantly with the size of the problem. In particular, the method takes a long time to solve up to 100 turbines. The time taken by C&R varies considerably depending on the geographical distribution of the turbines, even for the same size case. The ALNS algorithm takes less than 10s for all instances on average.

For all instances, CPLEX cannot find the optimal solution in 1800 s and up bounds can be obtained for small and medium instances. ALNS provides the lowest objective except for (25.5.c) and (25.5.rc). The average objective using ALNS is lower than C&R in all instances except for (25.5.r). It can be concluded that ALNS seems well suited to solve instances with different sizes and geographic distribution in a reasonable time at a good quality.

4.3 Experiments with Real Wind Farms

Since the generated instances may not reflect the effectiveness of the algorithm in the actual situation, we select 3 real cases with different sizes. The cases are real wind farms in the United States and are provided by Baik and Valenzuela (2020). The small-size case is Golden Spread Panhandle Wind Project which comprises 34 wind turbines. The medium-size case is Hackberry Wind Project which comprises 72 wind turbines. 100 wind turbines are comprised in the Langford Wind Project which represents a large-size case. Detailed data can be found in Baik and Valenzuela (2020). Note that the depots are located at the mass center of the wind turbines.

The experimental results are shown in Table 4. We study the effect of total operation time under the cases. Three parts of total operation time were compared in ALNS and C&R. The columns labeled *gap* is formulated as

$$gap = \frac{T_{total}^{C\&R} - T_{total}^{ALNS}}{T_{total}^{C\&R}}.$$

It indicates the percentage deviation in the total operating time obtained by the ALNS and C&R algorithm.

From the results, we observe that ALNS outperforms C&R for the three wind farm sizes. ALNS improves total inspecting time as the wind farm sizes increases. In the small-size case Golden, although the solution in ALNS and C&R have the same number of clusters, the total operation time obtained by ALNS is shortened by 0.43% compared with C&R because of the reduction of truck moving time. In the medium-size case Hackberry, the reduction of cluster number leads to a significant reduction in pre-and post-flight procedure time and truck moving time, which decreases by 2.31% compared with C&R in total time. In the large-size case Langford, the number of ALNS clusters is reduced by 9 compared with C&R, the pre-and post-flight procedure time is shortened by 45 minutes, and the truck moving time is reduced by about 12 minutes. Although the flight time increases, the total time of ALNS is reduced by 3.84%.

Taking a medium-scale case as an example, Figure 6 shows the truck-drone inspecting route of ALNS algorithm while Figure 7 shows the truck-drone inspecting route of C&R. It can be seen that the number of clusters in ALNS is smaller than C&R. The limitations of K-means algorithm are also shown in Figure 7: poor recognition of non-convex shapes. The geographic distribution of the wind turbines in each cluster is a kind of circular or circular-like shape in Figure 7, which shows that the C&R could easily fall into local optimal. The ALNS algorithm has a better performance by searching the solution globally.

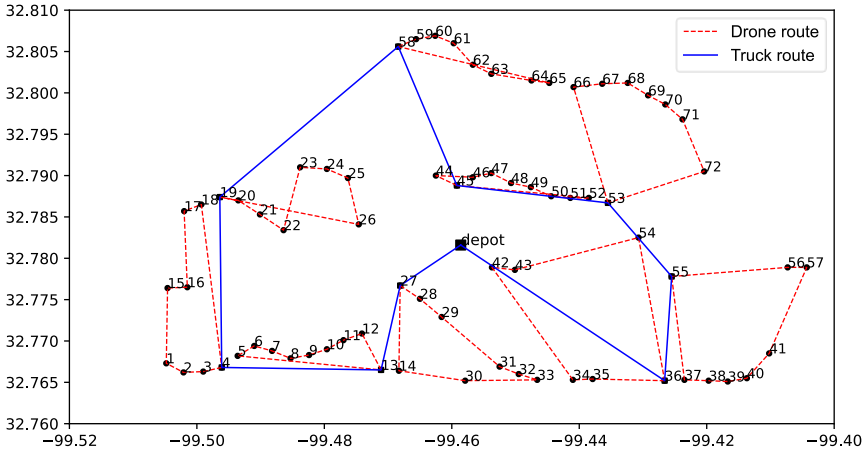

Figure 6: Medium-sized wind farm routing solution of the drone and truck by ALNS.

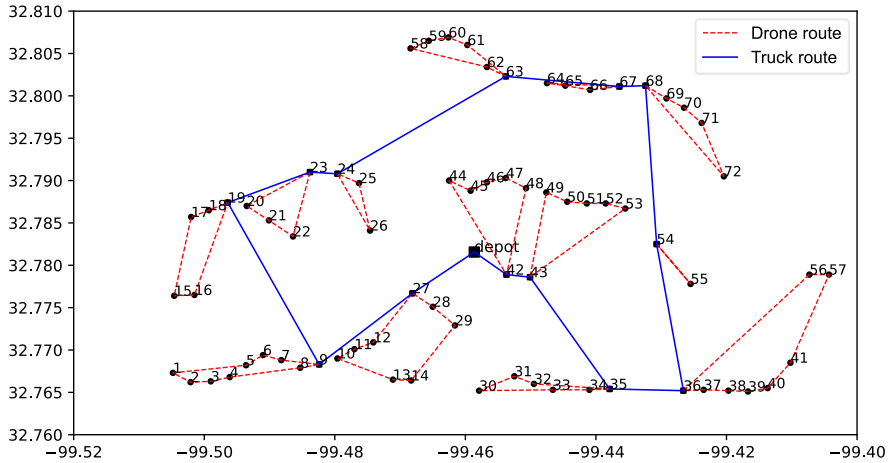

Figure 7: Medium-sized wind farm routing solution of the drone and truck by C&R.

Table 4: Comparative results of total operation time.

case	C&R				ALNS				*Gap* (%)
	T_{drone} (min)	T_{pre} (min)	T_{truck} (min)	T_{total} (min)	T_{drone} (min)	T_{pre} (min)	T_{truck} (min)	T_{total} (min)	
Small	190.94	25	21.93	237.87	193.16	25	18.67	236.83	0.43
Medium	400.33	65	36.17	501.50	406.29	45	38.63	489.91	2.31
Large	553.59	110	75.65	739.24	582.43	65	63.39	710.83	3.84

5 CONCLUSION

In this paper, a joint inspection system with a truck and drone is introduced. An adaptive large neighborhood search (ALNS) algorithm was provided to solve the problem. We created instances of various sizes and distributions by simulating real wind farms to test ALNS. The results showed that ALNS proposed by this paper outperforms the method proposed by predecessors in both solution quality and execution time. In addition, results in real wind farms showed that the total operation time in ALNS is shorter than that in another method, which comes from the reduction of truck moving time and pre-and post-flight procedure time.

In future research, using multiple drones in this inspection system can be explored and the stop location of the truck can be considered instead of the location of wind turbines. Also, due to the wind conditions, the flight time between two wind turbines may not be known with certainty. The uncertain amount of flight time can be considered in the future.

ACKNOWLEDGMENTS

This research is partially supported by the National Science Foundation of China [Grant 72071108; 71701095] and by the Humanities and Social Science Foundation of the Ministry of Education [Grant 17YJC630050].

REFERENCES

Agatz, N., P. Bouman, and M. Schmidt. 2018. “Optimization Approaches for the Traveling Salesman Problem with Drone”. *Transportation Science* 52(4):965–981.

Baik, H., and J. Valenzuela. 2020. "An Optimization Drone Routing Model for Inspecting Wind Farms". *Soft Computing* 25(3):2483–2498.

Bouman, P., N. Agatz, and M. Schmidt. 2018. "Dynamic Programming Approaches for the Traveling Salesman Problem with Drone". *Networks* 72(4):528–542.

Cao, P., Y. Liu, M. Qiu, C. Yang, and S. Xie. 2019. "MEC-Driven UAV Routine Inspection System in Wind Farm under Wind Influence". In *Proceedings of 12th International Conference on Intelligent Computation Technology and Automation*, Oct 26th-27th, Xiangtan, China, 672–677.

Domínguez, S., R. Suarez Fernandez, and P. Campoy. 2017. "L1 Adaptive Control for Wind Gust Rejection in Quad-Rotor UAV Wind Turbine Inspection". In *Proceedings of 2017 International Conference on Unmanned Aircraft Systems*, Jun 13th-16th, Miami, USA, 1840–1849

Drexl, M. 2014. "Branch-And-Cut Algorithms for the Vehicle Routing Problem with Trailers and Transshipments". *Networks* 63(1):119–133.

Ha, Q. M., Y. Deville, Q. D. Pham and M. H. Hà. 2018. "On the Min-Cost Traveling Salesman Problem with Drone". *Transportation Research Part C: Emerging Technologies* 86:597–621.

Ha, Q. M., Y. Deville, Q. D. Pham and M. H. Hà. 2019. "A Hybrid Genetic Algorithm for the Traveling Salesman Problem with Drone". *Journal of Heuristics* 26(2):219–247.

Jeong, H. Y., B. D. Song, S. Lee. 2019. "Truck-Drone Hybrid Delivery Routing: Payload-Energy Dependency and No-Fly Zones". *International Journal of Production Economics* 214:220–233.

Juengert, A. 2008. "Damage Detection in Wind Turbine Blades Using Two Different Acoustic Techniques", NDT DataBase. https://www.ndt.net/article/v13n12/juengert.pdf, accessed 12th March 2022.

Luo, Z., Z. Liu, and J. Shi. 2017. "A Two-Echelon Cooperated Routing Problem for a Ground Vehicle and Its Carried Unmanned Aerial Vehicle". *Sensors (Basel)* 17(5).

Marinelli, M., L. Caggiani, M. Ottomanelli, and M. Dell'Orco. 2018. "En Route Truck-Drone Parcel Delivery for Optimal Vehicle Routing Strategies". *IET Intelligent Transport Systems* 12(4):253–261.

Murray, C. C., and A. G. Chu. 2015. "The Flying Sidekick Traveling Salesman Problem: Optimization of Drone-Assisted Parcel Delivery". *Transportation Research Part C: Emerging Technologies* 54:86–109.

Poikonen, S., B. Golden, and E. A. Wasil. 2019. "A Branch-And-Bound Approach to the Traveling Salesman Problem with a Drone". *INFORMS Journal on Computing* 31(2):335–346.

Ponza, A. 2016. "Optimization of Drone-Assisted Parcel Delivery". Master's thesis, Università Degli Studi Di Padova, 2016.

Roberti, R., and M. Ruthmair. 2021. "Exact Methods for the Traveling Salesman Problem with Drone". *Transportation Science* 55(2):315–335.

Ropke, S., and D. Pisinger. 2006. "An Adaptive Large Neighborhood Search Heuristic for the Pickup and Delivery Problem with Time Windows". *Transportation Science* 40(4):455–472.

Savuran, H., and M. Karakaya. 2015. "Route Optimization Method for Unmanned Air Vehicle Launched from a Carrier". *Lecture Notes on Software Engineering* 3(4):279–284.

Shaw, P. 1997. "A New Local Search Algorithm Providing High Quality Solutions to Vehicle Routing Problems". Technical report, Department of Computer Science, University of Strathclyde, Scotland.

Tasneem, Z., A. Al Noman, S. K. Das, D. K. Saha, M. R. Islam, M. F. Ali, M. F. R Badal, M. H. Ahamed, S. I. Moyeen, and F. Alam. 2020. "An Analytical Review on the Evaluation of Wind Resource and Wind Turbine for Urban Application: Prospect and Challenges". *Developments in the Built Environment* 4.

Villegas, J. G., C. Prins, C. Prodhon, A. L. Medaglia, and N. Velasco. 2013. "A Matheuristic for the Truck and Trailer Routing Problem". *European Journal of Operational Research* 230(2):231–244.

Wang, L., and Z. Zhang. 2017. "Automatic Detection of Wind Turbine Blade Surface Cracks Based on UAV-Taken Images". *IEEE Transactions on Industrial Electronics* 64(9):7293–7303.

AUTHOR BIOGRAPHIES

WENYU TAO is a post-graduate student in College of Economic and Management at Nanjing University of Aeronautics and Astronautics. He received his B.Eng. Degree from Dalian Marinetime University. His research mainly focuses on logistics and supply chain management and meta-heuristic algorithm. His email address is cemtwy@nuaa.edu.cn.

XINJIA JIANG is an Associate Professor in College of Economic and Management at Nanjing University of Aeronautics and Astronautics. He received his B.Eng. Degree from Nanjing University and Ph.D. from National University of Singapore. His research interests include port management,transportation and logistics optimization. His email address is cemjxj@nuaa.edu.cn.

DONGQIANG ZHAO is a post-graduate student in College of Economic and Management at Nanjing University of Aeronautics and Astronautics. He received his B.Eng. Degree from Jiangnan University. His research mainly focuses on the optimization in the warehouse and meta-heuristic algorithms. His email address is cemzdq@nuaa.edu.cn.