International Journal of
Operations Research

International Journal of Operations Research Vol. 17, No. 4, 101-116 (2020)

Optimal Collaborative Path Planning for Unmanned Surface Vehicles Carried by a Parent Boat Along a Planned Route

Ari Carisza Graha Prasetia[1], I-Lin Wang[2*], Aldy Gunawan[3]

[1,2]Department of Industrial and Information Management
National Cheng Kung University, Tainan, Taiwan

[3]School of Information Systems
Singapore Management University, Singapore

Received July 2020; Revised September 2020; Accepted October 2020

Abstract: In this paper, an effective mechanism using a fleet of unmanned surface vehicles (USVs) carried by a parent boat (PB) is proposed to complete search or scientific tasks over multiple target water areas within a shorter time . Specifically, multiple USVs can be launched from the PB to conduct such operations simultaneously, and each USV can return to the PB for battery recharging or swapping and data collection in order to continue missions in a more extended range. The PB itself follows a planned route with a flexible schedule taking into consideration locational constraints or collision avoidance in a real-world situation. Assuming that each target has a value, this research investigates how to route these USVs, including their schedules to rendezvous with the PB, so that they can maximize the total collected target values from the operation in a limited amount of time. We use a multi-layered time-space network to describe the USVs and PB movement over time and give an integer programming (IP) formulation for the coverage path planning problem. To further shorten the computational time, we propose the Iterative Clustering Heuristic (ICH) to firstly cluster the workspace, calculate the path for each USV to visit targets and meet with the PB for range extension. To evaluate the performance of the proposed IP model and ICH, test cases are designed based on a real-world scenario, as well as families of simulated grid-like networks. Based on the computational analysis of different USV area sizes, targets covered, and operation time-bound increments, the proposed heuristic ICH can solve larger sized cases faster than the IP commercial solver with higher quality results.

Keyword — Coverage Path Planning, Integer Programming, Time-Space Network, Unmanned Surface Vehicle

1. INTRODUCTION

Since unmanned surface vehicles (USVs) are now heavily relied upon for surveillance, intelligence, search, and rescue (Campbell, Naeem, & Irwin, 2012), the difficulties related to finding effective routing for their operations can be significantly affected by the chaotic nature of the wake region downstream of the ship stern as well as the sea conditions during the time of the incident. Several aspects have to be considered, such as the number of units needed for the operation and the coordinates of the targets. This operation is conducted in an environment where different types of services to be performed at some target areas with known coordinates by different types of vehicles. Although USVs are fast and mobile, their range of movement is limited due to smaller fuel or battery bank. Thus a larger ship, we call as a parent boat (PB), is needed to serve as a mobile refueling or recharging station for launching and receiving USVs. This operation can be described as a joint operation problem of a PB with multiple USVs or as a two-echelon problem.

The created model is intended to represent the corresponding problem and illustrate how differences in the number of USVs between each iteration and network structure can affect the time required to complete a task. In addition, we hope to find sufficient routing to perform the operations while considering USV battery capacity in a case of a PB path that is predetermined where targets can be covered in a limited amount of operating time. Understanding the time needed and the resources are very important for the decision-maker. Since the expected computational results will consume a lot of time, the numerical evaluations of the optimization model will be used to illustrate different cases in simulated networks and networks based on real data. These cases are then used to analyze how parameter changes can affect the performance of the proposed IP mathematical formulations and the heuristic.

[*]Corresponding author's e-mail: ilinwang@mail.ncku.edu.tw

http://doi.org/10.6886/IJOR.202012_17(4).0001

To perform the search task for each USV with consideration on the energy consumption, efficient path planning is needed. Traditional path planning algorithms, such as those based on the Traveling Salesman Problem (TSP), are not ideal for this kind of problem since such formulations only consider the distances traveled but ignore the energy or range constraint due to the limited fuel or battery capacity. Therefore, this research mainly analyzes other energy-based optimization methods utilizing Energy Efficient Coverage Path Planning, another formulation that considers the energy consumption characteristics of drones in path planning optimization. We not only consider the energy consumed when traveling between consecutive waypoints (similar to the TSP), but we also consider the energy consumed by the USV when it has to return to the recharging facility (e.g., a PB). The mathematical model is structured based on an Integer Linear Programming (IP) model. A linear energy consumption model allows for linear energy consumption constraints, which are required to derive a compatible result with some optimization solvers such as GUROBI. While several approaches are available for solving this problem, none appear to consider energy consumption as a function in their programming. Routes found using the above approaches may be infeasible for operation scenarios with batteries that are larger than necessary or too heavy to carry. Therefore, this research may complement research such as energy analysis-based research intended to find multiple USV capabilities related to conducting path planning combining the concept of a two-echelon routing orienteering problem with the concept of coverage path planning. Another point is that by analyzing the PB path in specific cases, this research can be used to evaluate a PB path decision that has been made beforehand and provide better decision making for the user in securing as many targets as possible in a limited time.

For the second perspective considering the number of PBs and USVs, most of the related research in a joint operation considers only cases of one-one operation. On the other hand, this research, which focuses on multiple USVs, may provide a better insight into managing multiple USVs and the computational time needed to do so, which is a weakness in research on joint operation path planning. The network in our model is constructed as follows: The PB follows a predetermined path, although the exact movement and the time of the movement are not yet known. USVs need to meet with the PB to have their batteries recharged. The rest of this paper is organized as follows: in Section 2, we provide some results of previous studies and related theories that support our research. The problem definitions and formulations, including the initial mathematical models, are explained in Section 3. Section 4 explains the heuristic developed for this research. Section 5 explains the computational experiments for the simulated grid-like networks and a real-world network, with a summary of the computational analyses. Section 6 provides the conclusions and suggestions for future research.

2. LITERATURE REVIEW

Among the literature on two-echelon location-routing problems (2E-LRPs) and its variants, two echelons interact through an intermediate set of tasks, wherein our case will be the recharging mechanism for the USV. The first echelon (the PB), consists of a primary and intermediate set of tasks related to assisting the USVs, and the second echelon (the USVs) consists of the intermediate battery consumption management and the targets rescued. The main question then is how to synchronize the flows of the two echelons at the intermediate facilities. Synchronization is important due to the limited operating time available for these vehicles. Two-echelon joint operation of unmanned vehicles is defined as having an operation of multiple unmanned vehicles supported by the main vehicle. Similar studies that have been done previously include a joint operation discussed by Garone, Naldi, Casavola, and Frazzoli (2010) about a class of carrier vehicles in which a slow carrier with an infinite operating range cooperates with a faster carrier vehicle that has only a limited range in which to operate.

Another similar study to the present study was conducted by Mirhedayatian, Crainic, Guajardo, and Wallace (2019). They address a new variant of the 2E-LRP that considers synchronization of transshipment and a sequence of delivery and pickup activities by using a primary vehicle and a secondary vehicle. Based on their research, larger instances of data may take a long time to solve since the synchronization problem is a difficult optimization problem. This is because it combines two NP-hard problems, the facility location problem and the vehicle routing problem, especially in the case of a larger instances problem. The results obtained with commercial solvers show that only very small instances are solved optimally, and larger instances cannot be solved with a feasible solution. Another example of joint operation research can be observed from the truck-drone synchronization discussed in Ferrandez, Harbison, Weber, Sturges, and Rich (2016). They added that it is easier to contrast the total time, cost, and energy involved in a hub configuration (star-distance) with truck-only delivery using a TSP route. However, refuel constraints were not considered in this research, which proves to be a good example, since refueling issues can make a joint operation more complex and may potentially be encountered with any vehicle, especially in an uncertain environment.

From the orienteering problem perspective, Labadie, Mansini, Melechovský, and Wolfler Calvo (2012) stated that the Team Orienteering Problem (TOP) is a known NP-hard problem that typically arises in vehicle routing and production scheduling contexts. This review is our consideration when attempting to incorporate a vehicle routing problem into the orienteering problem for a joint operation since this topic can be considered new in this field. El-Hajj and Dang (2016) presented ideas suggesting that further development of a Branch-and-Cut-and-Price type of algorithm is a promising direction towards improving the TOP solution method. In terms of orienteering problems, Mukhina, Visheratin, and Nasonov (2019) state that the majority of problems related to orienteering can be defined using three core components: targets, constraints, and scoring functions. The fourth core component, the algorithm, utilizes other components to generate the best solution for a specific problem. Based on this knowledge, we organize this research based on this structure to develop the optimal solution to the proposed problem considering the fuel consumption for each USV and the synchronization to the PB. In the present research, each target will be clustered based on its location. This was done previously by Yahiaoui, Moukrim, and Serairi (2019) to visit a set of targets with minimum travel time. These aspects are important because, due to the nature of the time-space network in this problem, the computational time of the proposed IP model is expected to be lengthy.

Derived from coverage planning, the Coverage Path Planning is the process of defining a path that passes over given points of an area of interest while avoiding obstacles (Galceran & Carreras, 2013). Based on this definition, our research can be categorized as coverage path planning for a given area of interest. One of the previous investigations included one by Zhang, Zhang, Liu, and Li (2019), who presented a strategy for an intelligent search performed by a USV. Before the start of the operation, global path planning was carried out based on prior information, such as the initial position of the USV, the predicted position of the target, and the range of the search area. An example of this prior information is a grid method used to plot the selected targets in a given area. Given an already decomposed area, each USV is going to be deployed to search for the optimal routing path. Some examples of these optimization techniques were performed by Avellar, Pereira, Pimenta, and Iscold (2015) on minimum time coverage of ground areas using multiple drones and Cruz-Chávez, Rodríguez-León, Rivera-Lopez, and Cruz-Rosales (2019) on vehicle planning with time windows. Both of these studies used an optimal coverage method as area decomposition technique and the Vehicle Routing Problem (VRP) to optimize the problem based on their area decomposition. Based on this research, VRP programming may be used as a feasible option for a multi-vehicle related problem, especially for coverage path planning for multiple vehicles, as in this research.

Based on Zhang et al. (2015), the use of a time-space network is very effective for multi-vehicle problems. A time-space network is also a standard method used for slot allocation problems that consider spatial and temporal requirements. The time-space network shows the position of an individual in time and space. Therefore, our approach will adapt it to the grid-based method to generate the optimal result. After decomposition, then the optimization step can be started.

3. MODEL DESCRIPTION

Our primary objective is to plan a route for a joint operation to service (or cover) the targets during a given period of time. To do this, we need to plan a grid network area for the USVs to operate in. Based on these generated grids, a feasible path to achieving the optimal result can be generated for the PB, and this path will be used as the workspace for the operational movement of the PB. To define a route for the USV, we need to divide the free workspace area into several waypoints. The mechanism for the routing of each vehicle is constructed using a time-space network. In this problem, there are two types of workspaces. The first is the USV workspace, which is exclusively for the USV, and the second is the PB workspace space, which is accessible to both the PB and the USV. Under actual conditions, the USV may roam freely in its workspace. To make a USV route, it is necessary to decompose the USV area into sub-regions. We use a grid-based method involving a decomposition technique that divides the airspace into a virtual uniform grid cell. Suppose that we have already done the preprocessing step and obtained the grid map for the PB area and USV area. We can transform this grid map into a network by changing the virtual cells into nodes and the connections between each cell into an arc. As for the PB workspace, the nodes represent the potential places for the PB to launch the USV, and the arcs represent the route. We then combine these two workspaces. Figure 1 shows the network for the combined operation of the PB and USVs.

Notice that in the network shown in Figure 1, there are arcs connecting the two workspaces. From a USV routing perspective, these arcs turn two distinct workspaces into one workspace. However, from a PB routing perspective, it only accesses the PB workspace but not the USV workspace. An important point here is that the PB is required to follow the predetermined path when the exact movement is still unknown. There are multiple reasons for this mechanism. A PB path is not as flexible as a USV path due to the size of PB, the locations of the harbors, and the need to avoid collisions with the other ship. This network then has similarity in terms of properties with the two-echelon routing problem network. We can use this concept to design our mathematical models. In order to record both the time and battery power consumption of the USVs, together with the PB movement at each time period, we use a time-space network. Figure 2 shows the results of the transformation from the original network into the time-space network.

The time-space network in our model limits the possibility of subtours, and provides detailed USV routing and energy status at any time and place, as suggested by Zhang et al. (2015). In summary, we attempt to model a combined PB and USV operation to cover specifically designated target areas. To build the proposed model, we use several

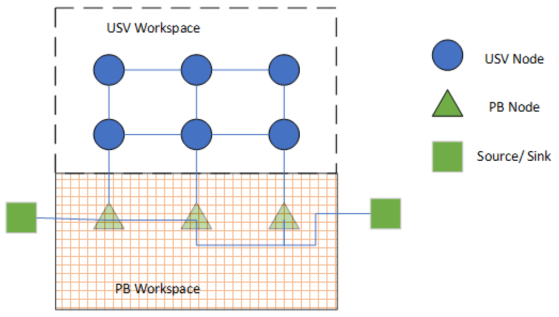


Figure 1: The schematic network for combined PB and USV operations

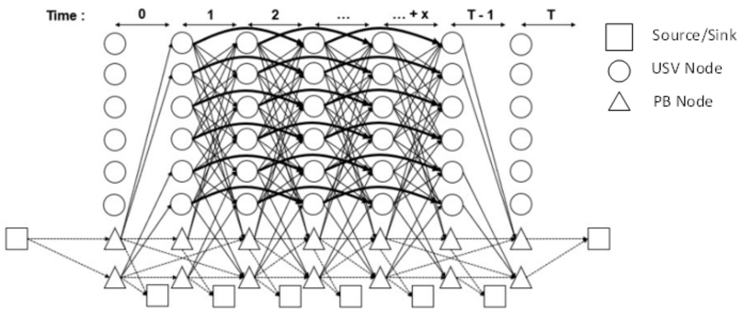


Figure 2: Schematic diagram of the proposed model

methods. To model how the USV performs its task, we use a coverage path planning method. We adopt the two-echelon routing problem concept since we will determine the route for the PB (first echelon) and the USV (second echelon). By combining the coverage path planning and the two-echelon routing problem, the model for the combined PB and USV operation can be developed. We also use a time-space network to track the movement of the USV at each time point, including its energy consumption and the movement of the PB.

An example of a case with one PB and multiple USVs is as follows: Suppose that we have a given area that is already decomposed into a 4x4 grid size. Based on this workspace, suppose that 3 sets of nodes: the PB path nodes, the USV nodes, and the target nodes, are given. As shown in Figure 3, with node 1, 2, 3, and 4 as nodes on the PB predetermined path, all the other nodes considered as USV area nodes, and nodes 11, 5, 6, and 7 are considered as the targets. Each target has its own service time (s_i) representing the required time for completing a task there, and a collected value (D_i) after the task has been completed. The use of a grid network simplifies the USV movement. In particular, we assume a USV can only access from a node to its neighboring node at one time. The region ranges for node 1 comprises cells 11, 2, and 5, and the area range for node 9 is all cells adjacent to it (cells 8, 7, and 10). These ranges determine the USV movement. For example, if a vehicle is located in node 1, the USV will only be able to scan cells 11, 2, and 5. To simplify this problem, we assume to use only one USV in this example. Based on its battery capacity and energy consumption, there are specific options that the USV will have to consider doing the operations, such as staying in the same node or moving back to the PB, in order to reach a target. Information for the traversal time (t_{ij}) of all arcs are known beforehand. Here in our illustrative example, we assume that all the service time on a target s_i and the travel time t_{ij} on an edge (i, j) consume 1 unit of time.

Suppose the PB, together with the USV, starts from node 1. The USV has its maximum battery capacity and a limited time to operate, and its operating time equals 10. We seek the route that gives the optimal collected target values within 10 units of time. We can try to solve the problem manually. For the first iteration, suppose we make the USV move with the PB from the source to node 1 and then launch the USV in node 1. During the 10-unit planning horizon based on the manual calculation, going to nodes 5, 6, and 7 can give us a higher total objective value than visiting target 11, as shown in Figure 5. This is the nature of the orienteering problem in our research, where we want to obtain the highest total collected values as possible in a limited amount of time. Therefore, target 11 could not be retrieved by t=10, but when we extend the planning horizon to be t=18, the optimal route will change, as shown in Figure 6.

In particular, both the USV and PB start from node 1. Each USV is able to reach targets 5, 6, and 7, stay in each target for 1 unit of time, and then return to the PB path (node 2) to continue the operation before t=10. In this routing, there are several possibilities that each USV can take to reach the target. For example, the USVs can take off at node 2 and reach targets 5, 6, and 7 in this order, but then the total operating time will become 11. On the other hand, once

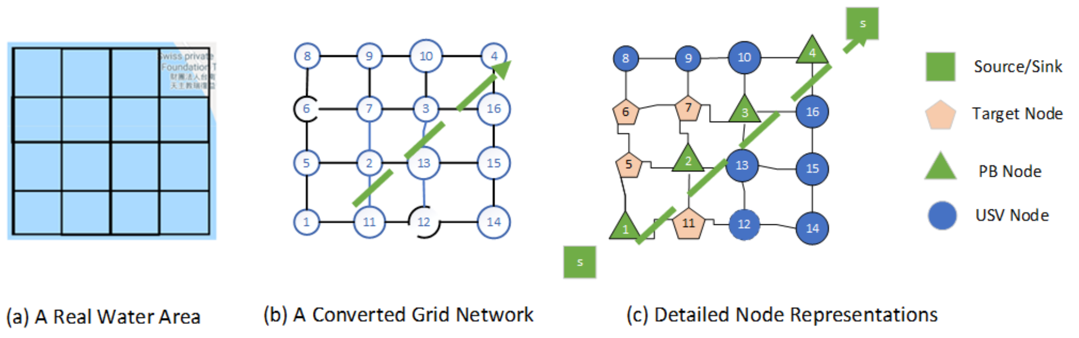

Source/Sink
Target Node
PB Node
USV Node
(a) A Real Water Area
(b) A Converted Grid Network
(c) Detailed Node Representations

Figure 3: Area Information: illustration

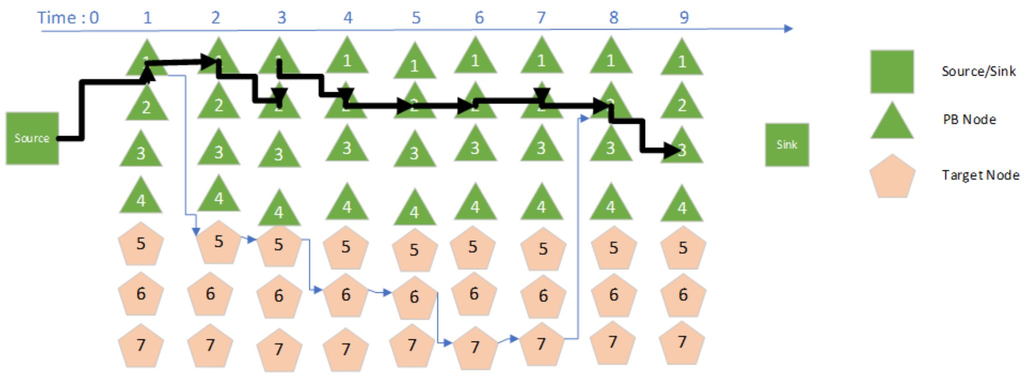


Figure 4: A time-space network structure

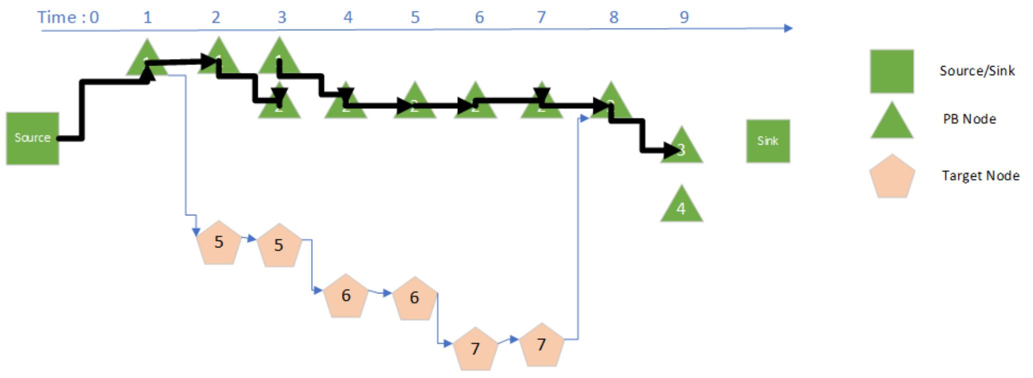


Figure 5: Model results in 10 units of time

the operation time-bound is set to be 18, after the USV and the PB meet in node 2 at $t = 8$, the USV can be launched to visit target 11 and collect higher values in total.

Since both USVs and the PB are starting from node 2, the total maximum time needed for every variable to return to the sink node now requires 18 total units of time. This gave us multiple possibilities of how the modeling can be used to find the optimal time needed for operations as well as to generate the optimum time required if the time is limited based on the importance of each target. Facing this situation, we can use the proposed mathematical programming-based approach. After dividing the size of the grid and the number of nodes in the system, we create the time-space network based on the area of space that we intend to analyze.

Let $G = (V, A)$ be an undirected graph, where V is the set of all nodes, including one main depot (node $i = 1$) and one destination (node $i = n$). Let V_p be a subset of V representing the rendezvous nodes in the PB path workspace, and V_u is a subset of V representing the cells in the USV workspace, which is acquired from the preprocessing step. Each rendezvous node in the PB space represents a location where the PB can stop to launch a USV.

A set of targets V_T, residing at $|V_T|$ different locations, where each target $i \in V_T$ is associated with an importance value (Di). The PB workspace and the USV workspace may overlap each other. Let $R_i = \{j | d_{ij} \leq \gamma, j \in V\}$ be a set of nodes adjacent (i.e., accessible in 1 unit of time) from node $i \in V$. Let A be the set of all edges, whose subset

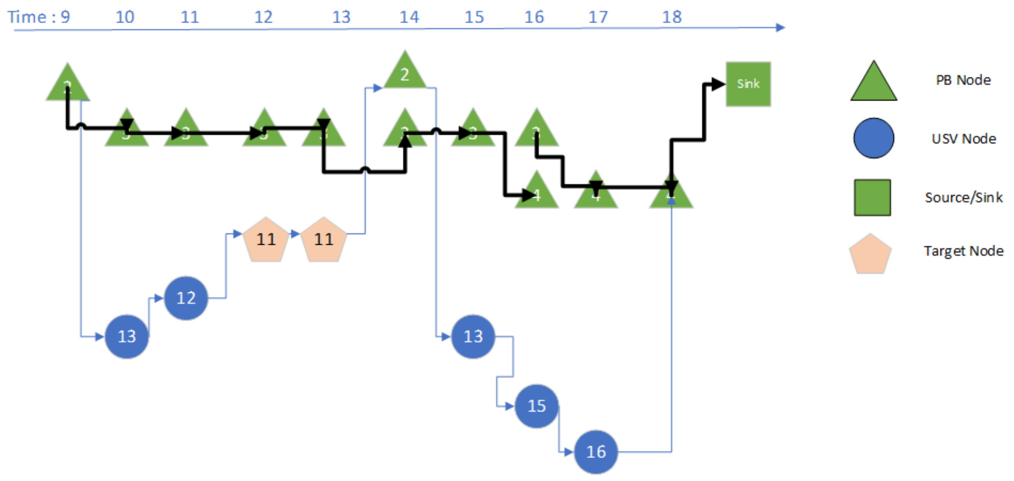


Figure 6: Model results in 18 units of time

$A_p = \left\{(i', j')|i', j' \in V_p\right\}$ is a set of all edges located in the PB path, subset $A_u = (i", j")┤|i", j"\square V_u$ is a set of all edges located in the USV workspace, subset $A_c = \left\{(i', j")|j" \in R_{i'}\right\} \cup \left\{(i", j')|j' \in R_{i"}\right\}$ is a set of all edges connecting the PB nodes with the USV workspace, and subset $A_s = \{(i", i")|i" \in V_u\} \cup \left\{(i', i')|i' \in V_p\right\}$ is a set of staying edges (self-loop) that indicate that the USV stays at a node $i" \in V_u$ to perform the service or to wait at a node $i' \in V_p$ when necessary. The PB will only move along edges in A_p, while the USV can move along all other edges. We introduce the following assumption for our mathematical programming model:

1. The service times and values for all target cells (nodes) and traversal time for all edges are known constants.
2. The targets are evenly distributed in the USV workplace.
3. The PB can only traverse edges in its network, and the PB path is known beforehand.
4. When the USVs return to the PB, each USV can charge its battery and start the next routing immediately. The time for charging the battery is short and negligible.

We summarize the mathematical notations, including the parameters, variables, and sets in Table 1. We aim to route k USVs to visit some targets, with support from a PB, to maximize the collected target values within a limited time T. Equation (1) defines the objective function, where w_i is the decision variable determining whether the target i is serviced (i.e., covered).

$$\max \sum_{i \in V_T} D_i w_i \tag{1}$$

The flow balance constraint (2) defines the movement for PB on the PB path.

$$\sum_{(i,j) \in A_p \cup A_{sink} \cup A_s} y_{ijt} - \sum_{(j,i) \in A_p \cup A_{source} \cup A_s} y_{ji(t-\Delta_{ji})} = 0 \quad \forall i \in V_p, t \in T \tag{2}$$

The flow balance constraint (3) defines the movement for each USV k. Note that a USV can appear in any node in V.

$$\sum_{(i,j) \in A_p \cup A_u \cup A_s \cup A_c} x^k_{ijt} - \sum_{(j,i) \in A_p \cup A_u \cup A_s \cup A_c} x^k_{ji(t-\Delta_{ji})} = 0 \quad \forall i \in V, t \in T, k \in K \tag{3}$$

Constraint (4) guarantees a covered target $i \in V_T$ requires some USV k to stay there for at least s_i units of time.

$$\sum_{t \in T} x^k_{iiy} \geq s_i w_i \quad \forall i \in V_T, k \in K \tag{4}$$

Table 1: Notations for the mathematical formulations

Parameters	
Δ_{ij}	Travel time for passing edge $(i,j), \forall (i,j) \in A$
s_i	Service time for visit node $i, \forall i \in V_T$
D_i	The value obtained by covering a target node i, $\forall i \in V_T$
b_{ij}	Battery consumptions for passing the edge (i,j) , $\forall (i,j) \in A$
B	Maximum battery capacity
Decision Variables	
x_{ijt}^k	1, if USV k at a time t start to passing edge $(i,j), \forall (i,j) \in A$; and 0, otherwise
y_{ijt}	1, if PB at a time t start to passing edge $(i,j), \forall (i,j) \in A_p$; and 0, otherwise
z_{it}^k	1, if the PB and USV k at a time t both are at node i, and 0, otherwise
M_{ijt}^k	1, if the PB and USV k both move along edge (i,j) in $A_p \cup A_s$, and 0, otherwise
R_{ijt}^k	1, if the USV k needs to be recharged along edge (i,j) in $A_p \cup A_c \cup A_s$, and 0, otherwise
w_i	1, if target i is covered, $\forall i \in V_T$, and, 0 otherwisear
α_{ijt}^k	the initial battery level for USV k to move along edge (i,j) starting at time t
β_{ijt}^k	the ending battery level for USV k to move along edge (i,j) starting at time t
Sets	
V	Set of all USV and PB workplace nodes, $V = V_u \cup V_p$
V_p	Set of PB path nodes
V_u	Set of USV workspace nodes
A	Set of all workplace edges, $A = A_p \cup A_u \cup A_c \cup A_s$
A_p	Set of edges for the PB path, $A_p = \left\{(i^{'},j^{'}) \mid i^{'},j^{'} \in V_p\right\}$
A_u	Set of edges for the USV workspace, $A_u = \left\{(i^{"},j^{"}) \mid i^{"},j^{"} \in V_u\right\}$
A_c	Set of connecting edges between PB and USV nodes, $A_c = \left\{(i^{'},j^{"}) \mid j^{"} \in R_{i'}\right\} \cup \left\{(i^{"},j^{'}) \mid j^{'} \in R_{i^{"}}\right\}$
A_s	Set of staying edges (self-loop), $A_s = \left\{(i^{"},i^{"}) \mid i^{"} \in V_u\right\} \cup \left\{(i^{'},i^{'}) \mid i^{'} \in V_p\right\}$
A_{sink}	Set of ending edges connecting to the sink node ψ , $A_{sink} = \{(i,\psi) \mid i \in V\}$
N_i	Set of nodes adjacent to the node i within range γ, $N_i = \{j \mid d_{ij} \leq \gamma, j \in V\}$
T	Set of time periods

Constraints (5)-(6) are used to determine whether a recharge may take place (i.e., $z_{it}^k = 1$) or not, when both a USV and PB move together.

$$Z_{it}^k \geq \sum_{(i,j) \in A_p \cup A_s \cup A_c} x_{ijt}^k + \sum_{(i,j) \in A_p \cup A_s} y_{ijt} - 1 \qquad \forall t \in T, i \in V_p, k \in K \tag{5}$$

$$Z_{it}^k \leq \left(\sum_{(i,j) \in A_p \cup A_s \cup A_c} x_{ijt}^k + \sum_{(i,j) \in A_p \cup A_s} y_{ijt} \right) \qquad \forall t \in T, i \in V_p, k \in K \tag{6}$$

Constraints (7)-(8) ensure that to trigger $M_{ijt}^k = 1$, when both a USV and PB move along the same arc (i.e., $x_{ijt}^k = y_{ijt} = 1$). M_{ijt}^k will be used in some battery consumption equations.

$$M_{ijt}^k \geq x_{ijt}^k + y_{ijt} - 1 \qquad \forall t \in T, (i,j) \in A_p \cup A_s, k \in K \tag{7}$$

$$M_{ijt}^k \leq \left(x_{ijt}^k + y_{ijt}\right)/2 \qquad \forall t \in T, (i,j) \in A_p \cup A_s, k \in K \tag{8}$$

Constraints (9)-(10) are used to set $R_{ijt}^k = x_{ijt}^k z_{it}^k$, which equals to 1 when USV k and PB meet (i.e., $z_{it}^k = 1$) and moves along edge (i,j) (i.e., $x_{ijt}^k = 1$). This constraint is needed for the battery recharge when USV is on the PB path.

$$R_{ijt}^k \geq x_{ijt}^k + z_{it}^k - 1 \qquad \forall t \in T, (i,j) \in A_p \cup A_s, k \in K \tag{9}$$

$$R_{ijt}^k \leq (x_{ijt}^k + z_{it}^k)/2 \qquad \forall t \in T, (i,j) \in A_p \cup A_s, k \in K \tag{10}$$

To limit the battery capacity for a USV, constraints (11)-(13) are used to set the starting battery level (α_{ijt}^k) associated with an arc $(i,j) \in A$ to be no more than the maximum capacity (B) at anytime, and α_{ijt}^k becomes full in the beginning (i.e., $x_{ij0}^k = 1$) or while carried by PB (i.e., $R_{ijt}^k = 1$).

$$\alpha_{ijt}^k \leq Bx_{ijt}^k \qquad \forall t \in T, (i,j) \in A, k \in K \tag{11}$$
$$\alpha_{ij0}^k = Bx_{ij0}^k \qquad \forall t \in T, (i,j) \in A_p \cup A_s \cup A_c, k \in K \tag{12}$$
$$\alpha_{ijt}^k \geq BR_{ijt}^k \qquad \forall t \in T, (i,j) \in A_p \cup A_s \cup A_c, k \in K \tag{13}$$

Constraint (14) sets the battery consumption along an arc $(i,j) \in A_u$, and (15) ignores that change when a USV moves with PB (i.e.,$x_{ijt}^k = M_{ijt}^k = 1$).

$$\beta_{ijt}^k = \alpha_{ijt}^k - b_{ij}^k x_{ijt}^k \qquad \forall t \in T, (i,j) \in A_u, k \in K \tag{14}$$
$$\beta_{ijt}^k = \alpha_{ijt}^k - b_{ij}^k (x_{ijt}^k - M_{ijt}^k) \qquad \forall t \in T, (i,j) \in A_p \cup A_s, k \in K \tag{15}$$

For each USV node i passed by USV k, its entering battery level equals its leaving battery level, as shown in constraint (16). On the other hand, if USV k passes a PB node, it may be recharged (i.e.,$R_{ijt}^k = 1$) if necessary by constraint (17).

$$\sum_{(1,j) \in A_u \cup A_c \cup A_s} \alpha_{ijt}^k - \sum_{(j,i) \in A_u \cup A_c \cup A_s} \beta_{ji(t-\Delta_{ji})}^k = 0 \qquad \forall t \in T, i \in V_u, k \in K \tag{16}$$
$$\sum_{(1,j) \in A_u \cup A_c \cup A_s} (\alpha_{ijt}^k - BR_{ijt}^k) - \sum_{(j,i) \in A_p \cup A_c \cup A_s} \beta_{ji(t-\Delta_{ji})}^k \leq 0 \qquad \forall t \in T, i \in V_p, k \in K \tag{17}$$

Both USV k and PB are set to go to the sink node by the end of the planning horizon, meaning the operation is finished, as represented in constraints (20)-(21)

$$\sum_{t \in T} \sum_{(i,\psi) \in A_{sink}} y_{i\psi t} = 1 \tag{18}$$
$$\sum_{t \in T} \sum_{(i,\psi) \in A_{sink}} x_{i\psi t}^k = 1 \quad \forall k \in K \tag{19}$$

The domains for all variables are defined in constraints (20) - (24).

$$x_{ijt}^k \in \{0,1\} \qquad \forall t \in T, (i,j) \in A_u, k \in K \tag{20}$$
$$y_{ijt}, M_{ijt}^k, R_{ijt}^k \in \{0,1\} \qquad \forall t \in T, (i,j) \in A_p, k \in K \tag{21}$$
$$R_{ijt}^k \in \{0,1\} \qquad \forall t \in T, (i,j) \in A_p \cup A_s \cup A_c, k \in K \tag{22}$$
$$z_{it}^k \in \{0,1\} \qquad \forall t \in T, i \in V_p, k \in K \tag{23}$$
$$w_i \in \{0,1\} \qquad \forall t \in T, i \in V_T, k \in K \tag{24}$$

4. THE ITERATIVE CLUSTERING HEURISTIC (ICH)

Since we are dealing with an NP-hard problem, the IP formulation in Section 3 requires long computational times for large instances, even by the state-of-the-art commercial software such as GUROBI. To calculate good solutions within shorter computational times, we propose a heuristic approach for the joint operation model in this section. There are three main steps in our proposed heuristic. The first step is to apply a k-means algorithm to cluster the workspace for each USV. We determine a good sequence for those clusters based on the distance of the geometric center to the depot. Finally, we solve the simplified IP formulation associated with each cluster to determine optimal USV routing for target covering and rendezvousing with PB by GUROBI.

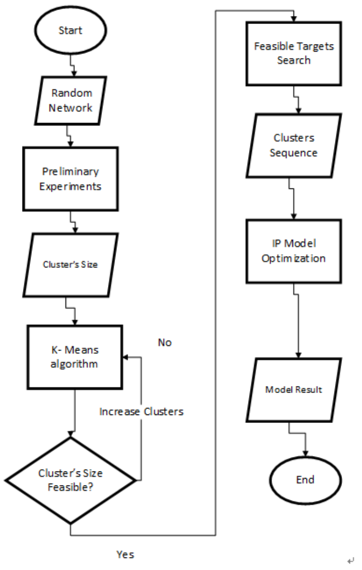


Figure 7: The Iterative Clustering Heuristic flowchart

Figure 7 illustrates the procedures of our proposed heuristic. After the algorithm reads the problem data (i.e., all the information associated with nodes and arcs), we conduct a quick preliminary experiment to determine a good number of clusters (a multiple of k). Then we apply a k-means algorithm to cluster USV nodes so that each cluster forms a connected subnetwork of similar size (i.e., number of USV nodes including targets) and is assigned to a USV. A good cluster visiting sequence is determined by the increasing order of the distance between the geometric cluster center to the depot. For each cluster, we formulate a smaller IP to solve by GUROBI for the USV routes as well as its rendezvous with PB. Finally, we connect PB nodes between clusters, as well as the USV routes inside each cluster, which defines all the USV routes and PB route. The entire procedure contains iterative runs of node clustering and routings for USVs. Thus we name it as an Iterative Clustering Heuristic (ICH).

4.1 Preliminary Experiments

The purpose of our preliminary experiments is to seek good parameter settings for ICH to attain a good compromise between efficiency and effectiveness. The most time-consuming procedure in ICH is to solve routing for each USV cluster by GUROBI. The larger a USV cluster is, the better solution we can get but with longer computational time. Here we simulate sample grid networks, and try out the relation between the computational time and size of solving orienteering problems by GUROBI, to determine an appropriate size (e.g., number of USV nodes) of a USV cluster. We then use the test result to determine the number of clusters (in this paper, a multiple of k) for ICH.

For example, we may simulate a set of 6×10 random networks containing 1×10 PB nodes in the bottom layer and 5×10 USV nodes in upper layers, randomly select some (e.g., 30 among 50) USV nodes as targets where each target has its given service time and value. With a given battery capacity (e.g., $B = 100$) for each USV and the battery consumption on each target and along each edge, we construct an orienteering problem IP formulation to test the best routing for a USV within its range (i.e., before its battery level reaches 0). Based on the tests on simulated networks, we can observe the average number of USV nodes (e.g., 35) to be traversed by a USV without recharging. Suppose there are a total 300 USV nodes, and there are 5 USVs, then we may divide the USV nodes into 10 clusters. This is because among the 5, 10, and 15 clusters, 10 clusters give the best compromise (10*35=350 is closer to 300 than 5*35 and 15*35). Figure 8 illustrates a clustering example.

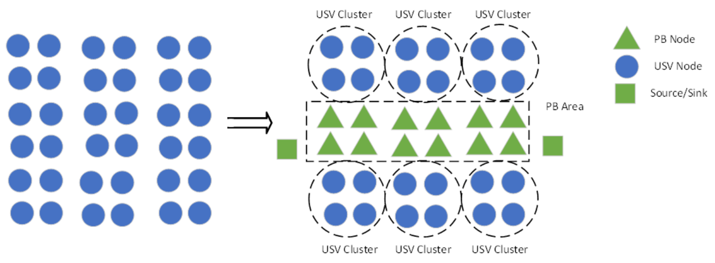


Figure 8: A possible USV clustering

4.2 USV Area Clusters

Our ICH algorithm is a divide-and-conquer heuristic. Based on the result of preliminary experiments, we have calculated a good cluster number (k^*). We then divide the entire workspace into k^* clusters, so that we can evenly assign k USVs for these clusters, and solve the corresponding simplified IP model associated with each cluster for its best USV routing by GUROBI. Since the size of each IP model is much smaller than the original IP model, as shown in Section 3, the entire procedure requires less computational time.

Here we suggest using the K-means algorithm to determine the composition of k^* clusters, since we assume the targets are somewhat fairly distributed on the USV workspace. If the targets are not evenly distributed, one needs to design better clustering methods. After the clusters are formed, we can estimate a good visiting sequence for the PB, by the increasing order of the distance of cluster center to the depot.

4.3 Predicting Feasible Targets

ICH calculates a USV route for each cluster by solving a simplified IP model, yet it might still suffer from the computational burden. To further speed up the procedure, we have applied the GRASP (Greedy Randomized Adaptive Search Procedure) algorithm to calculate the approximation targets that can be reached by the USVs within the time-bound in each k^* clusters. In particular, targets outside of the feasible area will has less priority compared to the one within the range, thus limiting the possible movements for the USVs. It then determines a good target visiting sequence for ICH, which then can be used as a constraint for the USV routing IP model to generate good solution quickly.

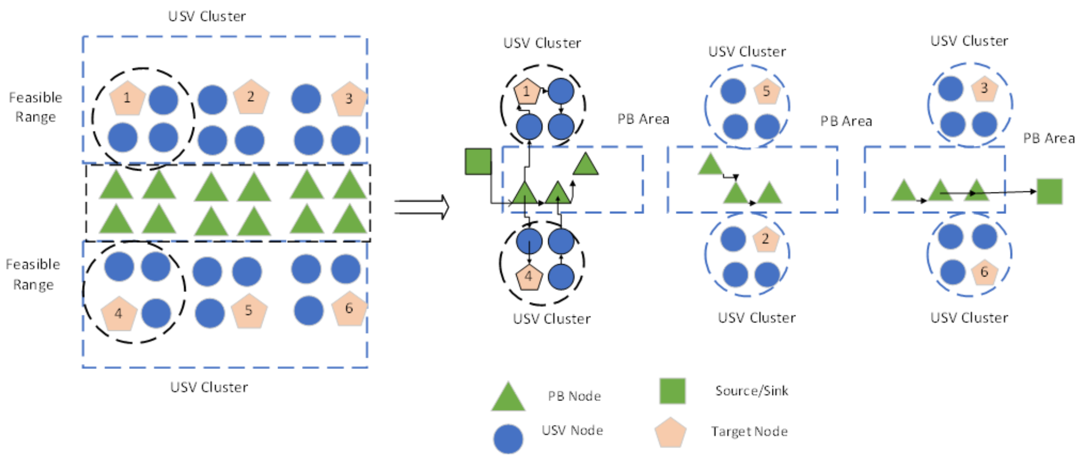


Figure 9: Illustration of predicting Targets

In Figure 9, assuming that we have obtained the clusters, we can find an initial feasible route that will cover the targets and use the target visiting sequence as a constraint for the IP model to limit the movement of the USVs. In feasible range 1, targets 1 and 4 are within the range, unlike targets 5, 3, 2, and 6. Therefore, we can set targets 1 or 4 to be visited first, before 5, 3, 2, and 6.

4.4 Summary

Both the clustering and the targets prediction are used to simplify the model. With the calculated USV cluster, in each iteration, the USVs will use different datasets to solve each cluster individually. To understand this mechanism better, Figure 10 provides an integration of the clusters shown in Figure 9. Assuming that each cluster has an identification number (cluster 1, cluster 2, $\cdots$, etc.), each cluster will function as a self-contained problem with a PB route as well as USV workspace with its targets.

Figure 10 describes the simplified network results from this heuristic, with each USV assigned to a cluster where the workspace for each USV will be limited and thus allows faster computational results. However, assigning the cluster will make each USV's movement less flexible than the original network. Therefore, a good design and the network structure used in the clustering process are both important for this heuristic.

5. EXPERIMENTAL RESULTS

There are two sets of experiments in this Section. Section 5.1 provides the specifications for the computational settings discussed. Section 5.2. describes the simulated experiments that are generated using a random network generator, while the second case is a simulated network based on a real-world location, which is discussed in Section 5.3.

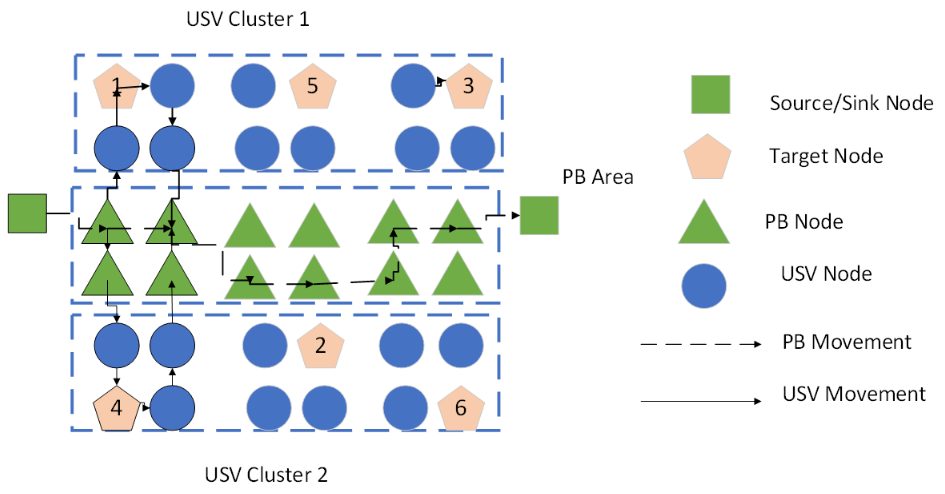


Figure 10: Simplified network results

5.1 Technical Specifications

With the proposed problem, the model is constructed in the GUROBI optimization solver. The computational experiments are conducted on a personal computer (PC). The specifications for the PC and the software used to perform the computational experiments are shown in Table 2.

Table 2: PC Specifications

PC specifications	
Operating System	Microsoft Windows 10 1 x64-based
Processor	Intel ® Core ™ i7-6700HQ CPU @ 2.60 GHz
RAM	16.00 GB
Software specifications	
Python (Cython Compiler)	Version 3.7.6
Anaconda	Version Anaconda 2020.02 (64 Bit)
GUROBI	Version 9.0.3

5.2 Experimental Settings for the Simulation

For the experimental cases, we mapped the area based on the grid network used for USV operations. The grid is used to define the USV workspace used to cover points that are considered to be nodes. The example in Figure 11 shows that a simple network is formed based on the diagonal edges between each node. The network formed follows the grid rules, in that each node is connected to its adjacent neighbor (i.e., vertically or horizontally one-edge away) nodes.

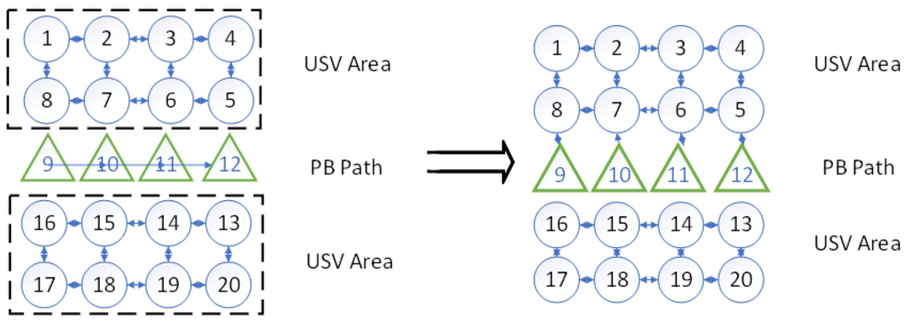


Figure 11: A simulated grid network example

There are different types of edges used in this problem to model the connecting edges that cannot be put into the same category as the USV area edges since they are related to the possibility of recharging the USV in the PB nodes.

Therefore, the three types of edges are as follows: the USV area edges, the connecting edges between the PB area and USV area, and the PB area edges. In this case, the path for the PB is known beforehand. While the exact movement time prior to the PB having to stop is not known, the PB does not have to search for a feasible path. The illustration for this case is shown in Figure 11. The PB path is defined as numbered nodes illustrating a predetermined path from an origin node/area (node 9) to a destination node/area (node 12).

The scenarios are divided into three sections. In the first scenario (S1), we fix the number of targets to be 15 with random locations among the 60, 100, and 200 USV nodes, denoted by problem sets S1_60, S1_100, and S_200, respectively. In the second scenario (S2), we fix the number of USV nodes to be 100, and generate 3 problem sets S2_10, S2_30, S2_50 that contains 10, 30, and 50 randomly located targets. In the third scenario (S3), we fix the number of randomly located targets to be 15 (similar to S1) from 100 USV nodes (similar to S2) with different operating time bounds 30, 50, and 65, to form the problem sets S3_30, S3_50, and S3_65, respectively.

Each of the nine problem sets was run for ten times, which generating 90 random networks for our testing. Then, we solve each random case by our IP model and ICH heuristics to analyze the average performance of 10 random cases for each problem set. There were several parameters used as shown in Table 3, where $|V_u|$ represents the number of USV nodes; $|V_T|$ represents the number of the target nodes; $|A_u|$ is the number of USV edges, $|A_P|$ is the number of PB edges, and $|K|$ is the number of USVs that is set to be 2 in each case. T represents the maximum period for the model. $|V_p|$ represents the total number of PB nodes in which the PB path will be set as a linear path from the origin node to the destination node, with 10 nodes in each row ($|A_p| = 20$). CPU represents the total computational time necessary to obtain the solution, which becomes 3600(s) if an optimal solution cannot be found within a 1-hour time limit. The GAP represents the relative difference in the objective value between the best-known solution and the best-bound value. We set the battery capacity at 100 ($B = 100$) for all cases, and the number of ICH clusters was set at 4. From Scenario 1 (USV nodes increment), we can conclude that due to the clustering technique, the workspace for

Table 3: Experimental results for the simulated networks

							IP Model		Iterative Cluster Heuristic	
Problem Set	$\|A_u\|$	$\|A_p\|$	$\|V_u\|$	$\|V_t\|$	$\|V_p\|$	T	CPU(s)	GAP(%)	CPU(s)	GAP(%)
S1_60	230	20	60	15	10	50	3600.52	52.94	532.97	0.90
S1_100	382	20	100	15	10	50	3600.23	-	1988.43	4.80
S1_200	722	20	200	15	10	50	3600.10	-	2855.12	10.64
Problem Set	$\|A_u\|$	$\|A_p\|$	$\|V_u\|$	$\|V_t\|$	$\|V_p\|$	T	CPU(s)	GAP(%)	CPU(s)	GAP(%)
S2_10	382	20	100	10	10	50	2707.21	0.00	434.67	0.90
S2_30	382	20	100	30	10	50	3600.00	31.75	1145.26	4.06
S2_50	382	20	200	50	10	50	3600.00	-	3600.95	31.45
Problem Set	$\|A_u\|$	$\|A_p\|$	$\|V_u\|$	$\|V_t\|$	$\|V_p\|$	T	CPU(s)	GAP(%)	CPU(s)	GAP(%)
S3_30	382	20	100	15	10	30	2995.51	5.48	264.30	0.48
S3_50	382	20	100	15	10	50	3600.00	46.68	1167.60	1.76
S3_65	382	20	200	15	10	65	3600.00	-	3067.17	10.19

the USV in ICH was simplified and managed to lower the computational time significantly. For example, in the case of S1_100 and S1_200, the IP model was not able to achieve a feasible solution within the 1-hour time limit. In Scenario 2 (Targets increment), as expected, increasing the targets also increased the GAP (%) in the ICH. The problem's difficulty is more sensitive in the number of targets ($|V_t|$) than the number of USV nodes ($|V_u|$). Note that the 31.45% GAP of the ICH performance for solving the problem set S2_50 may be misleading. The actual GAP is better than the current one since here we compare with an inaccurate IP model upper bound. If we extend the time limit (T), we can get a more accurate bound, and then the GAP reported can be reduced. The boxplots on the optimality gaps by ICH for scenarios 1, 2, and 3 are shown in Figure 12, 13, and 14, respectively.

In the case of the S1_60 (60 nodes) the GAP (%) tended to be smaller, and with each USV node added to the network, the upper bound and the median for the cases increased exponentially. This was due to the branch-and-bound nature of the solver, where adding more USV nodes in the same number of clusters allowed more USV nodes in the same cluster. For example, in S1_60, there will be approximately 15 nodes in each USV's cluster, and in S1_200 (200 nodes), there will be approximately 50 nodes in each USV's cluster. Having more nodes in the same cluster requires a longer search process and thus increases computational time exponentially, and the commercial solver is not able to reach optimality as quickly as is the case with fewer USV nodes.

The same phenomenon can be observed in the target increment scenario (Cases S2). When more targets are added to the same network, there will be multiple possibilities for total objectives values for each USV. Since the USVs are required to search for all possible combinations, adding more targets are proven to increase both CPU (s) and the

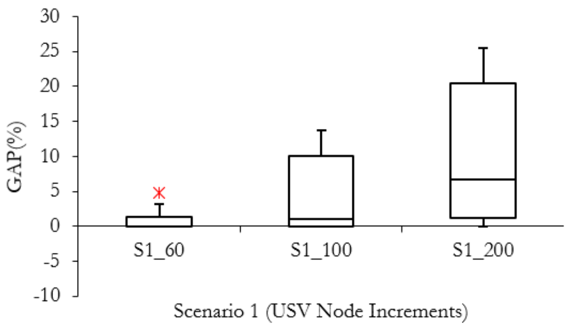


Figure 12: ICH GAP (%) comparisons of the scenario USV nodes (15 targets and 50 units of time)

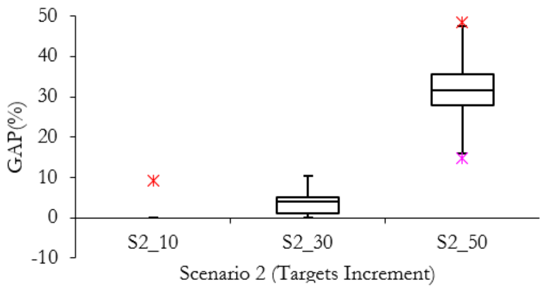


Figure 13: ICH GAP (%) comparison of scenario targets (100 USV nodes and 50 Units of time)

GAP (%). In the S2_10 (10 targets) cases, the ICH performs the best when the GAP (%) could reach 0, where with each additional target added to the same network, the computational time increases exponentially where S2_50 spends more than 1-hour limit in average of each computational run. Thus, the heuristic exhibited an elevated GAP(%).

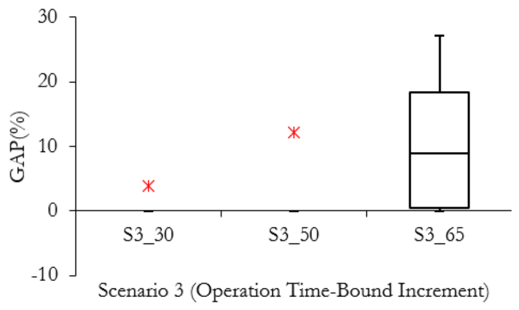


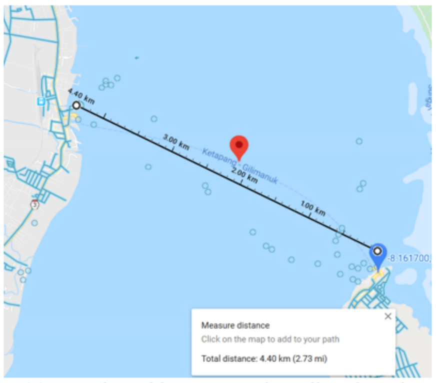
Figure 14: ICH GAP (%) Comparison of Scenario Time-Bound Operation (15 Targets and 100 USV Nodes)

In the time-bound increment cases, the nature of the branch-and-bound of the commercial solver is very important. In the cases of 30 units of time and 50 units of time, the heuristic was able to perform with optimal results. Meanwhile, in the higher time-bound cases, each case and the search process required 65 units of time and exponentially increased the computational time. Based on these simulated experimental results, we can conclude that although the ICH is able to solve the problem faster than the IP model, it still cannot escape the curse of dimensionality. In particular, ICH still suffers from a cluster's size and the length of the planning horizon.

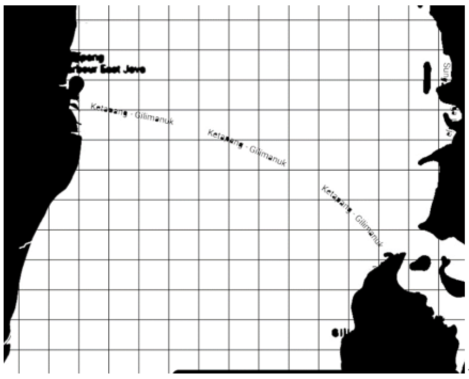
5.3 Real-world Data Experimental Setting

For the construction network of the real data, the grid network as shown in Figure 15 was taken from a real map located in Bali province, Indonesia. Since there are two main harbors connecting Bali (Gilimanuk Harbor) and Java (Ketapang Harbor) that can be used as the harbor for the PB, the route follows the real-world data taken from Google Maps and are computed to the grid network composed by cells with simulated targets.

Based on the real data and the distance measurement, the size of a cell was set to be 400 m × 400 m, with a total operating time of 35 minutes to reflect the 35 minutes of travel time required for transportation from Gilimanuk


4.40 km
3.00 km
Ketapang - Gilimanuk
2.00 km
1.00 km
Measure distance
Click on the map to add to your path
Total distance: 4.40 km (2.73 mi)

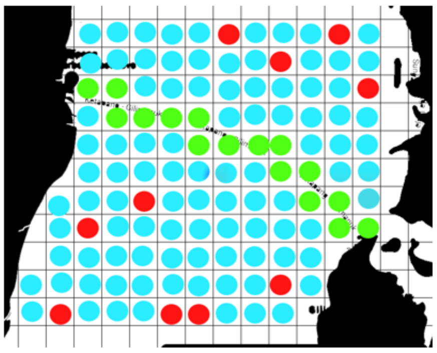
(a) A real-world water area in Bali, Indonesia



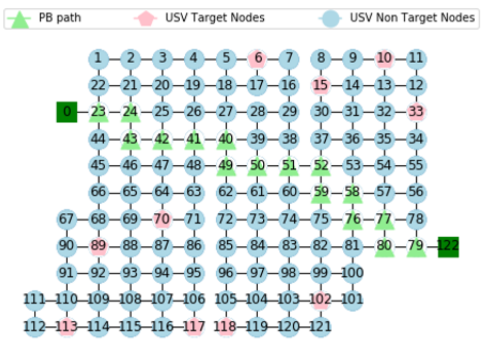
(b) A grid network approximation

Figure 15: Google Map Data Conversion to Grid Network

Harbor to arrive at Ketapang Harbor. There were a total of 10 randomized targets spread along the path, with a total of 96 blue nodes (USV area nodes). The green nodes represent the predetermined path of the PB (16 nodes), and the red nodes represent the targets. The number of vehicles used in the model was set to 2 USVs and a PB.



(a) Representations of USV, PB, target nodes



(b) A converted grid network approximation

Figure 16: Representation of a real-world grid network

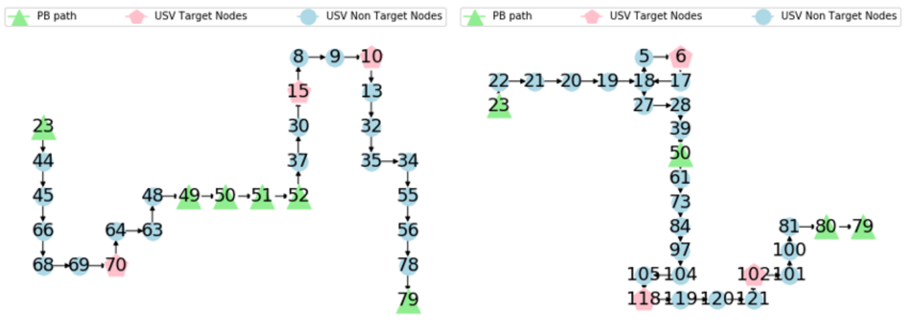
From the network described in Figure 16., both the IP model and the ICH were implemented, with the results obtained is shown in Table 4 below. The computational results obtained from both the IP model and the ICH heuristic are shown in Table 4. Each of the methods was limited to a 3600 (s) maximum computational time to limit the time spent to solve the case.

Table 4: Gilimanuk-Ketapang Experimental Results

Method	GAP (%)	Objective value	# Opt	CPU (s)	Upper Bound
Iterative Clustering Heuristic	4.76	40	6	351.66	42
IP-model Commercial Solver	21.43	33	2	3601.95	42

Based on these results, the ICH method performed the best, in general, based on both the GAP (%) and CPU (%). There were six optimal solution counts from the ICH and two optimal solution counts from the IP commercial solver although the results obtained were not the optimal solution since the upper bound objective was not the same as the highest objective. Applying the ICH to this structure did result in solving the problem much faster compared to the IP model since within the 1-hour time limit, the IP model only found two solutions. In comparison, the ICH was able to find six solutions ten times faster than the IP model. This means that the use of the ICH method as a heuristic with the intention of solving the problem faster with good performance is effective, and in these settings, it is the best result that can be obtained. The details of the routing for each USV are shown in Figure 18.

From the computational results shown in Figure 17, in 35 units of time, 6 numbers of 10 targets are saved. USV 1 retrieved targets 70, 15, and 10 with total objective values of 23, and USV 2 retrieved targets 6, 118, and 102 with total objective values of 17. Recharging occurred four times with drone 1 at time 12 at nodes 49-50 and at time 14 at nodes 50 $\rightarrow$ 51, and recharging occurred with drone 2 at time 14 at nodes 50 $\rightarrow$ 61 and at time 32 at nodes 80 $\rightarrow$ 79.



(a) USV 1 Route (b) USV 2 Route

Figure 17: Representation of a real-world grid network

These computational results prove that the IP mathematical formulations are able to solve the problem and to provide the detailed movement of each USV as well as the PB. The number of times recharging occurs for each USV and the location the PB is required to move with the USVs, as well as which targets can give the highest total objective values for the targets that are retrieved.

6. CONCLUSION

This paper focuses on the joint operation of a parent boat (PB) in a predetermined path and unmanned surface vehicles (USVs) with a limited battery capacity. This limitation can be solved by treating a PB as a mobile recharging station. This research aims to maximize the total value of targets rescued within a limited operating period and to calculate the optimal routes that satisfy the USV energy constraints during the process. To solve this problem, we proposed a combined concept, including two-echelon routing, an orienteering problem, and coverage path planning with energy considerations. We formulated IP mathematical formulations that can solve the problem with detailed results. However, the commercial solver for solving the IP mathematical formulations requires a lot of time. To solve the problem faster, we proposed the Iterative Clustering Heuristic (ICH) to limit the movements of the USVs by assigning each USV to a cluster. The solver was only applied to solve the simplified problem. There were two experiments conducted on this problem, including simulated experiments with three scenarios, to understand the parameter changes in the IP model and the ICH, and an analysis was conducted based on a real-world case in Indonesia.

Based on the experimental results that have been conducted (USV nodes increment, targets increment, and the operation time-bound increment) over simulated random networks, we observe that the commercial solver's branch-and-bound nature dominates the computational time, which makes obtaining an optimal solution or even a feasible solution very difficult, as the size of the problem increases. On the other hand, our proposed ICH approach, designed based on the divide-and-conquer concept, can avoid performance stalling due to the branch-and-bound nature. The computational experiments indicate the increase in target numbers has more effects than the increase in USV nodes on the problem difficulty. Increasing the planning horizon length does further boost the performance of ICH, yet it is not clear to set a proper length of the planning horizon. Similarly, in the real-world case experiment based on two main harbors connecting Bali (Indonesia- Gilimanuk Harbor) and Java (Indonesia-Ketapang Harbor), the ICH has consistently shown better performances in both solution quality and computational time than the IP model.

There are some related research topics that we would suggest for further investigations. Firstly, the heuristic that we developed uses commercial solver, but it is very time-consuming due to too many iterations of the branch-and-bound operations. Therefore, developing efficient, effective heuristic algorithms for this challenging problem would be beneficial. Second, in our setting, the PB route was planned, while in the real world, how this route can be planned in a more collaborative way with USV routes is still unclear. Third, we assumed that a USV moves faster than the PB and consumes less time in the USV nodes. However, it is also possible that some USV nodes require longer time and greater battery consumptions. A more practical examination of the USV battery consumption would make the problem more practical and challenging to solve.

7. ACKNOWLEDGMENTS

I-Lin Wang was partially supported by the Ministry of Science and Technology of Taiwan under Grant MOST 105-2221-E-006 -182 -MY3, MOST 108-2221-E-006-058, and MOST 109-2221-E-006-150.

REFERENCES

Avellar, G., Pereira, G., Pimenta, L., & Iscold, P. (2015). Multi-uav routing for area coverage and remote sensing with minimum time. *Sensors*, *15*(11), 27783-27803.

Campbell, S., Naeem, W., & Irwin, G. (2012). A review on improving the autonomy of unmanned surface vehicles through intelligent collision avoidance maneuvers. *Annual Reviews in Control*, *36*(2), 267-283.

Cruz-Chávez, M., Rodríguez-León, A., Rivera-Lopez, R., & Cruz-Rosales, M. (2019). A grid-based genetic approach to solving the vehicle routing problem with time windows. *Applied Sciences*, *9*(1), 3656.

El-Hajj, R., & Dang, A., D.-C.and Moukrim. (2016). Solving the team orienteering problem with cutting planes. *Computers & Operations Research*, *74*, 21-30.

Ferrandez, S. M., Harbison, T., Weber, T., Sturges, R., & Rich, R. (2016). Optimization of a truck-drone in tandem delivery network using k-means and genetic algorithm. *Journal of Industrial Engineering and Management (JIEM)*, *9*(2), 374-388.

Galceran, E., & Carreras, M. (2013). A survey on coverage path planning for robotics. *Robotics and Autonomous Systems*, *61*(12), 1258-1276.

Garone, E., Naldi, R., Casavola, A., & Frazzoli, E. (2010). *Cooperative mission planning for a class of carrier-vehicle systems.* Atlanta, Georgia, USA: IEEE: In 49th IEEE Conference on Decision and Control (CDC).

Labadie, N., Mansini, R., Melechovský, J., & Wolfler Calvo, R. (2012). The team orienteering problem with time windows: An lp-based granular variable neighborhood search. *European Journal of Operational Research*, *220*(1), 15-27.

Mirhedayatian, S. M., Crainic, T. G., Guajardo, M., & Wallace, S. W. (2019). A two-echelon location-routing problem with synchronisation. *Journal of the Operational Research Society*, 1-16.

Mukhina, K. D., Visheratin, A. A., & Nasonov, D. (2019). Orienteering problem with functional profits for multi-source dynamic path construction. *PLOS ONE*, *14*(4), e0213777.

Yahiaoui, A.-E., Moukrim, A., & Serairi, M. (2019). The clustered team orienteering problem. *Computers & Operations Research*, *11*, 386-399.

Zhang, J., Jia, L.-m., Niu, S., Zhang, F., Tong, L., & Zhou, X. (2015). A space-time network-based modeling framework for dynamic unmanned aerial vehicle routing in traffic incident monitoring applications. *Sensors (Basel, Switzerland)*, *15*, 13874-13898.

Zhang, J., Zhang, F., Liu, Z., & Li, Y. (2019). Efficient path planning method of usv for intelligent target search. *Journal of Geovisualization and Spatial Analysis*, *3*(2), 13.