OmniaScience

Journal of Industrial Engineering and Management
JIEM, 2023 – 16(2): 215-235 – Online ISSN: 2013-0953 – Print ISSN: 2013-8423
https://doi.org/10.3926/jiem.4381

Route Planning of Heterogeneous Unmanned Aerial Vehicles under Recharging and Mission Time with Carrying Payload Constraints

Kriangsak Phalapanyakoon, Peerapon Siripongwutikorn

Department of Computer Engineering, King Mongkut's University of Technology Thonburi (Thailand)

kriangsak.phala@mail.kmutt.ac.th, peerapon.sir@mail.kmutt.ac.th

Received: August 2022
Accepted: March 2023

Abstract:

Purpose: We consider the problem of route planning of multiple rechargeable heterogeneous UAVs with multiple trips under mission time and payload carrying constraints. The goal is to determine the types and number of UAVs to be deployed and their flying paths that minimizes the monetary cost, which is a sum of the recharging energy cost of each UAV, the UAV rental cost, and the cost of violating the mission time deadline.

Design/methodology/approach: The problem is formulated as a mixed integer programming (MIP). Then, the genetic algorithm (GA) is developed to solve the model and the solutions are compared to those obtained from the exact method (Branch-and-Bound). Novel chromosome encoding and population initializations are designed, and standard procedures for crossover and mutation are adapted to this work. Test problems on grid networks and real terrains are used to evaluate the runtime efficiency and solution optimality, and the sensitivity of GA parameters is studied based on two-level factorial experiments.

Findings: The proposed GA method can find optimal solutions for small problem sizes but with much less computation time than the exact method. For larger problem sizes, the exact method failed to find optimal solutions within the limits of time and disk space constraints (24 hours and 500 GB) while the GA method yields the solutions within a few minutes with as high as 49% better objective values. Also, the proposed GA method is shown to well explore the solution space based on the variation of the total costs obtained.

Originality/value: The unique aspects of this work are that the model optimizes the sum of three different costs – the electricity recharging cost, the UAV rental cost, the penalty cost for mission deadline violation, and the recharging period based on the remaining energy, the payload capacity, and the heterogeneity of UAVs are incorporated into the model. The model is formulated as a mixed integer programming and the genetic algorithm is developed to solve the program. Novel chromosome encoding and population initializations are designed, and standard procedures for crossover and mutation are adapted to this work.

Keywords: unmanned aerial vehicle (UAV), heterogenous rechargeable unmanned aerial vehicles, route planning, genetic algorithm (GA)

To cite this article:

Phalapanyakoon , K., & Siripongwutikorn, P. (2023). Route planning of heterogeneous unmanned aerial vehicles under recharging and mission time with carrying payload constraints. *Journal of Industrial Engineering and Management*, 16(2), 215-235. https://doi.org/10.3926/jiem.4381

1. Introduction

The cutting-edge unmanned aerial vehicle (UAV) technology has revolutionized various industries, including agriculture, construction, weather forecasting, or logistics. The UAV has been improved to be smaller, faster, able to carry more payload weight, and fly longer. This enables the benefits of only using UAVs in parcel delivery missions, especially in rural areas or areas inaccessible to humans. For example, a UAV-based vaccine delivery system was deployed in North-Eastern India (Press Information Bureau of Delhi, 2021). UAVs can deliver vaccines to specified drop-off locations in the coverage area over 31 square km in less than 15 minutes, which usually takes three to four hours by ground transportation. Such mission requires UAV route planning, where a sequence of locations to visit and the amount payloads to deliver in each location are determined for each UAV. Depending on the characteristics of UAVs deployed in the mission, UAVs may be required to fly multiple trips by occasionally returning to the base station to recharge batteries or replenish payloads, especially in a delivery mission with a large area or with a high demand for items being delivered. Inappropriate planning of UAV routes most likely results in excessive electricity recharging costs and mission time. Therefore, optimal UAV route planning is essential to the efficient deployment of UAVs.

Different types of UAVs have different battery capacities, payload capacities, speeds, energy consumption, and recharging rates, which incur different electricity costs that depend on the remaining battery energy before recharging when returning to the base station. Adding more UAVs or using high-performance UAVs will shorten the mission time but increase the UAV rental cost. The mission time, which is considered a soft constraint, will be penalized if it exceeds the mission deadline. Battery and payload capacities were considered hard constraints, which can result in UAV flying multiple trips in large coverage or high-demand areas. Our objective is to determine flying paths of UAVs that minimize the total operational cost. This type of problem is known to be an NP-hard problem (Laporte, 1992). The exact solution method is inadequate because it cannot solve for optimal solutions in large problem sizes. Thus, the metaheuristic approach, which is a genetic algorithm, is developed and implemented for our proposed mathematical model in this research.

This work considers UAV route planning in the context of heterogenous UAVs with the impacts of UAV payload capacity and recharging time being accounted for. The major contributions of this research are two-fold:

- The mathematical programming model for UAV route planning with heterogenous UAVs is formulated, wherein different rental costs, flying speeds, battery consumption rates, battery capacities, recharging rates and payload capacities of the UAVs are incorporated to minimize the total operating cost. The proposed model determines the types and number of UAVs to be deployed and their flying paths that minimize the sum of electricity recharging cost, UAV rental cost, and cost of excess mission time.
- We develop the Genetic Algorithm (GA) to solve the model. Extensive experimentation on several test problem scenarios with a wide variety of problem sizes and model parameters as well as real-terrain scenarios is conducted to assess the effectiveness and computational efficiency of the solution approach. Effects of GA parameters are also investigated for sensitivity and robustness of the proposed algorithm.

The remainder of this paper is organized as follows. Section 2 reviews related works on the Genetic Algorithm and their modifications. In Section 3, the problem description, assumptions, notations, mathematical models, and proposed genetic algorithm are presented. Section 4 discusses the computational results from our model, and the sensitivity of GA parameters also studied by using designed experiments. In Section 5, we conclude the work and provide directions for future research.

2. Related Works

The problem of item or parcel delivery by using UAVs closely resembles the well-known vehicle routing problem (VRP), which is known to be NP-hard. In VRP, a fleet of vehicles must visit the predefined locations with the cost of visiting locations being minimized under the vehicle capability and other constraints (Dantzig & Ramser, 1959). Essentially, the UAV routing planning problem is an extension of VRP with more complicated constraints, decision variables, and objective functions. Due to the NP-hard characteristics of the problems, exact solution methods based on mathematical models are inadequate, and meta-heuristics are commonly adopted as an alternative solution approach. Because of the discrete combinatorial problem structure of VRP, genetic algorithm (GA) is deemed as a suitable solution approach and a great deal of works has successfully applied (GA) to VRP and UAV routing planning with different objectives and how the mutation and crossover operations are designed. For the objective of minimizing the total traveling distance, Kurnia, Wahyuni, Pembrani, Gardini and Aditya (2018) used the rank selection method to select parent chromosomes for position-based crossover and mutation to handle multi-depot and multi-compartment problems. Sbai, Krichen and Limam (2020) proposed embedded variable neighborhood search (VNS) in the mutation operator with tournament selection and partially mapped crossover (PMX). May, Jariyavajee and Polvichai (2021) compared several mutation techniques and assessed performance by well-known Solomon benchmarks. Anggodo, Ariyani, Ardi and Mahmudy (2017) handled multi-trip by injecting the stop location into the randomly generated chromosomes and using well-known elitism selection, one-cut point crossover, and reciprocal exchange mutation. The crossover and mutation rates were also varied to investigate their influence on the fitness score. Yuan, Zhu, Li, Huang and Wu (2021) considered heterogeneous UAV logistics with the objective of minimizing the maximum completion time in a multi-trip manner under the restriction of the maximum flight time of each UAV. With a similar crossover procedure to Yuan et al.'s work, Zhen, Ma, Wang, Xiao and Zhang (2020) proposed a modified fitness evaluation function that includes both costs and the diversity of the population to improve the quality of the solution to the last-mile distribution. The proposed hybrid algorithm can handle both multi-depot and multi-trip situations. Local search-variable neighborhood descent (LS-VND) was embedded in the mutation operators to improve the quality of solutions that lead to near-optimal results. In 2021, Euchi and Sadok (2021) developed a hybrid genetic-sweep algorithm that collaborates in both sidekick and independent flying modes.

Many researchers considered multiple cost metrics in the fitness functions such as the total traveling distance and the cost of vehicles. Mutingi and Mbohwa (2013) developed a group genetic algorithm (GGA) to handle the complexity of multi-trip and heterogeneous UAVs. Similar to Mutingi and Mbohwa's work, Liu, Huang and Ma (2009) developed GA with revised local search to enhance the exploitation rate of solutions. In Setiawan, Masruroh and Pramuditha (2019), the two-level factorial design was implemented for parameter tuning that includes population size, maximum generation, crossover, and mutation probability. Khoukhi, Yaakoubi, Bojji and Bensouda (2019) implemented GA to solve pick-up and delivery in a hospital, where two-cut points crossover and two-opts mutation are integrated to ensure the exploration and diversity of the population. The stopping cost or the cost of overtime of the mission time is considered in Ayadi and Benadada (2013) and Eroglu, Gencosman, Cavdur and Ozmutlu (2014). Ayadi and Benadada (2013) used a local search algorithm after each newborn child generated by the crossover and mutation process to minimize the maximum overtime of vehicles and routing costs, while Eroglu et al. (2014) proposed a hybrid genetic-local search algorithm applied to the fitness function calculation process to eliminate infeasible solutions.

The fuel or battery capacity constraints result in refueling or recharging time that must be added to the total traveling time. Nevertheless, most existing works only obtained solutions for small problem sizes based on mathematical models. Dorling, Heinrichs, Messier and Magierowski (2017) proposed a multi-trip VRP for UAV delivery that accounted for the impact of payload weight on energy consumption with the objectives of minimizing the total cost under the time limit and minimizing the total time under budget constraints. Battery swapping was used in their work instead of recharging batteries based on the remaining energy but the cost of using UAVs was wnot considered. Both exact solution method and simulated annealing were implemented to obtain the solutions. However, it was found that simulated annealing does not well capture the characteristics of VRP. Coelho, Coelho, Coelho, Ochi, Haghnazar, Zuidema et al. (2017), UAVs are allowed to visit the charging station to refuel or recharge based on their remaining energy in each trip. Only the exact method was implemented in their work, while they also suggested that implementing a metaheuristic algorithm would be a reasonable approach to achieving good solutions for larger problem sizes. In Troudi, Addouche, Dellagi and Mhamedi (2018), a fleet of homogeneous UAVs was deployed to visit a set of customers and fly back to the depot for instant battery swapping, reload packages, or continue to visit customers if it's not running out of battery. The objective is to minimize the total distance, total used UAVs, and the total number of batteries used in swapping. Even though the mathematical models mentioned above were inspiring and interesting, they all suggested that only exact or heuristic methods are insufficient for obtaining the optimal solution for larger problem sizes and complex problems.

3. Proposed Work

The problem of heterogeneous UAVs route planning under recharging, carrying payload, and mission time constraints is formally defined in this section. Then, the mixed-integer programming (MIP) model, along with the Genetic Algorithm (GA), was developed for the problem. The MIP model presented here is extended from the previous work (Phalapanyakoon & Siripongwutikorn, 2021) to handle the heterogeneity of UAVs and the payload constraints.

3.1. Problem Description and Assumptions

Given a base station and a set of locations with known coordinates and required amount of demands, the mission is to deliver items to fulfill all the location demands by using UAVs that are launched from the base station and return to the base station after completing the mission. The mission has the deadline to complete, which may incur some penalty if being violated. A set of UAVs with different battery capacities, speeds, energy consumption and recharging rates, maximum carrying capacities, and rental prices are available to be chosen for deployment. Due to limited UAV battery and carrying-payload capacities, each UAV may have to return to the base station for recharging and/or replenishing its payload, which is referred to as a trip. To complete the mission, individual UAVs may take multiple trips, and the sequence of visited locations taken by each UAV on all its trips is referred to as a route. Note that the route taken by a UAV can have different sequences of locations from one trip to another. Examples of trips and routes are depicted via timing diagrams in Figure 1. In the figure, UAV 1 and 2 take two trips, where only UAV 1 recharges its battery when returning to the base station (Location 1). The routes of UAV 1 and UAV 2 are respectively 1-2-11-1-12-3-1 and 1-6-7-1-5-4-1. UAV 3 only takes one trip. The mission time constraint is violated as UAV 1 returns to the base station after the deadline.

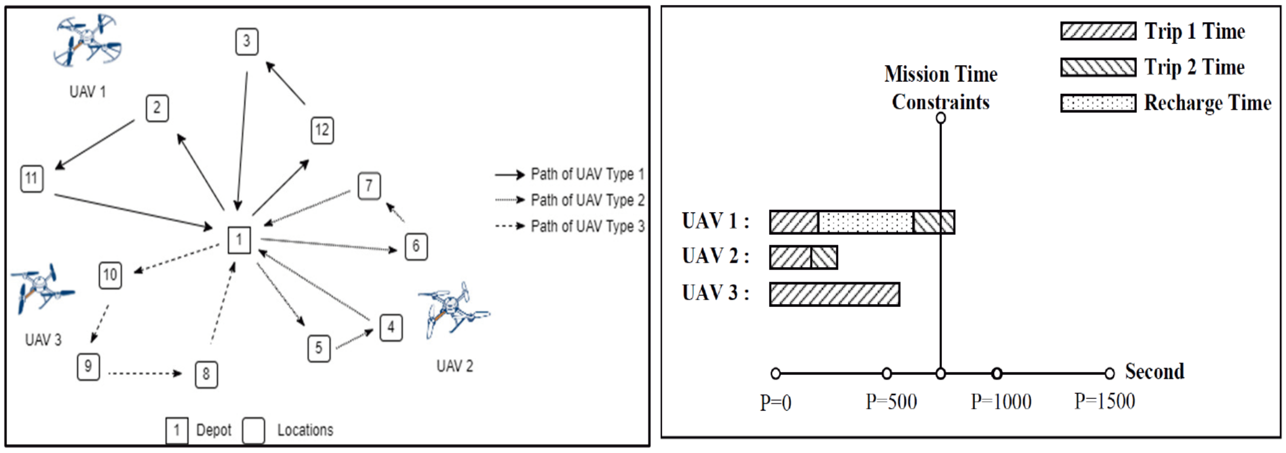


Figure 1. An example of routes and timing diagrams of heterogeneous UAVs

From the problem description, the total cost of the mission is a weighted sum of the electricity cost from recharging, the cost of UAV rentals, and the penalty cost of exceeding the mission time deadline. Our goal is to determine a number of required UAVs of each type to be used together with their route and the recharging and payload replenishing schedules so that the total cost is minimized. Note that the optimization metrics such as flying distance and time are respectively handled by the recharging cost and the deadline penalty cost. The following assumptions are made in formulating the model to solve the problem:

- UAVs have the full battery level at the beginning of the mission.
- UAVs start and end their routes at the base station.

- UAVs are heterogeneous, so they can have different energy consumption rates, recharging rates, flying speeds, battery capacities, and payload carrying capacities.
- The energy consumption of UAVs linearly increases with flying distance.
- Times taken by UAVs to replenish payloads are very small compared to the recharging time and flying time, and are not taken into account.
- Each of the UAV's capacities must be at least the maximum of the location demands.
- No recharging is performed on the last trip of each UAV.

3.1. Model Formulation

3.1.1. Notations

The proposed heterogeneous UAV route planning is formulated as the MIP model. All the parameters and variables used in the formulation are shown in Tables 1 to 3, most of which has already been explained in Phalapanyakoon and Siripongwutikorn (2021). The model presented here is extended from Phalapanyakoon and Siripongwutikorn (2021) to handle the heterogeneity of UAVs and the payload constraints. **V** is a set of locations $\{0, 1, 2, \ldots, N\}$ with known coordinates with location 0 being the base location. d_{ij} denoted the distance between locations i and j which was calculated from the Euclidean distance. Each location i has payload demand of D_i. The number of UAVs is represented by $N_U = |\mathbf{U}|$. Each UAV u has a battery capacity of Q^u and a payload carrying capacity of P^u. **T** denotes a set of trip indices, with the maximum allowed trips of T_{max}. Different UAVs have different velocities V^u, and the flying time between location i and j is $t_{ij}^u = d_{ij}/V^u$. Based on the area size and mission purpose, the mission time deadline T is given. Each UAV has the overtime of zero if it finishes the last trip by the deadline, or the amount of time exceeded the mission deadline otherwise. The mission overtime is the maximum of UAV overtimes.

Sets/Parameters	Description
V	Set of locations $\{0, 1, 2, \ldots, N\}$
U	Set of UAVs $\{1, 2, \ldots, N_U\}$
T	Set of trips $\{1, 2, \ldots, T_{max}\}$
T_{max}	Number of maximum allowed trips for each UAV
T	Mission time constraint
M	Sufficiently large enough number
θ	The limit percentage of battery capacity of each UAV in each trip
α	Fixed cost of time penalty based on application when exceeding time constraint per second
β^u	The usage cost of UAV u per mission
γ	Fixed cost of energy consumed in mWh
d_{ij}	Distance between location i and j
E_{ij}^u	Energy consumed between location i and j of UAV u
Q^u	The battery capacity of UAV u
V^u	Velocity of UAV u
CR^u	The energy consumption rate of UAV u per metre
RR^u	Recharging rate of UAV u per second
P^u	The payload capacity of UAV u
D_i	The payload demand of location i

Table 1. Sets and parameters

Computation Variables	Description
$RemainEnergy^{tu}$	The remaining energy of UAV *u* at the base location in trip *t*
$TotalTime^{tu}$	The time consumption of UAV *u* in trip *t* including recharging time
$RemainofUAV^{u}$	The remaining battery capacity at last trip of UAV *u*
$LateTime^{u}$	The penalty of overtime of UAV *u*
$RemainStock^{tu}$	The remaining payload of UAV *u* at the base location in trip *t*
$RemainStockofUAV^{u}$	The remaining payload capacity at last trip in each round of UAV *u*
$timeofeachUAV^{u}$	The total time of each UAV *u*
Overtime	The overtime of the mission

Table 2. Computation variables

Decision Variables	Description
x_{ij}^{tu}	Binary equals one if UAV *u* travels from *i* to in trip *t*
$Recharge^{tu}$	Binary equals one if UAV *u* consumes energy more than (Q · *θ*) in trip *t*
$UAVUsed^{u}$	Binary equals one if UAV *u* is deployed in the mission
μ_i^{tu}	The order of location already visited by UAV *u* before entering *i* in trip *t*
$Restock^{tu}$	Binary equals one if UAV *u* consumes payload more than zero in trip *t*

Table 3. Decision variables

3.1.2. Mathematical Model

The proposed problem is a minimization problem with the objective function (1) consisting of three monetary cost components. The first component is the electricity charging cost of each UAV, which corresponds to the sum of all travel distances. The coefficient *γ* is the unit monetary cost of electricity usage. The second component is the cost of using UAVs, where β^u is the cost of using UAV type u. The last component is the penalty of time, which is proportional to the mission overtime. The coefficient α is set based on the importance of the mission deadline. For example, the one used in the mission of distributing fertilizer on a farm should be lower than the one used to deliver vaccines or food in a flood area.

$$Minimize\left(\sum_{t\in\mathbb{T}}\sum_{u\in\mathbb{U}}\sum_{i\in\mathbb{V}}\sum_{j\in\mathbb{V}}\left(\gamma\cdot\left(E_{ij}^{u}\cdot x_{ij}^{tu}\right)\right)+\left(\sum_{u\in\mathbb{U}}\left(\beta^{u}\cdot UAVUsed^{u}\right)\right)+\left(\alpha\cdot Overtime\right)\right) \tag{1}$$

The constraints are divided into four groups to handle different aspects of the problem, including constraints from standard VRP (2-8), constraints to keep track of UAVs energy and battery recharging (9-17), constraints to handle the UAVs maximum carrying capacity and restocking (18-26), and constraints to deal with time spent by each UAV and time penalty calculation (27-32).

$$x_{ii}^{tu}=0,\quad \forall i\in\mathbb{V},\forall t\in\mathbb{T},\forall u\in\mathbb{U} \tag{2}$$

$$\sum_{t\in\mathbb{T}}\sum_{u\in\mathbb{U}}\sum_{j\in\mathbb{V}}x_{ij}^{tu}=1,\quad \forall i\in\mathbb{V} \tag{3}$$

$$\sum_{t\in\mathbb{T}}\sum_{u\in\mathbb{U}}\sum_{i\in\mathbb{V}}x_{ij}^{tu}=1,\quad \forall j\in\mathbb{V} \tag{4}$$

$$\sum_{j\in\mathbb{V}} x_{0j}^{tu} - \sum_{i\in\mathbb{V}} x_{i0}^{tu} = 0, \quad \forall t\in\mathbb{T}, \forall u\in\mathbb{U} \tag{5}$$

$$\sum_{i\in\mathbb{V}} x_{ih}^{tu} - \sum_{j\in\mathbb{V}} x_{hj}^{tu} = 0, \quad \forall h\in\mathbb{V}, \forall t\in\mathbb{T}, \forall u\in\mathbb{U} \tag{6}$$

$$1 \le \mu_i^{tu} \le N, \quad \forall i\in\mathbb{V}, \forall t\in\mathbb{T}, \forall u\in\mathbb{U} \tag{7}$$

$$\mu_i^{tu} - \mu_j^{tu} + N\cdot x_{ij}^{tu} \le N-1, \quad \forall (i,j)\in\mathbb{E}, \forall t\in\mathbb{T}, \forall u\in\mathbb{U} \tag{8}$$

- Constraint (2) prohibits the UAVs from looping back to the same location.
- Constraints (3) and (4) ensure that each UAV visits each location exactly once.
- Constraint (5) requires the UAVs to fly at least one trip and ensure that each UAV begins and returns at the base.
- Constraint (6) assures that each UAV arrives and departs the location *i* when it visits.
- The subtour elimination constraints are formulated on constraints (7) and (8), which are used to arrange all visited locations without the base station (Miller, Tucker & Tucker, 1960).

$$\sum_{i\in\mathbb{V}}\sum_{j\in\mathbb{V}} \left(E_{ij}^{u}\cdot x_{ij}^{tu}\right) \le Q^{u}, \quad \forall t\in\mathbb{T}, \forall u\in\mathbb{U} \tag{9}$$

$$RemainEnergy^{tu} = Q^{u} - \sum_{i\in\mathbb{V}}\sum_{j\in\mathbb{V}} \left(E_{ij}^{u}\cdot x_{ij}^{tu}\right), \quad \text{where } t=1, \forall u\in\mathbb{U} \tag{10}$$

$$\begin{aligned} &RemainEnergy^{tu} = RemainEnergy^{(t-1)u} + Charge^{(t-1)u} - \sum_{i\in\mathbb{V}}\sum_{j\in\mathbb{V}} \left(E_{ij}^{u}\cdot x_{ij}^{tu}\right), \\ &t\in\mathbb{T}\setminus\{1\}, \forall u\in\mathbb{U} \end{aligned} \tag{11}$$

$$\begin{aligned} &-\left(Charge^{tu} + RemainEnergy^{tu} - Q^{u}\right) \le M\cdot(1-Recharge^{tu}), \\ &\forall t\in\mathbb{T}, \forall u\in\mathbb{U} \end{aligned} \tag{12}$$

$$\left(Q^{u}\cdot\theta\right) - RemainEnergy^{tu} \le M\cdot Recharge^{tu}, \quad \forall t\in\mathbb{T}, \forall u\in\mathbb{U} \tag{13}$$

$$RemainEnergy^{tu} + Charge^{tu} \le Q^{u}, \quad \forall t\in\mathbb{T}, \forall u\in\mathbb{U} \tag{14}$$

$$Charge^{tu} \le M\cdot Recharge^{tu}, \quad \forall t\in\mathbb{T}, \forall u\in\mathbb{U} \tag{15}$$

$$RemainEnergy^{tu} - \left(Q^{u}\cdot\theta\right) \le M\cdot(1-Recharge^{tu}), \quad \forall t\in\mathbb{T}, \forall u\in\mathbb{U} \tag{16}$$

$$RemainofUAV^{u} = RemainEnergy^{tu}, \quad \text{where } t=T_{\max}, \forall u\in\mathbb{U} \tag{17}$$

- Constraint (9) ensures that the energy consumed in each trip by the UAV u does not exceed the UAV's battery capacity.
- Constraint (10) states that if the UAV u is deployed on the first trip, the amount of current energy capacity is equal to full capacity minus the energy consumption to the next location.
- For the remaining trips, constraint (11) calculates the remaining energy of each UAV.
- If there is a battery recharging process, the charging amount is equal to the battery capacity minus the remaining energy stated in constraints (12) and (13).
- Constraint (14) ensures that the sum of the remaining energy and the charging amount must not exceed the battery capacity of each UAV.
- Constraints (15) and (16) ensure that there will be no recharging process if the remaining energy is more than the limit capacity of each UAV.
- On the last trip, the remaining energy of each UAV will be calculated by constraint (17).

$$\sum_{i\in\mathbb{V}\setminus\{0\}}\sum_{j\in\mathbb{V}}\left(D_i\cdot x_{ij}^{tu}\right)\le P^u,\quad \forall t\in\mathbb{T},\forall u\in\mathbb{U} \tag{18}$$

$$RemainStock^{tu}=P^u-\sum_{i\in\mathbb{V}\setminus\{0\}}\sum_{j\in\mathbb{V}}\left(D_i\cdot x_{ij}^{tu}\right),\quad \text{where } t=1,\forall u\in\mathbb{U} \tag{19}$$

$$RemainStock^{tu}=RemainStock^{(t-1)u}+Stock^{(t-1)u}-\sum_{i\in\mathbb{V}\setminus\{0\}}\sum_{j\in\mathbb{V}}\left(D_i\cdot x_{ij}^{tu}\right),\\ t\in\mathbb{T}\setminus\{1\},\forall u\in\mathbb{U} \tag{20}$$

$$-\left(Stock^{tu}+RemainStock^{tu}-P^u\right)\le M\cdot(1-Restock^{tu}),\ \forall t\in\mathbb{T},\forall u\in\mathbb{U} \tag{21}$$

$$P^u-RemainStock^{tu}\le M\cdot Restock^{tu},\quad \forall t\in\mathbb{T},\forall u\in\mathbb{U} \tag{22}$$

$$RemainStock^{tu}+Stock^{tu}\le P^u,\quad \forall t\in\mathbb{T},\forall u\in\mathbb{U} \tag{23}$$

$$Stock^{tu}\le M\cdot Restock^{tu},\quad \forall t\in\mathbb{T},\forall u\in\mathbb{U} \tag{24}$$

$$RemainStock^{tu}-P^u\le M\cdot(1-Restock^{tu}),\quad \forall t\in\mathbb{T},\forall u\in\mathbb{U} \tag{25}$$

$$RemainStockofUAV^u=RemainStock^{tu},\quad \text{where } t=T_{\max},\forall u\in\mathbb{U} \tag{26}$$

- Constraint (18) ensures that each UAV's cumulative demands from visiting locations do not exceed its payload carrying capacity on each trip.
- On the first trip, constraint (19) ensures that the current remaining stocks equal the total payload carrying capacity minus the demand of the next location.
- Except the first trip, constraint (20) calculates the remaining stocks of each UAV.
- If there is a restock process, constraints (21) and (22) ensure that the restocking amount equals the payload carrying capacity of the UAV u minus the remaining stock.
- Constraint (23) states that the sum of remaining stock and restocking must not exceed each UAV's payload carrying capacity.
- Constraints (24) and (25) ensure that if the remaining stock is depleted, the restocking process will begin.

- Constraint (26) calculates the remaining stock of the UAV for the last trip of the mission.

$$\left(\sum_{i\in\mathbb{V}}\sum_{j\in\mathbb{V}}(d_{ij}/V^{u})\cdot x_{ij}^{tu}\right)+(Charge^{tu}\cdot RR^{u})-TotalTime^{tu}\leq M\cdot(1-Recharge^{tu}),\quad \forall t\in\mathbb{T},\forall u\in\mathbb{U} \tag{27}$$

$$\left(\sum_{i\in\mathbb{V}}\sum_{j\in\mathbb{V}}(d_{ij}/V^{u})\cdot x_{ij}^{tu}\right)-TotalTime^{tu}\leq M\cdot Recharge^{tu},\quad \forall t\in\mathbb{T},\forall u\in\mathbb{U} \tag{28}$$

$$timeofeachUAV^{u}=\sum_{t\in\mathbb{T}}TotalTime^{tu},\quad \forall u\in\mathbb{U} \tag{29}$$

$$timeofeachUAV^{u}-(Charge^{tu}\cdot RR^{u})-T\leq LateTime^{u},\quad \text{where } t=T_{\max},\forall u\in\mathbb{U} \tag{30}$$

$$LateTime^{u}\leq Overtime,\quad \forall u\in\mathbb{U} \tag{31}$$

$$0\leq LateTime^{u},\quad \forall u\in\mathbb{U} \tag{32}$$

- Constraint (27) calculates the total time of each UAV as equal to the time consumed plus the recharging time if there is a recharging process.
- Constraint (28) accounts for the case of no recharging on each trip, where total time equals time spent visiting locations.
- The total time of each UAV was tracked and recorded by constraint (29).
- Constraint (30) calculates the time penalty of each UAV, and since we do not consider the recharging time of the last trip, the recharging time will be subtracted from the total time.
- Constraints (31) and (32) minimize the overtime of each UAV and ensure that the time penalty is non-negative, respectively.

3.3. Proposed Genetic Algorithm

This section presents the GA procedure to solve the model formulated in Section 3.2, which is based on standard GA procedure originally proposed by Holland (1975). First, the population of feasible solutions is generated, and individual solutions in the population are repaired, evaluated for their fitness, and sorted by their fitness score. Then, the genetic operators, including selection, crossover, and mutation, are iteratively applied to the population for a predefined number of generations to find the fittest solution. The following subsections describe the chromosome encoding and the details of each genetic operator.

3.3.1. Chromosome Encoding

The chromosome is an integer string with two parts – UAV part and Route part. The UAV part contains types of UAVs used while the route part contains the routes of UAVs. Because we assume that every UAV has the payload capacity at least the maximum of location demands, the number of UAVs needed is at most the number of locations $|\mathbf{V}|$. So, the length of UAV part is set to $|\mathbf{V}|$, where the value of the i^{th} gene is the type of UAV i and zero if UAV i is not used. An example of the chromosome is shown in Figure 2. Suppose the number of locations is 11 and the number of available UAVs is four. In the example, three UAVs are deployed – UAV 1 is type 1, UAV 2 is type 4, and UAV 3 is type 2. The route part in the chromosome contains UAV routes corresponding to the UAVs in the UAV part. The UAV routes are highlighted in green in Figure 2. Each gene in a UAV route denotes a location, and the route must start and end with gene one (the base station location). Value zeros are used to separate individual UAV routes. Unused genes following the last UAV route are set to one. So, two consecutive genes of ones in the second part of chromosome indicates the end of UAV routes. In the worst case, each UAV visits only one location and returns to the base station, which takes three genes in the chromosome, and there are at most $|\mathbf{V}|$-1 of gene zeros to separate the UAV routes. So, the chromosome length of $|\mathbf{V}| + 3|\mathbf{V}| + (|\mathbf{V}|-1) \sim 5|\mathbf{V}|$ is sufficient to handle all input scenarios.

The number of maximum allowed trips plays an important role in finding optimal solutions. UAVs may not visit all the locations if the number of maximum allowed trips is set too low while too high values unnecessarily increase the computation time. So, the number of maximum allowed trips can be determined from the worst-case scenario based on the lowest battery capacity and lowest payload capacity of the UAVs. First, we apply the Clarke and Wright saving heuristic (Clarke & Wright, 1964) to a single-depot VRP to estimate the single-trip route covering all the locations. From the route, the number of trips taken by a single UAV with the lowest battery capacity and the number of trips taken by a single UAV with the lowest payload capacity over such a route are determined. Then, the number of maximum allowed trip is the maximum of the two. The algorithm to estimate the number of maximum allowed trips is shown in Figure 3.

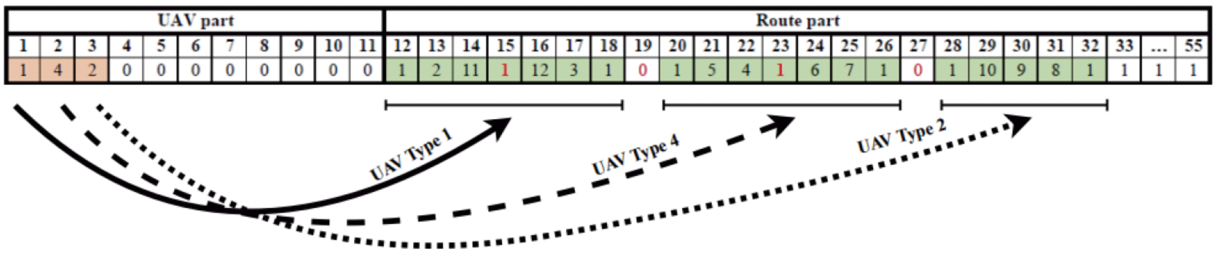


Figure 2. Example chromosomes $|\mathbf{V}|$ of = 11 locations with two maximum allowed trips and three UAVs being used

```
Algorithm  Maximum Allowed Trip per UAV Calculation
Input : Set of locations and demands, Lowest UAV battery capacity, Lowest UAV payload capacity
Output: Number of maximum allowed trips
 1: Estimate the shortest route visiting all locations using Clarke and Wright savings heuristic.
 2: Determine the total energy to complete the route.
 3: Determine $Nb$ the number of trips taken on such route if the UAV with lowest battery capacity is
    used.
 4: Determine $Np$ the number of trips taken on such route if the UAV with lowest payload capacity
    is used.
 5: Set the number of maximum allowed trips to $\max(Nb, Np)$.
```

Figure 3. Algorithm to determine the number of maximum allowed trips.

3.3.2. Population Generation

Due to the complex constraints, it is challenging to randomly generate feasible chromosome for the initial population. We design two methods to generate a population of feasible solutions – Exhaustive method and Clustering method. The exhaustive method starts by initializing the UAV part of the chromosome as a permutation of all the locations. Then, the type of UAV 1 is randomly selected and put in the first bit of the UAV part. Initially, the chromosome contains only a single UAV using a single-trip route that covers all the locations, and the repair process is applied to ensure feasibility. Based on the battery capacity and payload capacity of the type of UAV 1, locations in the route are visited one by one until either the battery capacity or the payload capacity of UAV 1 is depleted. Gene 1 is then inserted after the last visited location in the chromosome to signify a new trip and the process continues with full battery and payload capacities. If the number of maximum allowed trips is reached, gene 0 is inserted to indicate the next UAV route, and the type of UAV 2 is randomly selected to put in the second bit of the UAV part. The route of UAV 2 continues from the remaining unvisited locations and so on. A feasible chromosome is obtained if the UAV routes cover all locations within the number of maximum allowed trips and the number of available UAVs. Otherwise, the chromosome generation process starts over until the desired number of feasible chromosomes in the population is obtained. The algorithm of the exhaustive method is given in Figure 4(a).

The clustering population initialization method applies K-Mean clustering based on the location coordinates to cluster the locations into the number of groups equal to the number of UAVs used. Then, each UAV route is initialized as the permutation of locations in their corresponding cluster so that each UAV will only visit the assigned locations from the clustering. The repair process like in the exhaustive method is then applied to generate feasible solutions. The algorithm of the clustering method is given in Figure 4(b). The major difference between the two methods is that the exhaustive method forces the first UAV to visit as many locations as possible before deploying an additional UAV. On the other hand, the clustering method generates solutions with more UAVs used. To increase the population diversity, both methods are used together where half of the population is generated from each method.

```
Algorithm  Exhaustive Initialization
Input : Number of chromosomes required, Locations and demands, Number of UAVs and capacities,
Number of maximum allowed trips
Output: Chromosomes with the required number
 1: Permute the locations
 2: Randomly selected the type of UAV 1 and put in the first bit of the UAV part
 3: Based on the type of UAV 1's capabilities, locations in route are visitied one by one until either
    the battery or payload capacity of UAV is depleted.
 4: Insert 1 to chromosome after the last visited location to signify new trip and continue the visiting
    process.
 5: Insert 0 to chromosome if the maximum allowed trips is reached and randomly selected the type
    of UAV 2 to put in the second bit of UAV part.
 6: The route of UAV 2 continues from the remaining unvisited locations and so on
 7: Verify the obtained chromosome
```

(a) Exhaustive method

```
Algorithm  Clustering-based Initialization
Input : Number of chromosomes required, Locations and demands, Number of UAVs and capacities,
Number of maximum allowed trips
Output: Chromosomes with the required number
 1: Apply K-Mean clustering based on the location coordinates to cluster the locations into the number
    of groups equal to the number of UAVs used
 2: Permute the locations as UAV route in their corresponding cluster, which each UAV will only visit
    the assigned locations from the clustering.
 3: Apply exhaustive method to each cluster as repairing process to generate feasible solution
 4: Verify the obtained chromosome
```

(b) Clustering method

Figure 4. Population initialization methods

3.3.3. Fitness Score

The quality of feasible solutions is evaluated by using the fitness score. Since GA attempts to find solutions with the highest fitness score, we use the fitness score that is the inverse of the cost function in (33). In particular,

$$g(x) = (\gamma \cdot RechargingCost) + (\beta \cdot UAVUsageCost) + (\alpha \cdot Overtime) \tag{33}$$

$$FitnessScore = (1 / g(x)) \tag{34}$$

The cost components in (33) can be readily determined from the chromosome as shown in Figure 5. The UAV usage cost is calculated from the first part of the chromosome. For each UAV route in the second part, the recharging cost and the overtime are respectively determined from the energy usage per trip and the total traveling time.

```
Algorithm  Fitness Score Evaluation
Input : Chromosome, Energy unit cost (γ), UAV unit rental cost (β), Time penalty factor (α)
Output: Fitness score
 1: Calculate UAV usage cost from UAV part
 2: If a single 1 is encountered in chromosome, determine recharging process and calculate cumulative
    energy and time of each trip
 3: If a single 0 is encountered in chromosome, calculate cumulative energy and time of each UAV
 4: If two adjacent ones are encountered, calculate total energy usage and the overtime of each UAV
 5: Calculate Fitness score
```

Figure 5. Algorithm for fitness score evaluation

3.3.4. Selection Operation

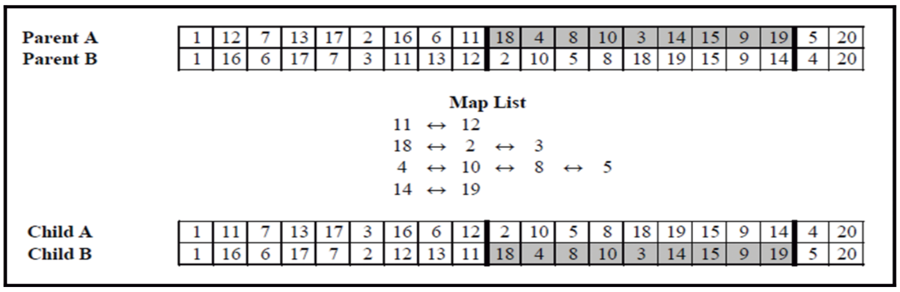
Parent selection is very important in the convergence rate and contributes to the improvement of fitness scores in successive generations. The standard roulette wheel selection is adopted in this work, whereby a fitter individual has a greater probability of being selected. For example, the selection percentage is set to 0.9, which means we select the top 90% of solutions in the current population to produce the next generation. The effect of the selection probability will be studied in the parameter sensitivity section.

3.3.5. Crossover Operation

Crossover operation generates offspring chromosomes from two parent chromosomes to explore the solution space. We adopt Partially-Mapped Crossover (PMX) (Goldberg & Lingle., 1985) for the crossover operator in this work, as shown in Figure 6. This crossover technique randomly selects two parts, one from each parent to swap. The procedure starts by selecting two adjacent chromosomes as the parents and then extracting the UAV part. The repair process is applied by using the route construction (exhaustive and clustering methods) based on the extracted UAV part and routes from PMX crossover and the one with the better fitness score is selected. PMX has been shown to give relatively good results compared to other crossover operations (Kumar, Karambir and Kumar, 2012). Furthermore, this crossover technique allows offspring to inherit parts of the parent chromosome such that parts of the route are reserved. An example of PMX is shown in Figure 7.

```
Algorithm  Crossover Operation
Input : Chromosome ,Locations and demands, Number of UAVs and capacities, Number of maximum
allowed trips, Energy unit cost (γ), UAV unit rental cost (β), Time penalty factor (α), Percentage of
crossover
Output: Chromosome
 1: Initialize parameters
 2: Select two adjacent chromosomes as parent
 3: Extract the UAV part of both parent
 4: Extract the routes from both parents and perform PMX crossover
 5: Perform route construction (Exhaustive and Clustering Method) based on the extracted UAV part
    and routes from PMX crossover
 6: Compare the fitness score of both methods and select the better one
 7: Verify the integrity of all feasible solutions
 8: Return Chromosomes
```

Figure 6. Algorithm for crossover operation



Parent A	1	12	7	13	17	2	16	6	11	18	4	8	10	3	14	15	9	19	5	20
Parent B	1	16	6	17	7	3	11	13	12	2	10	5	8	18	19	15	9	14	4	20

Map List

11 ↔ 12
18 ↔ 2 ↔ 3
4 ↔ 10 ↔ 8 ↔ 5
14 ↔ 19

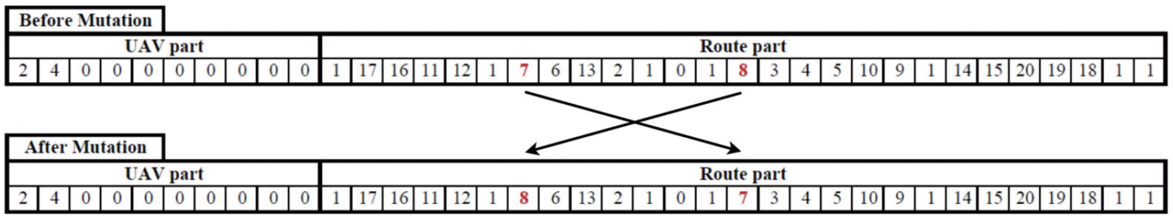
Child A	1	11	7	13	17	3	16	6	12	2	10	5	8	18	19	15	9	14	4	20
Child B	1	16	6	17	7	2	12	13	11	18	4	8	10	3	14	15	9	19	5	20

Figure 7. Example of PMX crossover

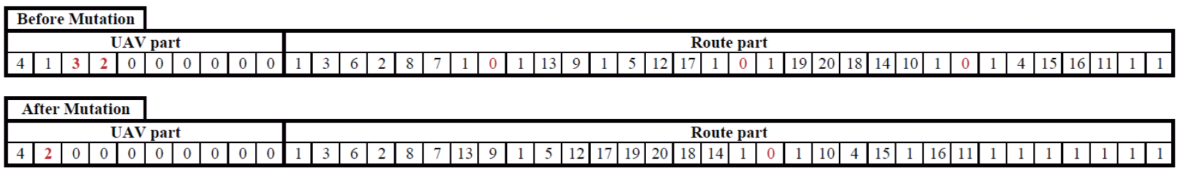
3.3.6. Mutation Operation

Mutation enhances the quality of feasible solutions obtained from the crossover so that the solutions have diverse characteristics and premature convergence to local optimum can be avoided. In this work, mutation is applied to both UAV part and route part as follows:

- **Mutation of Route part** Randomly selected two locations in the route part and swapped them with the pre-specified mutation probability (called the mutation rate). An example of route mutation is shown in Figure 8(a). If the swapped locations violate the battery or payload capacity constraints, no mutation is applied.
- **Mutation of UAV part** Genes in UAV part (except the first one) are randomly replaced by values (including zero) other than the current one. Value zero means the UAV is removed from the mission. If the gene is replaced by zero, all the remaining genes will be set to zero. Therefore, several UAVs can be removed in the mutation. Then, the route repair is performed by applying the clustering method discussed earlier to obtain a feasible chromosome. An example of UAV mutation is shown in Figure 8(b).



(a) Chromosomes before and after route mutation



(b) Chromosomes before and after UAV mutation

Figure 8. Mutations of routes and UAVs deployed

4. Results and Analysis

A set of experiments are conducted to examine the computational performance of both exact and proposed GA methods. Square grid networks, or Manhattan networks, with various sizes as well as a real terrain map are used for test problems. The location demands are randomly generated from a discrete uniform distribution in [1, 10]. The vertical and horizontal distance between two adjacent grid points is 1000-meter for square grid networks. The real terrain map is excerpted from the city of Nong Khai province, Thailand. The map is first downloaded from Google Maps and manually discretized into 25 by 15 grid points with a 130-meter distance between adjacent grid points. Thirty points, which are major city locations, are manually specified as the grid points to visit as shown in Figure 9. Two-level factorial design is applied to study effects of the GA parameters on the solution quality. The exact method uses the branch and bound algorithm from IBM ILOG CPLEX Optimization Studio Version 12.10.0.0 (International Business Machine, 2019) running on Ubuntu 18.04 LTS with an Intel® Xeon® 2.6 GHz 8-core E5-2640 V3 CPU and a VMware 500 GB hard disk. For the computational resource constraints of the exact method, we limit the CPU running time and hard disk resources to 24 hours and 500 GB respectively. The model solving will be prematurely terminated if either one of these conditions is met, and the best current results will be taken as the final solutions. The GA method is solved by using MATLAB version R2021b running on the same machine as above. In each test problem, the algorithm will be stopped if it reaches its maximum number of population generations. The algorithm is repeated 10 times with different initial populations in each test problem and the best result is taken.

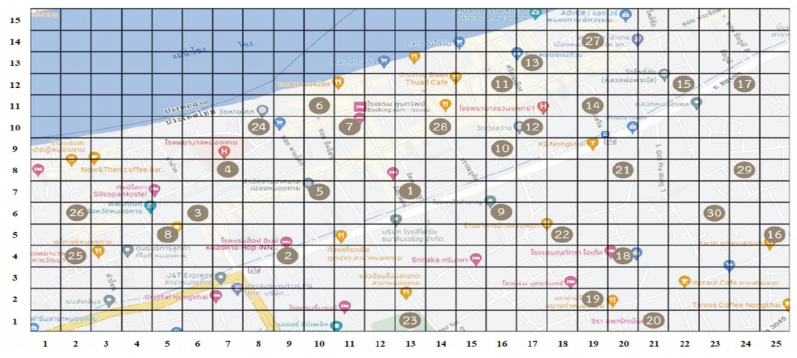


Figure 9. Example of real terrain case with 30 locations (Google, 2022)

Four types of UAVs are considered, with their specifications shown in Table 4. Note that *β*, *Q*, *V*, *CR*, *RR*, and *P* are specific to UAV types, and each UAV type can carry the parcel half of its weight. Besides the number of grid points in test problems, the model parameters that are varied in the experiments include α (the penalty cost of mission overtime), the mission time, and number of available UAVs, as in Table 5. The energy unit cost in baht per mWh (*γ*) is fixed to 0.0039 in the experiments.

No	UAV	Price / Day (Thai Baht) (β)	Capacity (mwh) (Q)	Speed (m/s) (V)	Energy consumption Rate (wh/m) (CR)	Battery recharging Rate (s/wh) (RR)	Payload Capacity (P)
1	DJI Spark	600	16872	13.85824	1.268198559	0.1849217639	15
2	DJI Mavic Air	907.01	27431.25	19	1.145833333	0.1203007519	21
3	DJI Phantom 4 Pro	1360.85	89224	20.1168	2.464054367	0.04707253654	69
4	Mavic Pro	1015.84	43662	17.8816	1.507239389	0.1099354129	36

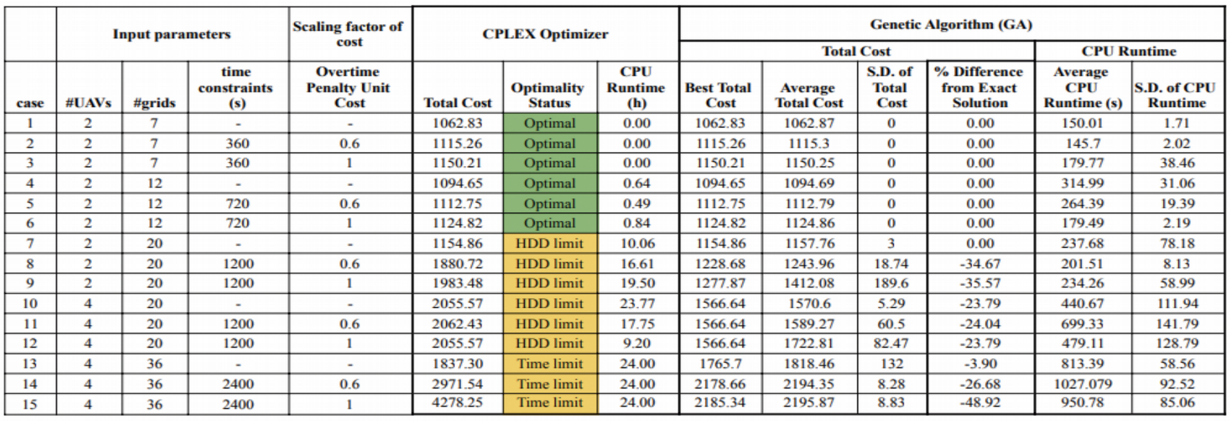
Table 4. UAV specifications (CameraLens Rentals, 2022)

Varied Parameters	Values
α – Overtime penalty unit cost	None, 0.6, 1 baht/second
N - Number of grid points	7, 12, 20, 36 for test problem cases 12, 15 for real-terrain cases
N_v - Number of available UAVs	2, 4
T - Mission time	None, 360, 720, 1200, 2400 seconds

Table 5. Varied model parameters

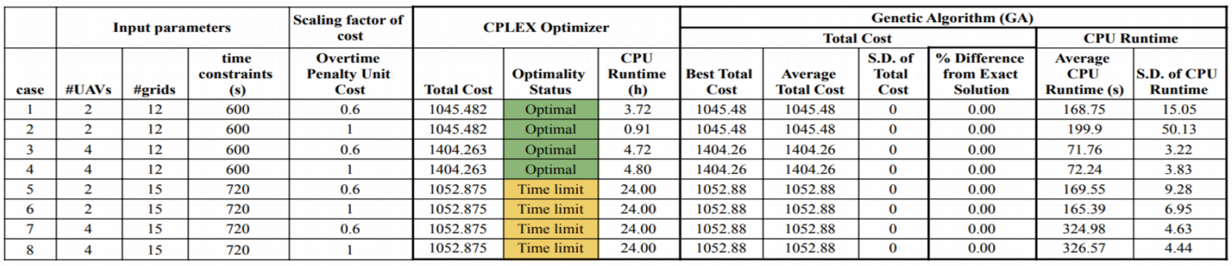
4.1. Comparison with Exact Solutions

Solutions obtained from the exact method and the GA method are compared in the grid-network problems and real-terrain cases to verify the integrity of the GA method. The GA parameters are set to the following population size of 1000, number of maximum generations of 1000, selection probability of 90%, crossover rate of 95%, route mutation rate of 30%, and UAV mutation rate of 30%. The runtime results of grid-network problems are shown in Table 6. For the GA method, the best, average, and standard deviation of objective values are shown, as well as the percentage difference from the result obtained from the exact method. The GA method can obtain optimal solutions as in the exact method for cases 1 to 6, where the number of UAVs used and the network size are relatively small (2 UAVs and 12 grid points). Additionally, the GA method can find the optimal solutions regardless of the initial populations as can be seen from zero standard deviation of the objective values. As the number of UAVs or the network size increases (cases 8-15), the exact method cannot find optimal solutions due to the limits of computational resources but the GA method can find solutions whose objective values are 24% to 49% better and take several orders of magnitude less computation time (a few minutes vs. almost a day).



	Input parameters			Scaling factor of cost	CPLEX Optimizer			Genetic Algorithm (GA)					
								Total Cost				CPU Runtime	
case	#UAVs	#grids	time constraints (s)	Overtime Penalty Unit Cost	Total Cost	Optimality Status	CPU Runtime (h)	Best Total Cost	Average Total Cost	S.D. of Total Cost	% Difference from Exact Solution	Average CPU Runtime (s)	S.D. of CPU Runtime
1	2	7	-	-	1062.83	Optimal	0.00	1062.83	1062.87	0	0.00	150.01	1.71
2	2	7	360	0.6	1115.26	Optimal	0.00	1115.26	1115.3	0	0.00	145.7	2.02
3	2	7	360	1	1150.21	Optimal	0.00	1150.21	1150.25	0	0.00	179.77	38.46
4	2	12	-	-	1094.65	Optimal	0.64	1094.65	1094.69	0	0.00	314.99	31.06
5	2	12	720	0.6	1112.75	Optimal	0.49	1112.75	1112.79	0	0.00	264.39	19.39
6	2	12	720	1	1124.82	Optimal	0.84	1124.82	1124.86	0	0.00	179.49	2.19
7	2	20	-	-	1154.86	HDD limit	10.06	1154.86	1157.76	3	0.00	237.68	78.18
8	2	20	1200	0.6	1880.72	HDD limit	16.61	1228.68	1243.96	18.74	-34.67	201.51	8.13
9	2	20	1200	1	1983.48	HDD limit	19.50	1277.87	1412.08	189.6	-35.57	234.26	58.99
10	4	20	-	-	2055.57	HDD limit	23.77	1566.64	1570.6	5.29	-23.79	440.67	111.94
11	4	20	1200	0.6	2062.43	HDD limit	17.75	1566.64	1589.27	60.5	-24.04	699.33	141.79
12	4	20	1200	1	2055.57	HDD limit	9.20	1566.64	1722.81	82.47	-23.79	479.11	128.79
13	4	36	-	-	1837.30	Time limit	24.00	1765.7	1818.46	132	-3.90	813.39	58.56
14	4	36	2400	0.6	2971.54	Time limit	24.00	2178.66	2194.35	8.28	-26.68	1027.079	92.52
15	4	36	2400	1	4278.25	Time limit	24.00	2185.34	2195.87	8.83	-48.92	950.78	85.06

Table 6. Runtime results of exact and GA methods in the grid-network test problems



	Input parameters			Scaling factor of cost	CPLEX Optimizer			Genetic Algorithm (GA)					
								Total Cost				CPU Runtime	
case	#UAVs	#grids	time constraints (s)	Overtime Penalty Unit Cost	Total Cost	Optimality Status	CPU Runtime (h)	Best Total Cost	Average Total Cost	S.D. of Total Cost	% Difference from Exact Solution	Average CPU Runtime (s)	S.D. of CPU Runtime
1	2	12	600	0.6	1045.482	Optimal	3.72	1045.48	1045.48	0	0.00	168.75	15.05
2	2	12	600	1	1045.482	Optimal	0.91	1045.48	1045.48	0	0.00	199.9	50.13
3	4	12	600	0.6	1404.263	Optimal	4.72	1404.26	1404.26	0	0.00	71.76	3.22
4	4	12	600	1	1404.263	Optimal	4.80	1404.26	1404.26	0	0.00	72.24	3.83
5	2	15	720	0.6	1052.875	Time limit	24.00	1052.88	1052.88	0	0.00	169.55	9.28
6	2	15	720	1	1052.875	Time limit	24.00	1052.88	1052.88	0	0.00	165.39	6.95
7	4	15	720	0.6	1052.875	Time limit	24.00	1052.88	1052.88	0	0.00	324.98	4.63
8	4	15	720	1	1052.875	Time limit	24.00	1052.88	1052.88	0	0.00	326.57	4.44

Table 7. Runtime results of exact and GA methods in test problems (real-terrain cases)

The cost and runtime results for the real-terrain network with selected 12 and 15 grid points are shown in Table 7. Like in the grid-network cases, the GA method can find the same solutions as the exact method with significantly less computational time. However, the solution optimality is not guaranteed in cases 5-8 since the exact method is prematurely terminated by the 24-hour time limit.

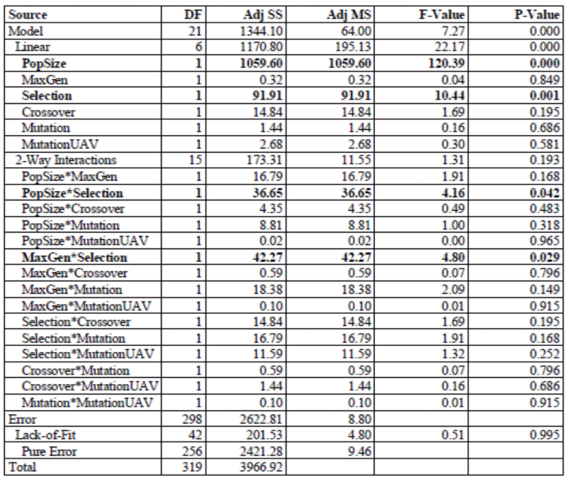
4.2. Sensitivity of GA Parameters

To assess the sensitivity of GA parameters and determine appropriate GA parameter values to use, we conduct a two-level factorial design and investigate the main and interaction effects of GA parameters on the average total cost from ten replications on two problem sizes – 12 and 36 grid points with the mission deadline of 720 and 2400 seconds respectively. The range of GA parameters in the experimental design is shown in Table 8.

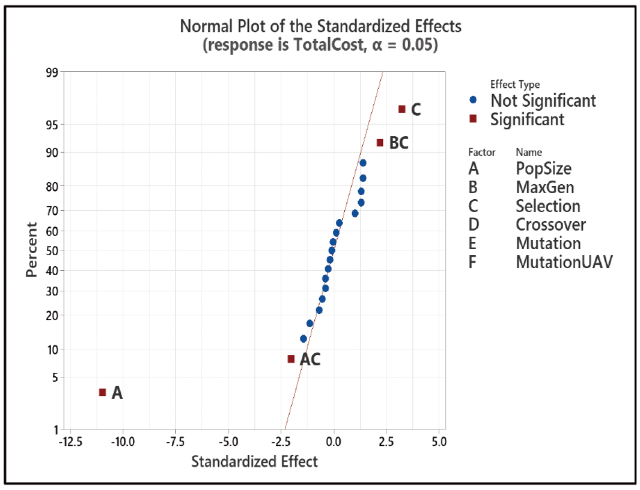
Experimental Domain of 2^6 Full Factorial Design		
Parameters	Low Level	High Level
Population Size	200	1000
Max Generation	200	1000
Selection Rate	0.6	0.9
Crossover Rate	0.6	0.9
Mutation Rate	0.1	0.4
MutationUAV Rate	0.1	0.4

Table 8. Range of GA parameters used in two-level factorial design

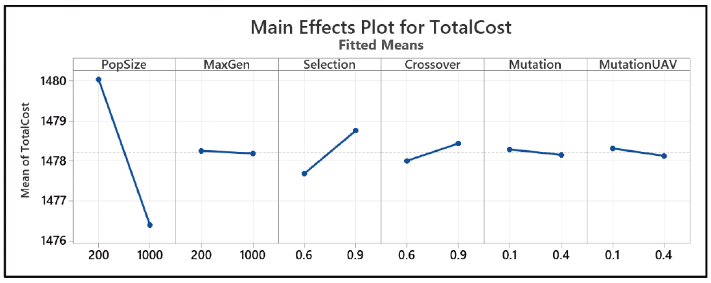
The normal plot, main effects, and interactions plots from the experiments of 12 and 36 grid points are shown in Figures 9 and 10 respectively. The normal plot in Figure 9(a) shows that for the 12 grid-point network, population size, selection rate, and the interaction of maximum generation and selection rate are all significant factors. From the main effect plots in Figure 9(b), the population size of 1000 and the selection rate of 0.6 can be chosen as the default values to be used. Due to the interaction between selection rate and the maximum generation in Figure 9(c), we recommend using the selection rate of 0.6 and the maximum generation of 1000. Since all the other parameters are not significant over the ranges studied, we can arbitrarily set their values to be in the ranges.



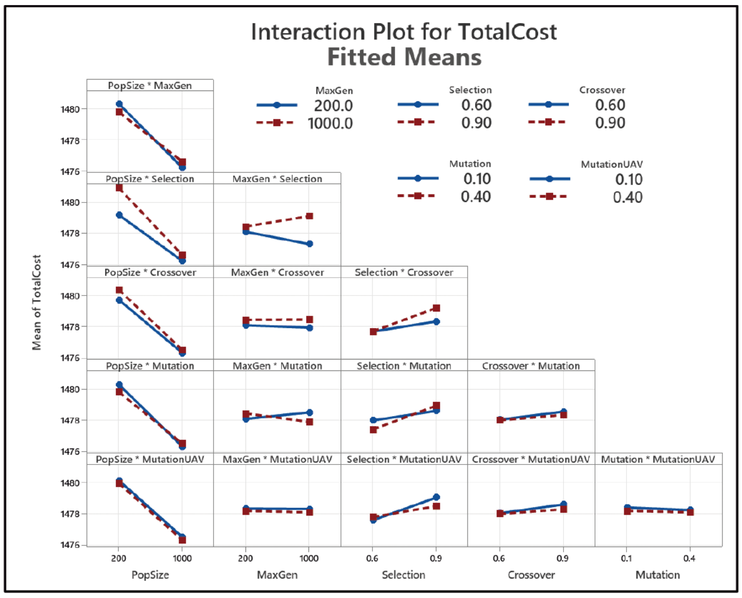
Source	DF	Adj SS	Adj MS	F-Value	P-Value
Model	21	1344.10	64.00	7.27	0.000
Linear	6	1170.80	195.13	22.17	0.000
PopSize	**1**	**1059.60**	**1059.60**	**120.39**	**0.000**
MaxGen	1	0.32	0.32	0.04	0.849
Selection	**1**	**91.91**	**91.91**	**10.44**	**0.001**
Crossover	1	14.84	14.84	1.69	0.195
Mutation	1	1.44	1.44	0.16	0.686
MutationUAV	1	2.68	2.68	0.30	0.581
2-Way Interactions	15	173.31	11.55	1.31	0.193
PopSize*MaxGen	1	16.79	16.79	1.91	0.168
PopSize*Selection	**1**	**36.65**	**36.65**	**4.16**	**0.042**
PopSize*Crossover	1	4.35	4.35	0.49	0.483
PopSize*Mutation	1	8.81	8.81	1.00	0.318
PopSize*MutationUAV	1	0.02	0.02	0.00	0.965
MaxGen*Selection	**1**	**42.27**	**42.27**	**4.80**	**0.029**
MaxGen*Crossover	1	0.59	0.59	0.07	0.796
MaxGen*Mutation	1	18.38	18.38	2.09	0.149
MaxGen*MutationUAV	1	0.10	0.10	0.01	0.915
Selection*Crossover	1	14.84	14.84	1.69	0.195
Selection*Mutation	1	16.79	16.79	1.91	0.168
Selection*MutationUAV	1	11.59	11.59	1.32	0.252
Crossover*Mutation	1	0.59	0.59	0.07	0.796
Crossover*MutationUAV	1	1.44	1.44	0.16	0.686
Mutation*MutationUAV	1	0.10	0.10	0.01	0.915
Error	298	2622.81	8.80		
Lack-of-Fit	42	201.53	4.80	0.51	0.995
Pure Error	256	2421.28	9.46		
Total	319	3966.92			



(a) Analysis of variance and normal plot



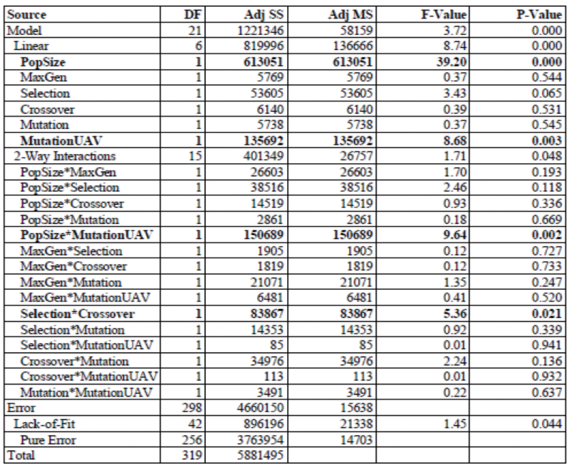
(b) Main effects plot



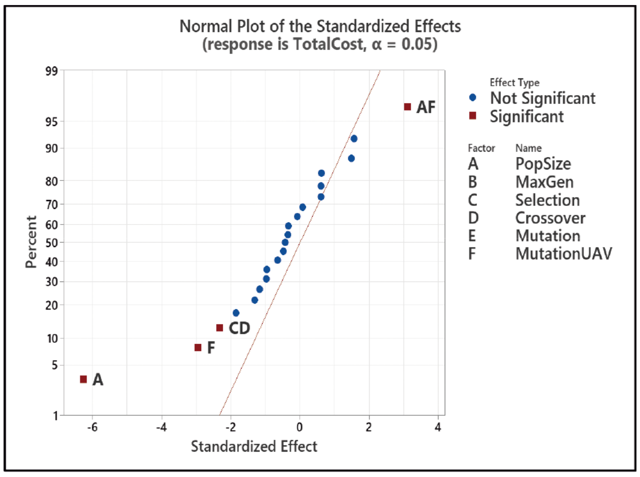
(c) Interactions plot

Figure 9. Results from two-level factorial experiment of the 12 grid-point network

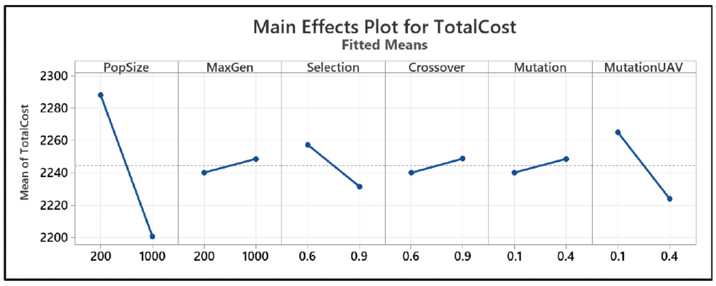
For the case of 36 grid points, the significant factors are shown in Figure 10(a). The main effects plot in Figure 10(b) shows that population size and UAV mutation rate are both significant, and their default values can be set to 1000 and 0.4. From the ANOVA table, we can observe that the interaction between selection rate and crossover rate is significant. It is suggested using the selection rate of 0.9 and the crossover rate of 0.9 to achieve a lower totalcost as shown Figure 10(c). Similar to the previous case, the other GA parameters are not significant and using arbitrary values selected from the specified ranges should only have negligible impact on the results.



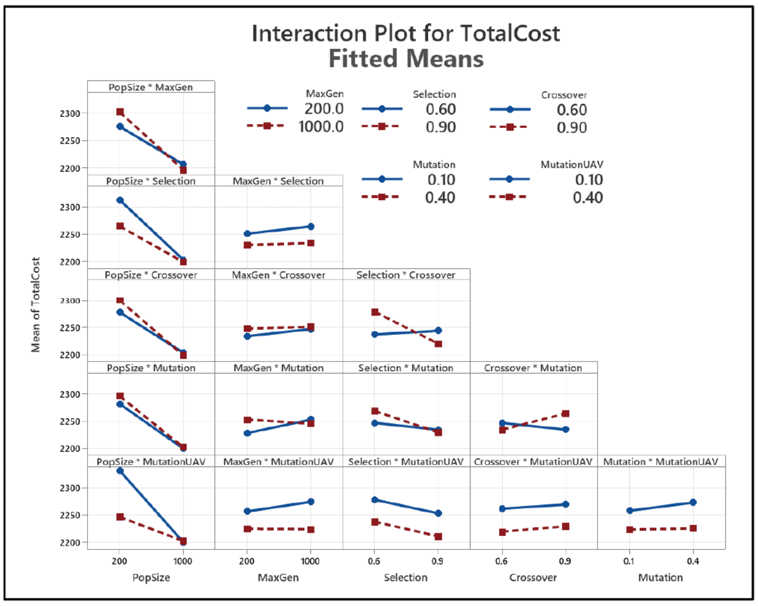
Source	DF	Adj SS	Adj MS	F-Value	P-Value
Model	21	1221346	58159	3.72	0.000
Linear	6	819996	136666	8.74	0.000
PopSize	**1**	**613051**	**613051**	**39.20**	**0.000**
MaxGen	1	5769	5769	0.37	0.544
Selection	1	53605	53605	3.43	0.065
Crossover	1	6140	6140	0.39	0.531
Mutation	1	5738	5738	0.37	0.545
MutationUAV	**1**	**135692**	**135692**	**8.68**	**0.003**
2-Way Interactions	15	401349	26757	1.71	0.048
PopSize*MaxGen	1	26603	26603	1.70	0.193
PopSize*Selection	1	38516	38516	2.46	0.118
PopSize*Crossover	1	14519	14519	0.93	0.336
PopSize*Mutation	1	2861	2861	0.18	0.669
PopSize*MutationUAV	**1**	**150689**	**150689**	**9.64**	**0.002**
MaxGen*Selection	1	1905	1905	0.12	0.727
MaxGen*Crossover	1	1819	1819	0.12	0.733
MaxGen*Mutation	1	21071	21071	1.35	0.247
MaxGen*MutationUAV	1	6481	6481	0.41	0.520
Selection*Crossover	**1**	**83867**	**83867**	**5.36**	**0.021**
Selection*Mutation	1	14353	14353	0.92	0.339
Selection*MutationUAV	1	85	85	0.01	0.941
Crossover*Mutation	1	34976	34976	2.24	0.136
Crossover*MutationUAV	1	113	113	0.01	0.932
Mutation*MutationUAV	1	3491	3491	0.22	0.637
Error	298	4660150	15638		
Lack-of-Fit	42	896196	21338	1.45	0.044
Pure Error	256	3763954	14703		
Total	319	5881495			



(a) Analysis of variance and normal plot



(b) Main effects plot



(c) Interactions plot

Figure 10. Results from two-level factorial experiment of the 36 grid-point network

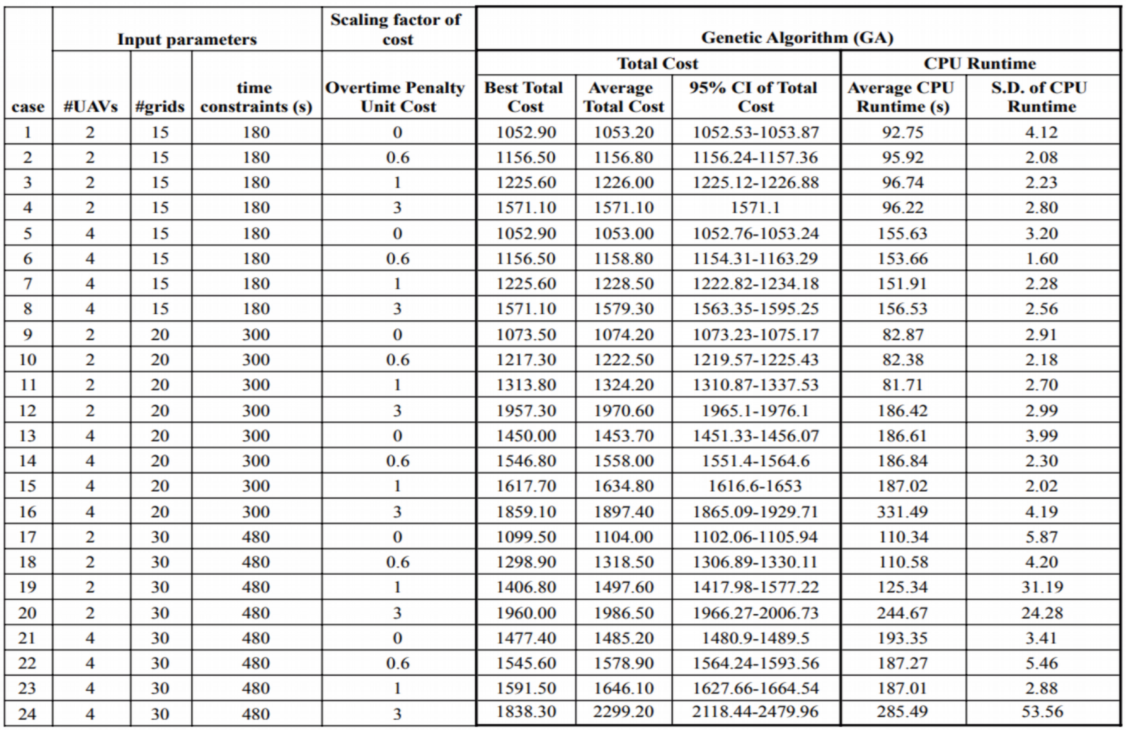
The experimental results above suggest the default values of the GA parameters can be used for other problems or used as starting points for later tuning as shown in Table 9. The population size and the maximum generation can be both set to 1,000. In fact, the larger the values for these two parameters the better the total cost because the larger population size and number of generations will provide more chances for GA to discover the optimal solution. The selection rate of 0.9 appears to give low total costs in both 12 and 36 grid-point networks while the crossover rate of 0.9 provides consistently good results. MutationUAV rate can be set to 0.4 based on the studied, while mutation rate is set to 0.1.

Parameters	Recommended Value
Population Size	1000
Max Generation	1000
Selection Rate	0.9
Crossover Rate	0.9
Mutation Rate	0.1
MutationUAV Rate	0.4

Table 9. Summary of recommended GA parameter values

4.3. Real Terrain Scenarios

From the default GA parameter values obtained from the previous section, we apply them to real-terrain networks with 15, 20, and 30 grid points to assess the robustness of the proposed GA method and the GA parameter values. The number of available UAVs, time penalty constraints, and the time penalty cost are varied and the results of the experiment are shown in Table 10. For 15 and 20 grid points, the best total costs are close to the lower 95% confidence limits of the mean total cost and the standard deviation of the total costs are quite small. However, when the problem size gets large as in the cases of 30 grid points, there exists more variability in the resulting total costs, which can be seen in wider 95% confidence intervals of the mean total cost. Many of the best total costs are much less than the lower 95% confidence limits, implying that the GA method is capable of exploring different regions of the solution space for solutions.



	Input parameters			Scaling factor of cost	Genetic Algorithm (GA)				
					Total Cost			CPU Runtime	
case	#UAVs	#grids	time constraints (s)	Overtime Penalty Unit Cost	Best Total Cost	Average Total Cost	95% CI of Total Cost	Average CPU Runtime (s)	S.D. of CPU Runtime
1	2	15	180	0	1052.90	1053.20	1052.53-1053.87	92.75	4.12
2	2	15	180	0.6	1156.50	1156.80	1156.24-1157.36	95.92	2.08
3	2	15	180	1	1225.60	1226.00	1225.12-1226.88	96.74	2.23
4	2	15	180	3	1571.10	1571.10	1571.1	96.22	2.80
5	4	15	180	0	1052.90	1053.00	1052.76-1053.24	155.63	3.20
6	4	15	180	0.6	1156.50	1158.80	1154.31-1163.29	153.66	1.60
7	4	15	180	1	1225.60	1228.50	1222.82-1234.18	151.91	2.28
8	4	15	180	3	1571.10	1579.30	1563.35-1595.25	156.53	2.56
9	2	20	300	0	1073.50	1074.20	1073.23-1075.17	82.87	2.91
10	2	20	300	0.6	1217.30	1222.50	1219.57-1225.43	82.38	2.18
11	2	20	300	1	1313.80	1324.20	1310.87-1337.53	81.71	2.70
12	2	20	300	3	1957.30	1970.60	1965.1-1976.1	186.42	2.99
13	4	20	300	0	1450.00	1453.70	1451.33-1456.07	186.61	3.99
14	4	20	300	0.6	1546.80	1558.00	1551.4-1564.6	186.84	2.30
15	4	20	300	1	1617.70	1634.80	1616.6-1653	187.02	2.02
16	4	20	300	3	1859.10	1897.40	1865.09-1929.71	331.49	4.19
17	2	30	480	0	1099.50	1104.00	1102.06-1105.94	110.34	5.87
18	2	30	480	0.6	1298.90	1318.50	1306.89-1330.11	110.58	4.20
19	2	30	480	1	1406.80	1497.60	1417.98-1577.22	125.34	31.19
20	2	30	480	3	1960.00	1986.50	1966.27-2006.73	244.67	24.28
21	4	30	480	0	1477.40	1485.20	1480.9-1489.5	193.35	3.41
22	4	30	480	0.6	1545.60	1578.90	1564.24-1593.56	187.27	5.46
23	4	30	480	1	1591.50	1646.10	1627.66-1664.54	187.01	2.88
24	4	30	480	3	1838.30	2299.20	2118.44-2479.96	285.49	53.56

Table 10. Results of real-terrain network with different problem sizes

5. Conclusion

In this paper, a mathematical model for route planning of heterogeneous UAVs under recharging and mission time with payload carrying constraints and multi-trips has been formulated and solved by using a genetic algorithm. Unlike existing works, the proposed work includes the recharging period into the mission time based on the remaining energy of the UAV in each trip and the payload carrying capacity of different types of UAVs. The objective function is the monetary cost, which is the sum of the recharging cost of each UAV, the UAV rental cost, and the cost of mission overtime. The problem is formulated as a mixed-integer programming model and a genetic algorithm is developed to solve the problem. Our results show that the exact solution method using branch-and-bound is only feasible for small problem sizes with a few UAVs and locations while the proposed GA method can obtain the optimal solutions as the exact method but with remarkably less running time. For larger problem sizes, the GA method can find solutions in a few minutes, with 29% to 49% lower costs than suboptimal solutions obtained from the exact method. The sensitivity of GA parameters has been studied by using two-level factorial experiments. The population size and maximum generation are significant, and there exists an interaction between the selection rate and the crossover rate. Appropriate default values of the GA parameters are identified and then applied to larger problem sizes of real terrain networks. The results show that when the problem size becomes large and complex, the GA method can explore the solution space under different initial populations to find solutions. For further research, the problem assumptions could be relaxed to handle more flexible constraints. For example, UAVs may not start with full battery capacity, the network can have multiple base stations, and individual location demands are allowed to be larger than the UAV payload capacity so that some locations must be visited more than once.

Declaration of Conflicting Interests

The authors declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

This research is supported by the Petchra Pra Jom scholarship from King Mongkut's University of Technology Thonburi.

References

Anggodo, Y., Ariyani, A., Ardi, M., & Mahmudy, W. (2017). Optimization of multi-trip vehicle routing problem with time windows using genetic algorithm. *Journal of Environmental Engineering and Sustainable Technology*, 3(2), 92-97. https://doi.org/10.21776/ub.jeest.2017.003.02.4

Ayadi, R., & Benadada, Y. (2013). Memetic algorithm for a multi-objective vehicle routing problem with multiple trips. *International Journal of Computer Science and Applications*, 10(2), 72-91.

CameraLens Rentals (2022). Drone Rental. Camera Lens RENTALS. Available at: https://www.cameralensrentals.com/rent-drone/rental?sort=default (Accessed: August 2022)

Clarke, G., & Wright, J.W. (1964). Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, 12(4), 568-581. https://doi.org/10.1287/opre.12.4.568

Coelho, B.N., Coelho, V.N., Coelho, I.M., Ochi, L.S., Haghnazar, K.R., Zuidema, D. et al. (2017). A multi-objective green UAV routing problem. *Computers Operations Research,* 88, 306-315. https://doi.org/10.1016/j.cor.2017.04.011

Dantzig, G.B., & Ramser, J.H. (1959). The truck dispatching problem. *Management Science,* 6(1), 80-91. https://doi.org/10.1287/mnsc.6.1.80

Dorling, K., Heinrichs, J., Messier, G.G., & Magierowski, S. (2017). Vehicle routing problems for drone delivery. *IEEE Transactions on Systems, Man, and Cybernetics: Systems,* 47(1), 70-85. https://doi.org/10.1109/TSMC.2016.2582745

Eroglu, D., Gencosman, B.C., Cavdur, F., & Ozmutlu, H.C. (2014). Introducing the MCHF/OVRP/SDMP: Multicapacitated/heterogeneous fleet/open vehicle routing problems with split deliveries and multiproducts. *The Scientific World Journal,* ID 515402. https://doi.org/10.1155/2014/515402

Euchi, J., & Sadok, A. (2021). Hybrid genetic-sweep algorithm to solve the vehicle routing problem with drones. *Physical Communication*, 44. https://doi.org/10.1016/j.phycom.2020.101236

Goldberg, D.E., & Lingle, R. (1985). Alleles, loci and the traveling salesman problem. Proceedings of the *1st International Conference on Genetic Algorithms* (154-159). Los Angeles.

Google (2022). Google Maps from Nong Khai Province, Thailand. Available at: https://goo.gl/maps/6ENu8aVvamrbWc8e6 (Accessed: August 2022)

Holland, J.H. (1975). *Adaptation in natural and artificial systems.* University of Michigan Press.

International Business Machines Corporation (2019). Ibm ilog cplex optimization studio (Version 12.10.0.0). Available at: https://www.ibm.com/products/ilog-cplex-optimization-studio (Accessed: August 2022)

Khoukhi, S., El Yaakoubi, O., Bojji, C., & Bensouda, Y. (2019). A genetic algorithm for solving a multi-trip vehicle routing problem with time windows and simultaneous pick-up and delivery in a hospital complex. Proceedings of the *3rd International Conference on Machine Learning and Soft Computing* (76-80). Da Lat, Viet Nam. https://doi.org/10.1145/3310986.3311031

Kumar, N., Karambir, & Kumar, R. (2012). A comparative analysis of PMX, CX and OX crossover operators for solving travelling salesman problem. *International Journal of Latest Research in Science and Technology,* 1(2).

Kurnia, H., Wahyuni, E.G., Pembrani, E.C., Gardini, S.T., & Aditya, S.K. (2018). Vehicle routing problem using genetic algorithm with multi compartment on vegetable distribution. *IOP Conference Series: Materials Science and Engineering, Yogyakarta,* 325. https://doi.org/10.1088/1757-899x/325/1/012012

Laporte, G. (1992). The vehicle routing problem: An overview of exact and approximate algorithms. *European Journal of Operational Research,* 59(3), 345-358. https://doi.org/10.1016/0377-2217(92)90192-c

Liu, S., Huang, W., & Ma, H. (2009). An effective genetic algorithm for the fleet size and mix vehicle routing problems. *Transportation Research Part E: Logistics and Transportation Review,* 45(3), 434-445. https://doi.org/10.1016/j.tre.2008.10.003

May, A.T., Jariyavajee, C., & Polvichai, J. (2021). An improved genetic algorithm for vehicle routing problem with hard time windows. *2021 IntMolta molta ernational Conference on Electrical, Computer and Energy Technologies (ICECET)* (1-6). Cape Town, South Africa. https://doi.org/10.1109/ICECET52533.2021.9698698

Miller, C.E., Tucker, A.W., & Zemlin, R.A. (1960). Integer programming formulation of traveling salesman problems. *Journal of the ACM,* 7(4), 326-329. https://doi.org/10.1145/321043.321046

Mutingi, M., & Mbohwa, C. (2013). A group genetic algorithm for the fleet size and mix vehicle routing problem. *IEEE International Conference on Industrial Engineering and Engineering.* Hong Kong.

Phalapanyakoon, K., & Siripongwutikorn, P. (2021). Route planning of unmanned aerial vehicles under recharging and mission time constraints. *International Journal of Mathematical, Engineering and Management Science,* 6(5), 1439-1459. https://doi.org/10.33889/IJMEMS.2021.6.5.087

Press Information Bureau of Delhi (2021). First time in South Asia, a "Make in India" drone used to transport COVID-19 vaccines. Available at: https://pib.gov.in/PressReleasePage.aspx?PRID=1760806 (Accessed: August 2022)

Sbai, I., Krichen, S., & Limam, O. (2020). Two meta-heuristics for solving the capacitated vehicle routing problem: the case of the Tunisian Post Office. *Operational Research,* 22(1), 507-549. https://doi.org/10.1007/s12351-019-00543-8

Setiawan, F., Masruroh, N., & Pramuditha, Z. (2019). On modelling and solv- ing heterogeneous vehicle routing problem with multi-trips and multi-products. *Jurnal Teknik Industri,* 21, 91-104. https://doi.org/10.9744/jti.21.2.91-104

Troudi, A., Addouche, S.-A., Dellagi, S., & Mhamedi, A. E. (2018). Sizing of the drone delivery fleet considering energy autonomy. *Sustainability,* 10(9). https://doi.org/10.3390/su10093344

Yuan, X., Zhu, J., Li, Y., Huang, H., & Wu, M. (2021). An enhanced genetic algorithm for unmanned aerial vehicle logistics scheduling. *IET Communications,* 15(10), 1402-1411. https://doi.org/10.1049/cmu2.12106

Zhen, L., Ma, C., Wang, K., Xiao, L., & Zhang, W. (2020). Multi-depot multi- trip vehicle routing problem with time windows and release dates. *Transportation Research Part E: Logistics and Transportation Review,* 135. https://doi.org/10.1016/j.tre.2020.101866