Applied Mathematical Modelling 124 (2023) 279–299

Contents lists available at ScienceDirect

Applied Mathematical Modelling

journal homepage: www.elsevier.com/locate/apm

Covering tour problem with varying coverage: Application to marine environmental monitoring

Parisa Torabi [a,*], Anna Oleynik [a], Ahmad Hemmati [b], Guttorm Alendal [a]

[a] Department of Mathematics, University of Bergen, Allégaten 41, Bergen, Norway
[b] Department of Informatics, University of Bergen, Thormøhlens Gate 55, Bergen, Norway

ARTICLE INFO

Keywords:
Covering tour problem
Mathematical modeling
Heuristics
Marine monitoring
Offshore carbon capture and storage

ABSTRACT

In this paper, we present a novel variant of the Covering Tour Problem (CTP), called the Covering Tour Problem with Varying Coverage (CTP-VC). We consider a simple graph $G = (V, E)$, with a measure of importance assigned to each node in V. A vehicle with limited battery capacity visits the nodes of the graph and has the ability to stay in each node for a certain period of time, which determines the coverage radius at the node. We refer to this feature as *stay-dependent varying coverage* or, in short, *varying coverage*. The objective is to maximize a scalarization of the weighted coverage of the nodes and the negation of the cost of moving and staying at the nodes. This problem arises in the monitoring of marine environments, where pollutants can be measured at locations far from the source due to ocean currents. To solve the CTP-VC, we propose a mathematical formulation and a heuristic approach, given that the problem is NP-hard. Depending on the availability of solutions yielded by an exact solver, we evaluate our heuristic approach against the exact solver or a constructive heuristic on various instance sets and show how varying coverage improves performance. Additionally, we use an offshore CO_2 storage site in the Gulf of Mexico as a case study to demonstrate the problem's applicability. Our results demonstrate that the proposed heuristic approach is an efficient and practical solution to the problem of stay-dependent varying coverage. We conduct numerous experiments and provide managerial insights.

1. Introduction

The importance of a healthy marine environment is reflected in the UN's declaration of the 2020 s as the ocean decade and the increasing focus on holistic management of the oceans through Ecological-based Marine Spatial Planning [1]. An intrinsic part of achieving this will be a better understanding of the marine ecosystem [2], including habitat mapping (e.g. [3]) and continuous observations (such as the Ocean Observatories Initiative [4]).

Offshore activities have to monitor the local environment, in order to verify compliance with their license, such as their Environmental Impact Assessments. Such monitoring programs can consist of a combination of fixed installations and regular surveys with moving platforms. Technologies for marine surveys are developing fast, and with the recent Lab on Chip Technology (see, for instance, [5]), the measurements can be analyzed in real-time. Since the early 1980 s, the capabilities and use of moving platforms

* Corresponding author.
E-mail addresses: parisa.torabi@uib.no (P. Torabi), anna.oleynik@uib.no (A. Oleynik), ahmad.hemmati@uib.no (A. Hemmati), guttorm.alendal@uib.no (G. Alendal).

https://doi.org/10.1016/j.apm.2023.07.024
Received 28 December 2022; Received in revised form 21 July 2023; Accepted 24 July 2023

Available online 31 July 2023

such as Autonomous Underwater Vehicles (AUVs) have been steadily increasing. Modern AUVs can operate in water depths of up to 6000 meters and have hovering capabilities [6]. Here we will use monitoring programs related to offshore geological storage of CO_2 as an example. Carbon Capture and Storage (CCS) is a greenhouse gas mitigating technology in which CO_2 is captured from large emission points, transported, and injected into geological formations for permanent storage. In particular, offshore geological formations provide vast CO_2 storage potential, [7–9].

Since the offshore geological storage complexes are selected and managed to assure long-term confinement, migration of injected CO_2 out of the storage formation is considered unlikely [7]. Still, there is a small chance that subsurface pathways, such as faults, stratigraphic traps, and spill points, might allow parts of the stored CO_2 to migrate to the surface, [10–14]. So in order to comply with regulatory requirements [15] and to ensure public acceptance of CCS technology, e.g., [16–18], environmental monitoring of storage sites is essential, see [19] for a recent review of Monitoring and Verification related to geological storage of CO_2. With the insight gained from studying the processes involved when CO_2 seeps through the seafloor as bubbles or droplets gradually dissolve into the seawater [20–22], we expect that the local environment will experience reduced pH with a potential impact on the biota [11].

Since the area to be surveyed can be large, needs to be monitored for several years, and we are looking for a weak change in CO_2 content, the marine monitoring program will impose additional costs and challenges to storage projects [10–14]. Hence, technologies and procedures to design adequate and cost-efficient monitoring programs have been pursued for the last few years. Recently several in-situ experiments [13,23,24] have been performed to assist in developing programs to detect CO_2 seeps through the seafloor. With the right combinations of sensors and platforms, the aim is to develop cost-efficient and adequate marine monitoring [25].

Hvidevold et al. [26] studied optimal deployment strategies for fixed installations on the seafloor using the probability of detecting a leak as a metric, while Oleynik et al. [27] addressed the same problem through a set cover problem. Here we focus on using AUVs, and extend the results from [28]. The use of AUVs for various underwater exploration applications has been growing in popularity, including in commercial, military, and industrial applications. As a result, a large number of AUVs have been developed for various purposes, such as surveillance, ocean exploration and bathymetric study, mapping of the ocean floor, environmental monitoring, tracking of oil spills and gas leaks, and repair and maintenance in the industry [29]. For scientific applications, AUVs like Maya [30] are used in marine biology studies and environmental monitoring, while Fòlaga [31] is a low-cost AUV employed for coastal oceanographic purposes. Spilled Oil Tracking Autonomous Buoy (SOTAB) [32], on the other hand, is designed to track spilled oil autonomously and gather oceanographic data. In [33], researchers investigated the use of AUVs for environmental monitoring of offshore facilities to improve and expand overall monitoring programs. Our focus in this article is not on certain aspects related to AUVs' navigation and path control, as we assume that an AUV follows straight-line paths at a constant speed without drift or the effect of currents. For more information on AUV path planning in the presence of ocean currents, we refer readers to [34] and [35].

This article introduces the Covering Tour Problem with Varying Coverage (CTP-VC), which considers a finite number of locations, or nodes, in the area to be monitored. These nodes could be, for example, abandoned wells or a grid of the area. An AUV with mounted chemical sensors travels between nodes with the option of staying at one node and taking repeated measurements. This not only provides information about the exact node but also tracks the history of nearby locations due to transport processes. A node is considered covered if the AUV obtains information about it from the measurements at another node. The coverage radius increases with the time spent at the node, up to a certain limit. We refer to this feature as stay-dependent varying coverage or in short, varying coverage. The limit on the coverage is determined by how much information can be effectively collected from a distance. Additionally, the AUV uses battery power for traveling and staying while taking measurements. The main goal of this work is to design an optimal AUV tour that takes into account the importance of nodes, traveling distances, duration of stay in visited nodes, and AUV battery capacity. The coverage that varies with time spent in each node, distinguishes this problem from other routing problems, and hence, we call it the Covering Tour Problem with Varying Coverage. This problem can be reduced to a Covering Tour Problem, which is NP-hard [36] and cannot be solved using exact methods in a reasonable time for a large number of nodes.

In combinatorial optimization problems, finding an optimal solution is often infeasible due to the vast solution space. Consequently, heuristics are employed to approximate the optimal solution within a reasonable time frame. However, these heuristics are usually tailored to specific problems and cannot be readily applied to other types of combinatorial optimization problems. Although they may deliver favorable results for a given test instance, they cannot handle different (still similar) problems and they might not provide promising results for different instances of the same problem.

Recently, a metaheuristic framework called Adaptive Large Neighborhood Search (ALNS) has gained traction in the field of combinatorial optimization, owing to its ability to adapt to various problem types and explore a considerable portion of the solution space in a structured manner. ALNS is widely regarded as one of the most effective single-solution metaheuristics for discrete optimization [37], having delivered state-of-the-art results for routing problems [38].

Initially designed as an approach specific to routing problems, the ALNS framework has been expanded in recent years, with a growing number of studies applying it to other problem types, such as scheduling problems [39]. The ALNS framework boasts numerous advantages. For most optimization problems, a range of high-performing heuristics already exists, which can serve as the operators in the ALNS framework. The diversity of the neighborhoods that the ALNS algorithm explores allows for structured exploration of large portions of the solution space. As a result, ALNS is highly robust, capable of adapting to the individual characteristics of different problem instances and avoiding local optima [40]. We have developed an algorithm that is inspired by the ALNS framework, but we have made several modifications to it. While we have incorporated the core concept (adaptive layer) behind the ALNS approach, we have omitted the Large Destroy and Repair component. Furthermore, we have introduced new operators that differ substantially from those used in the original ALNS framework, as proposed by [41].

Our research is inspired by the need to monitor offshore locations, but we believe that incorporating the element of time into the decision-making process for the classical Covering Salesman Problem or Covering Tour Problem could improve planning for mobile healthcare delivery systems or disaster relief teams. When delivering healthcare products to rural areas, not everyone can travel to a central location to collect them. As a result, healthcare workers or volunteers must deliver the goods, and this requires factoring in the time it takes to reach the farthest location within the station's reach (the covering distance). This will clearly lead to the necessity to take into account the time that it takes to reach the farthest location within the covering distance, since it might not be necessary to spend the time to reach the maximum covering distance in many chosen locations to stop.

In summary, this paper presents several significant contributions. Firstly, a new variant of CTP is proposed, which takes into account the realistic assumption of varying coverage. Secondly, the problem is formulated as an Integer Programming problem. Thirdly, a metaheuristic approach is developed to solve the problem, and its efficacy is demonstrated through experiments. Finally, extensive computational experiments are conducted to examine the impact of varying coverage and provide managerial insights.

This article is organized as follows. In Section 2 we place the problem in the context of routing problems and review the relevant literature. In particular, we introduce and discuss three special cases of our problem. We give a mathematical formulation of CTP-VC and its modifications that lead to the special cases in Section 3. We describe the Adaptive Metaheuristic designed for CTP-VC in detail in Section 4. In Section 5 we test the designed algorithm and an exact solver on different instance sets and investigate the benefits of varying coverage. In particular, we solve the problem using a case study of an offshore CO_2 storage site near the coast of Houston, Texas. We make a conclusion and discuss avenues for future work in Section 6.

2. Related work

CTP-VC can be approached from various perspectives. On the one hand, it extends the Prize-Collecting Traveling Salesman Problem by incorporating the notion of coverage. On the other hand, it improves upon the Covering Tour Problem by incorporating time and introducing varying coverage. We place our problem first in connection with the Profitable Tour Problem, and later in connection with the Covering Tour Problem. Moreover, as part of our experiments, we compare our problem with the Covering Salesman Problem, in which all the nodes need to be covered or visited. Therefore we give an overview of this problem and its generalizations, applications, and solving methods in the literature.

Our problem is tightly linked to the Traveling Salesman Problem (TSP) [42], one of the most well-known problems in the field of combinatorial optimization. TSP and its variations and generalization have been extensively studied, for an overview see [43] and [44]. In one of the variants, TSP with profits [45], there is a profit associated with each node, and the constraint on visiting all the nodes is lifted. The problem has two competing goals, namely collecting as much profit as possible and causing as little travel cost as possible, which means that this is a bi-objective optimization problem. One way to deal with such a problem is to consider both objective functions simultaneously and seek a set of non-dominated solutions. The authors in [46] and [47] have used this method in similar problems related to monitoring using an Unmanned Surface Vehicle (USV). The other way to deal with this type of problem is to convert the multi-objective problem to a single-objective optimization problem. This is commonly done using the Weighted-Sum Approach, which scalarizes the different objectives into one, and the ϵ-constraint method, which optimizes one of the objectives and sets bounds for the rest [48]. The Profitable Tour Problem (PTP) [49] chooses the former as the objective function. Each node in PTP is associated with a penalty that is to be paid if the said node is not visited, and traversing each edge has a specific cost. The objective then is minimizing the summation of penalties and costs. Our problem can be viewed as a generalization of the PTP. We have implemented a similar objective function with a scalarization of the collected importance of nodes and the cost of visiting and staying at nodes, with the difference that we consider a coverage distance for each stop point.

CTP-VC can also be viewed as a generalization of the Covering Tour Problem [36]. CTP is defined on a graph $G = (V \cup W, E)$, where W is a set of vertices that must be covered and $T \subset V$ is a set of vertices that must be visited. CTP aims to determine a minimum-length Hamiltonian cycle on a subset of V such that every vertex of W is within a pre-specified distance from a vertex on the cycle. A special case of CTP-VC is when the function describing the coverage radius is constant in time, evaluated as the maximum coverage radius in CTP-VC. We call this problem the Covering Tour Problem with Fixed Coverage (CTP-FC). If in CTP-FC we set the costs per unit of time and the importance of all nodes to be zero and consider unlimited battery capacity, the problem is reduced to a CTP in which there are no nodes that need to be covered nor visited.

The classical routing problems, such as the TSP and the Covering Salesman Problem (CSP) [50], enforce the obligation to visit or cover all the nodes. In our analysis, we consider a special case of CTP-VC where the limit on the energy source is lifted, and the necessity of covering all the nodes is required. This problem is called the Covering Salesman Problem with Varying Coverage (CSP-VC), and its objective is to minimize the cost of operation while ensuring full coverage of all nodes. If the covering radius is constant in time, the corresponding problem is called the Covering Salesman Problem with Fixed Coverage (CSP-FC). By setting the cost per unit of time to zero in CSP-FC, the problem is simplified to CSP.

The CSP was introduced by Current and Schilling [50] as a general form of Traveling Salesman Problem, where instead of the obligation to visit every node, all the nodes need to be either visited or within a pre-specified distance of visited nodes. The mentioned distance is known as covering distance which is fixed as a parameter in the problem. The CSP is NP-hard since it can be reduced to the TSP by setting the covering distance smaller than the distance of the closest nodes in the node-set. Therefore, a heuristic procedure is proposed in [50].

A generalized CSP (GCSP) is introduced in [51], where there is a minimum number of times for each node to be covered and a cost associated with visiting each node. They further propose two local search heuristics to be applied to GCSP variants. Another generalization has been considered by [52], where the aim is to construct a minimum-length cycle over a subset of facilities while

covering a given number of customers, with potential application in humanitarian relief transportation and telecommunication networks. They have proposed two mathematical models and two metaheuristic algorithms to solve the problem. A time-constrained maximal covering salesman problem with weighted demand and partial coverage was studied by [53]. The authors propose four branch-and-cut schemes to solve the problem.

There are several applications that can be considered for CSP. Bowerman et al. [54] studied the urban school bus routing problem, in which students need to be picked up from stops within a maximum walking distance of their residence. They develop a heuristic solution method that groups the students into clusters, and then finds the route to be taken by the bus.

A similar problem as ours has been put forward by Ma and Yang [55], where the aim is to introduce a new data gathering mechanism for mobile collectors. They call the problem Single-hop Data Gathering, and the aim is to find a minimum-length tour in a set of polling points, where each polling point has a fixed set of adjacent sensors, which all need to be covered by at least one neighboring polling point. In the special case where the neighboring set can be modeled as a disk area, this problem could be reduced to CSP. More recently, Tripathy et al. [56] have studied a CSP with 2-coverage which ensures the availability of services such as blood supply chain, border surveillance using Unmanned Aerial Vehicles (UAVs), etc., in case one provider fails at any point.

Extensive studies have been conducted on methods for solving the CSP, including [57], combining the ant colony algorithm and dynamic programming to solve the CSP and demonstrate the results on benchmark instances. Li et al. [58] have used an unsupervised deep reinforcement learning approach to learn the structural patterns and give an approximate solution. Lu et al. [59] propose a hybrid evolutionary algorithm (HEA) that exceeds the state-of-the-art methods for several benchmark instances and can be adapted to be used in the Generalized Covering Salesman Problem as well. A branch-and-cut framework combining exact and heuristic algorithms, and using a set of valid inequalities, proves effective in [60].

In all the mentioned cases of studies of covering problems, the covering distance or covering set of nodes was pre-specified. There are no studies addressing the time-expanding coverage, which has many potential applications, as mentioned in the introduction. Here we bridge this gap by introducing the coverage radius as a function of the duration of stay in each location. Therefore, the CTP-VC contains the additional complexity of determining the optimal duration of stay at nodes, which we show can be effectively addressed by the introduced Adaptive Metaheuristic.

3. Mathematical formulation

We consider a simple weighted graph $G=(V,E)$ where $V=\{v_i, i=0,1,\dots,n,n+1\}$, $n\in\mathbb{N}$ is the set of $n+2$ vertices or nodes, and $E\subseteq V\times V$ is the set of edges. Each node $v_i\in V$ is associated with an importance $W_i\geq 0$, and each edge $(v_i,v_j)\in E$ is prescribed a weight $D_{ij}\geq 0$. Each node represents a point in $\mathbb{R}^2$ and D_{ij} are Euclidean distances between these points. For simplicity, we refer to nodes by their indices in the node-set, e.g. writing $j\in V$ is equivalent to $v_j\in V$ and $(i,j)\in E$ to $(v_i,v_j)\in E$. The nodes v_0 and v_{n+1} are reserved for the starting and ending position of the vehicle path and, thus, have zero importance, i.e., $W_0=W_{n+1}=0$, and are connected to all the other nodes in V. Moreover, let $T=\{T_0+k\Delta T, k=0,\dots,h\}$, $h\in\mathbb{N}$, be a discrete bounded time set. Here T_0 is an initial time, and the time step, ΔT, can be measured in e.g., seconds or hours, and is application-specific.

A vehicle can traverse edges, moving from one node to another, with a cost C_{ij} proportional to D_{ij}. It can also spend time at a node v_i with a cost $\hat{C}_i$ per ΔT, to cover nodes within a coverage distance. This coverage distance is a time-dependent function. Here we assume it to be piece-wise linear and given as

$$r(t)=\begin{cases}\alpha t+\beta, & 0\leq t<t_{\max}\\ R, & t\geq t_{\max}\end{cases},\quad R=\alpha t_{\max}+\beta,\quad \alpha,\beta\geq 0. \tag{1}$$

The function $r(t)$ could be made node-dependent, which will not change the mathematical formulation of the problem.

The vehicle starts a tour at v_0 and finishes at v_{n+1}, and aims to either visit or cover some nodes so as to maximize the objective function, defined as a scalarization of total importance and negation of the total cost of staying and moving.

We formulate the problem as an Integer Programming problem. The parameters, sets, and decision variables are given in the Table 1. In particular, the main decision variables x_{ijt} are binary, where $x_{ijt}=1$ if and only if the vehicle moves from node v_i to node v_j at time t. The binary variables y_i determine if the node v_i is on the visiting route, and z_i indicate if the node v_i is covered by another node and not visited. The importance W_i is collected when node v_i is visited or covered, i.e. when either $y_i=1$ or $z_i=1$.

The duration of stay at node v_j is computed as

$$t_j=\sum_{k\in N_j}\sum_{t\in T} t x_{jkt}-\sum_{i\in N_j}\sum_{t\in T}(t+T_{ij})x_{ijt},\quad j\in V\setminus\{v_0,v_{n+1}\}, \tag{2}$$

where N_j is the set of nodes i for which the edge (i,j) belongs to the edge set, E.

When dealing with the varying coverage radius $r_j=r(t_j)$, it is convenient to introduce auxiliary variables s_{ij} over the edge set E as

$$s_{ij}=\begin{cases}1, & \text{if } r_j\geq D_{ij}\\ 0, & \text{otherwise}\end{cases},\quad (i,j)\in E. \tag{3}$$

The Integer Programming model for CTP-VC accounts for the flow conservation and keeping track of time, as well as varying coverage and the limit to it, whether or not a specific node is covered, and the battery capacity of the vehicle, and is given by

Table 1
Notation used in the mathematical model of CTP-VC.

Sets	
V	Set of all the nodes of the graph, $V = \{v_0, v_1, \dots, v_{n+1}\}$
E	Set of all the edges of the graph, including the edges to v_0 and v_{n+1}
N_i	Set of indices for all the nodes adjacent to node v_i, $N_i = \{j \mid (i,j) \in E, j \in V\}, i = 0, 1, \dots, n+1$
T	Set of time steps, $T = \{T_0 + k\Delta T, k = 0, \dots, h\}$
Parameters	
n	Number of nodes
C_{ij}	Cost associated with edge $(i,j) \in E$
$\hat{C}_i$	Cost of staying at node v_i per time step
W_i	importance of node v_i
D_{ij}	The shortest distance between node v_i and node v_j
sp	Speed with which the vehicle moves
T_{ij}	The time it takes to move from node v_i to node v_j, $T_{ij} = \lceil \frac{D_{ij}}{sp} \rceil$
R	Maximum coverage distance
ω	Scaling parameter corresponding to importance coverage in the objective function compared to the costs
B	Battery capacity of the vehicle
U	Battery usage of the vehicle per kilometer while moving
$\hat{U}$	Battery usage of the vehicle per time step while staying in a location
M_1	A large enough number, $M_1 \geq \max\{D_{ij}, (i,j) \in E\}$
M_2	A large enough number, $M_2 \geq n-1$
Decision Variables	
x_{ijt}	Binary variable; 1 if the vehicle starts moving from node i to node j, $(i,j) \in E$, at time step t; 0 otherwise.
y_i	Binary variable; 1 if node v_i is directly visited by the vehicle; 0 otherwise.
s_{ij}	Binary variable; 1 if node v_i is covered by node v_j, $(i,j) \in E$; 0 otherwise.
z_i	Binary variable; 1 if node v_i is indirectly covered by staying in another node; 0 otherwise.
r_i	The radius of coverage for node v_i as a function of the duration of stay at the node. $r_i = r(t_i)$.

$$\text{maximize } \omega \sum_{j \in V} W_j (y_j + z_j) - \left(\sum_{(i,j) \in E} C_{ij} \sum_{t \in T} x_{ijt} + \sum_{j \in V} \hat{C}_j t_j \right) \tag{4}$$

subject to (2) and

$$\sum_{j \in N_0} x_{0,j,T_0} = 1, \tag{5}$$

$$\sum_{i \in N_{n+1}} \sum_{t \in T} x_{i,n+1,t} = 1, \tag{6}$$

$$t_j \geq 0, \quad j \in V \setminus \{v_0, v_{n+1}\}, \tag{7}$$

$$\sum_{(i,j) \in E} x_{ijt} \leq 1, \quad t \in T, \tag{8}$$

$$\sum_{i \in N_j} \sum_{t \in T} x_{ijt} - \sum_{k \in N_j} \sum_{t \in T} x_{jkt} = 0, \quad j \in V \setminus \{v_0, v_{n+1}\}, \tag{9}$$

$$U \sum_{(i,j) \in E} D_{ij} \sum_{t \in T} x_{ijt} + \hat{U} \sum_{j \in V} t_j \leq B, \tag{10}$$

$$y_j = \sum_{i \in N_j} \sum_{t \in T} x_{ijt}, \quad j \in V, \tag{11}$$

$$r_j - D_{ij} + M_1(1 - s_{ij}) \geq 0, \quad (i,j) \in E, \tag{12}$$

$$\sum_{(i,j) \in E} s_{ij} + M_2(1 - z_i) - 1 \geq 0, \quad i \in V, \tag{13}$$

$$\sum_{(i,j) \in E} s_{ij} - M_2 z_i \leq 0, \quad i \in V, \tag{14}$$

$$y_i + z_i \leq 1, \quad i \in V, \tag{15}$$

$$x_{ijt}, s_{ij}, y_i, z_i \in \{0,1\}. \tag{16}$$

The objective function in (4) represents a total gain associated with nodes visited or covered including losses associated with these operations. Constraints (5) and (6) ensure the route starts and ends at specific nodes. Constraint (7) enforces non-negative durations of stay in each node, as defined in (2), and (8) allows only one activity per time step. Constraint (9) ensures that the vehicle leaves the node it has entered. The battery capacity of the vehicle must not be exceeded. This is ensured by (10). We also need to satisfy $\sum_{i \in N_j} \sum_{t \in T} x_{ijt} \leq 1$ for $j \in V$, to ensure that no node is visited more than once. This requirement is however already met by defining the binary variable y_i. Constraint (11) states that a node v_j is visited when the vehicle moves from one of the nodes adjacent to v_j

to v_j at any time step. Constraint (12) makes sure that $s_{ij} = 0$ when node i is not covered by node v_j and accounts for (3) in the mathematical model. Conversely, when node i is within the coverage radius of another, the value of s_{ij} will be determined based on the value of y_i. If the node is visited, its importance is already counted in the objective function, thus $s_{ij} = 0$. Otherwise, the objective function pushes s_{ij} to take the value 1. Constraints (13) and (14) make sure that $z_i = 1$ when for at least one $j \in V$ we have $s_{ij} = 1$. Constraint (15) prevents double-counting of the importance of a node when it is both visited and covered by another node. Finally, (16) sets all the decision variables to be binary.

3.1. Formulation for the special cases

In this section, we mathematically formulate three special cases of CTP-VC mentioned in Section 2, namely CTP-FC, CSP-VC, and CSP-FC.

The mathematical formulation for CSP-VC is given in a very similar model to that of CTP-VC. The only changes to make are to use (17) as the objective function, lift the battery limitation (10), and replace (15) with (18). The mathematical model for CSP-VC is given by

$$\text{minimize} \sum_{(i,j)\in E} C_{ij} \sum_{t\in T} x_{ijt} + \sum_{j\in V} \hat{C}_j t_j \tag{17}$$

subject to

$$y_i + z_i = 1, \quad i \in V, \tag{18}$$

and (5)–(9), (11)–(14), and (16).

In order to model CSP-FC and CTP-FC we consider the graph $G = (V, E)$ introduced in Section 3. The main decision variables are the binary $x_{ij}, (i,j) \in E$ which take the value 1, only when the edge (i,j) is traversed by the vehicle. The variables y_i and z_i are defined the same way as defined for CTP-VC in Table 1. In these problems, the duration of stay in each node is pre-specified, as it is in the classical CSP. This value is determined by $t_{max} = \lfloor (R-\beta)/\alpha \Delta T \rfloor$. Thus, the two parameters $\tilde{C}_j$ and $\tilde{U}_j$ are defined as $\tilde{C}_j = t_{max}\hat{C}_j$ and $\tilde{U} = t_{max}\hat{U}$, respectively. Moreover, the set of the nodes covered by each node is given by $S_j = \{i \in \{1, \dots n\} \mid D_{ij} \le R\}$. CTP-FC is given by

$$\text{maximize } \omega \sum_{j\in V} W_j (y_j + z_j) - \left(\sum_{(i,j)\in E} C_{ij} x_{ij} + \sum_{j\in V} \tilde{C}_j y_j \right) \tag{19}$$

subject to

$$\sum_{j\in N_0} x_{0j} = 1, \tag{20}$$

$$\sum_{i\in N_{n+1}} x_{i,n+1} = 1, \tag{21}$$

$$\sum_{i\in N_j} x_{ij} - \sum_{k\in N_j} x_{jk} = 0, \quad j \in V, \tag{22}$$

$$U \sum_{(i,j)\in E} D_{ij} x_{ij} + \sum_{j\in V} \tilde{U}_j y_j \le B, \tag{23}$$

$$y_j = \sum_{i\in N_j} x_{ij}, \quad j \in V, \tag{24}$$

$$z_j \le \sum_{i\in S_j} y_i, \quad j \in V, \tag{25}$$

$$y_i + z_i \le 1, \quad i \in V, \tag{26}$$

$$\sum_{\substack{(i,j)\in E \\ i,j\in S}} x_{ij} \le |S| - 1, \quad S \subseteq V \setminus \{v_0, v_{n+1}\}, 2 \le |S|, \tag{27}$$

$$y_i, z_i, x_{ij} \in \{0,1\}. \tag{28}$$

In this formulation, (19) to (24) are corresponding to (4) through (6), and (9) to (11). Constraint (25) ensures that a node is considered covered when at least one of the nodes in its R-radius is visited. Constraint (26) plays the same role as (15). Constraint (27) states that we are looking for solutions without subtours. The size of $S \subseteq V$ need not be limited from above, since v_0 and v_{n+1} are not included in the subsets of V. Constraint (28) sets all the decision variables to be binary.

The mathematical formulation for CSP-FC uses the exact same equations as CTP-FC, except for taking (29) as the objective function, removing the battery constraint (23), and replacing (26) by (30) to ensure that all nodes are either directly visited or indirectly covered by visiting another node. Thus the mathematical formulation for CSP-FC is given by

$$\text{minimize} \sum_{(i,j)\in E} C_{ij}x_{ij} + \sum_{j\in V} \tilde{C}_j y_j \tag{29}$$

subject to

$$y_i + z_i = 1, \quad i \in V, \tag{30}$$

and (20)–(22), (24), (25), (27), and (28).

4. Adaptive metaheuristic algorithm

As mentioned in Section 2, CTP-VC is a generalization of CTP, which is an NP-hard problem. Therefore, heuristic-based methods are needed to solve this problem. Later in Section 5, we show CPU time for running experiments that confirm this. In this section, we describe the developed Adaptive Metaheuristic algorithm for CTP-VC, given in Algorithm 1, and its components.

Algorithm 1 Adaptive Metaheuristic algorithm for CTP-VC.

Require: maximum iterations i_{max}, segment iterations i_s, warm-up iterations i_{wu}, maximum non-improving iterations i_{esc}, objective function f, heuristic set $\mathcal{H}$, initial heuristic weights $\mathcal{W}_{h,0}$
Ensure: s_{best}
  Generate an initial solution s as described in Section 4.2
  $s_c \leftarrow s$, $s_{best} \leftarrow s$
  $i_{ni} \leftarrow 0$ set iterations passed since s_{best} was updated
  for i = 1 to i_{max} **do**
    if i = i_{wu} **then**
      Find $\mathcal{T}_0$ and γ as described in Section 4.5
    else if i < i_{wp} **then**
      $P \leftarrow p_s$
    else
      $\mathcal{T}_{i+1} \leftarrow \gamma \mathcal{T}_i$
    end if
    if $i_{ni} \geq i_{esc}$ **then**
      $s_c \leftarrow$ result of performing the escape algorithm from Section 4.6
      $i_{ni} \leftarrow 0$
    end if
    Select a heuristic $\hat{h}$ from $\mathcal{H}$ listed in Section 4.3based on $\mathcal{W}_{h,i}$
    $s_{new} \leftarrow$ result of applying $\hat{h}$ on s_c.
    if s_{new} is feasible **then**
      $\Delta E \leftarrow f(s_{new}) - f(s_c)$
      if acceptance criteria from Section 4.5is met **then**
        $s_c = s_{new}$
        if $f(s_{new}) > f(s_{best})$ **then**
          $s_{best} \leftarrow s_{new}$, $i_{ni} \leftarrow 0$
        else
          $i_{ni} \leftarrow i_{ni} + 1$
          if $i < i_{wu}$ **then**
            store ΔE
          end if
        end if
      end if
      Update $\pi_{h,i}$ according to Table 3
    end if
    if iterations in a segment is ended **then**
      update heuristic weights $\mathcal{W}_{i,h}$ as described in Section 4.4
    end if
  end for

4.1. Solution representation

There are three concurrent decisions to be made when solving the CTP-VC problem. Namely the set of nodes to be visited, the order in which they are visited, and the duration of stay in each node. The solution representation is designed to consider the three decisions in two vectors. The first vector reflects the first two decisions, while the second shows the values corresponding to the third decision.

The first vector has size $n + 1$ and presents the nodes the vehicle visits in the order of their visit, followed by a separator and the nodes that are not visited. The number of nodes that the vehicle visits is n_v. Note that the nodes that come after the separator, i.e., unvisited nodes, might be covered or not, depending on the duration of stay in the visited nodes and their distance from the unvisited node in question. The second vector has the size n_v, and is concerned with how many time steps the vehicle stays in each node it visits. When the dependence of the radius of coverage and the duration of stay is described using (1), the possible values

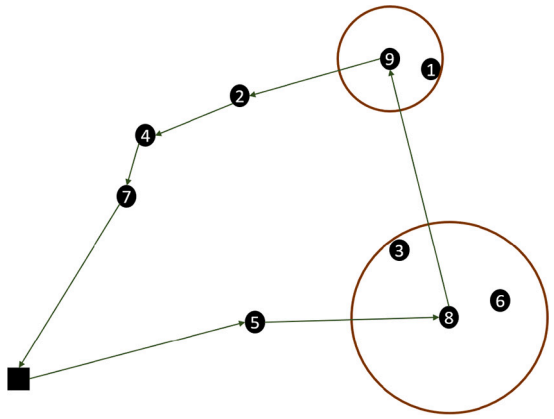


Fig. 1. A sample solution.

Table 2
Heuristics.

Heuristic number	Description
H1	Shaw remove and Insert greedy with the best duration
H2	Shaw remove and Insert greedy with zero duration
H3	Random remove and Insert greedy with the best duration
H4	Random remove and Insert greedy with zero duration
H5	Worst remove and Insert greedy with the best duration
H3	Worst remove and Insert greedy with zero duration
H7	Improve duration greedy
H8	Improve duration by 1 time step

for the entries of this vector are integers between 0 and $\lfloor (R-\beta)/(\alpha \Delta T) \rfloor$. An example of a solution is given in Fig. 1, for which the solution representation consists of the two vectors $[5,8,9,2,4,7,X,6,3,1]$ and $[0,5,3,0,0,0]$.

4.2. Initial solution

The algorithm starts with a feasible initial solution which is obtained by clustering nodes using the k-means method and adding the nodes closest to the centers of clusters to the visiting path (as many as feasibility allows). The initial values for the duration of stay for these nodes are set to zero. The number of clusters is chosen as the best number between two and $\min\{n, 10\}$, according to the *silhouette* analysis [61]. The silhouette analysis attempts to find the optimal number of clusters in a data set. The silhouette coefficient or silhouette score is a measure of similarity of a data point within-cluster, compared to other clusters. In our analysis, we set the number of clusters, k to each value in the range two to $\min\{n, 10\}$ and compute this coefficient for all data points in the solution of the k-means method. The average value of silhouette coefficients is a measure of goodness for the clustering for a specific value of k. Finally, the k which yields a clustering with the largest average silhouette score is picked and used to generate the initial solution.

4.3. Heuristics

There are eight heuristics that are designed to alter the solutions at each iteration of the algorithm. Each of them works on changing the solution either in the visiting path or the duration of stay at nodes or both vectors. The first six heuristics are combinations of one removal and one insertion heuristic. They first choose a random integer, q, between 2 and 5, then use their removal heuristic to remove q nodes from the incumbent solution, and finally insert the q nodes using their insertion heuristic. An overview of the eight heuristics is given in Table 2.

4.3.1. Random remove

This heuristic chooses q random nodes from the first part of the solution representation and removes them from the solution. In case any of the nodes on the visiting path are to be removed, their corresponding stay duration elements are also removed.

4.3.2. Shaw remove

This removal heuristic is inspired by the one introduced by Shaw [62,63]. Here we attempt to remove q similar nodes, measured by their distance in this problem. We start by selecting one random node to be removed, let us call it v_s. Then we sort the nodes based on their proximity to v_s and call this list L.

Table 3
Scoring system.

Score	Condition
1	If the heuristic yields a feasible solution
2	If the solution produced by the heuristic is accepted according to acceptance criteria
4	If the heuristic has produced a solution that has a larger objective value than the incumbent solution

We choose a random number $y \in [0, 1)$ and consider a randomization parameter $p = 6$ to avoid repeating the same sequences and introduce randomness in the process. We choose the node in the position $y^p|L|$ in L to add to the list of nodes we want to remove. This process continues until q nodes are chosen to be removed. Finally, the chosen nodes are removed from the solution and in case they had been on the visiting path before removal, their corresponding duration of stay elements are also removed.

4.3.3. Worst remove

This removal heuristic removes q different nodes based on the change in the objective function when they are removed from their respective positions in the incumbent solution, i.e. $cost(i, s) = f(s) - f(s')$, where f is the objective function, s the incumbent solution and s' the solution obtained by removing node i from s. This means the nodes which result in the highest *cost* for the incumbent solutions are most likely to be removed. However, this removal is randomized in the same way as Section 4.3.2, with sorting the nodes and choosing a random number that determines the node to be removed.

4.3.4. Insert greedy with the best duration

This heuristic aims to insert the removed nodes into a non-complete solution, in order to gain the most possible benefit to the objective function. We insert one node at a time at the position which is the least costly to the objective function. The chosen node is the one that increases the objective function the most. If this location is on the visiting path for the vehicle, the best duration of stay for the said node is also set, with the same algorithm as in Section 4.3.6.

4.3.5. Insert greedy with zero duration of stay

This heuristic is a special case of heuristic in Section 4.3.4. Here, if the inserted node is on the visiting path, the duration of stay at the node is set to zero.

4.3.6. Improve duration greedy

This heuristic chooses one random node in the visiting path, and for this specific node, changes the duration of stay to the best possible value, while keeping other parts of the solution. In other words, if by increasing the stay duration the benefit of covering the additional nodes compensates for the additional staying cost, changes the staying duration to such value that gains the most in the objective function, and vice versa in case of decreasing staying duration.

4.3.7. Improve duration by 1 time step

This heuristic works in the same manner as the one in Section 4.3.6, except that it attempts a change of one time step. Here we choose a random node in the visiting path and change the duration of stay in it by only one time step so that it benefits the objective function the most. If neither increasing nor decreasing the duration of stay by one time step is beneficial, the duration remains the same as before.

4.4. Adaptive weight adjustment

At each iteration of the algorithm, one heuristic is chosen to act upon the incumbent solution. The selection is based on the *roulette wheel selection* [64] method. The probability with which each heuristic is selected is $P_{h,i}$ defined as

$$P_{h,i} = \frac{\mathcal{W}_{h,i}}{\sum_{g \in \mathcal{H}} \mathcal{W}_{g,i}},$$

where $\mathcal{H}$ is the set of available heuristics, and $\mathcal{W}_{h,i}$ denotes the weight of heuristic h at segment i.

This weight depends on its performance in the previous iterations [41]. For this reason, we will divide the iterations into segments, and when one segment of iterations is over and the scores of heuristics have been collected, update these weights using

$$\mathcal{W}_{h,(i+1)} = (1 - r)\mathcal{W}_{h,i} + r\frac{\pi_{h,i}}{\theta_{h,i}},$$

where r is the *reaction factor* which determines how much the change in performance of a heuristic in the recent segment is reflected in its weight in the next segment. $\pi_{h,i}$ show the score that heuristic h has gained during segment i and $\theta_{h,i}$ is the number of times the heuristic h is used during segment i. The scoring system used in this study is given in Table 3.

4.5. Acceptance criteria and stopping condition

The stopping criterion is reaching the maximum number of iterations. When a new solution is obtained using a heuristic, we need to decide whether or not we should update our incumbent solution. If we accept any new solution irrespective of its objective, we would end up with *Random Search*. On the other hand, in case we only accept solutions that improve objective function, we are performing a *Local Search*.

By introducing the acceptance criteria of Simulated Annealing [65] we begin our search in the solution space by a larger probability of accepting new worse solutions and as the iterations increase, we move towards more intensification. This can be implemented using the cooling schedule suggested in [66], whereby the probability of acceptance of a solution s' when the incumbent solution is s, with $f(s') \leq f(s)$ is given by Boltzmann probability, $\exp((f(s') - f(s))/\mathcal{T}$ where f is the objective function and the temperature, $\mathcal{T}$ is updated each iteration i by $\mathcal{T}_{i+1} = \gamma \mathcal{T}_i$.

The cooling schedule should start by setting the temperature to a starting temperature, $\mathcal{T}_0$. Moreover, the cooling rate γ should be determined. Here we have used *warm-up iterations* in order to set a baseline. In these iterations, the acceptance probability is a fixed $p_s = 0.8$ and we keep account of $\Delta E = f(s') - f(s)$ whenever $f(s') \leq f(s)$. Then the starting temperature, $\mathcal{T}_0$, is computed as

$$\mathcal{T}_0 = \frac{\overline{\Delta E}}{\ln(p_s)},$$

where $\overline{\Delta E}$ is the average of ΔE over the iterations in the warm-up phase. To compute the cooling rate, we consider the final temperature, $\mathcal{T}_f = 0.1$ which is the desired temperature by the last iteration. Thus γ can be obtained using

$$\gamma = \left(\frac{\mathcal{T}_f}{\mathcal{T}_0}\right)^{\frac{1}{i_{cd}}},$$

where cool down iterations, i_{cd}, is the difference of maximum iterations and warm-up iterations, $i_{cd} = i_{max} - i_{wu}$.

4.6. Escape algorithm

In order to prevent the algorithm to get stuck in a local optimum point, especially when the nodes are spread out in an instance, an escape heuristic is introduced. It is called to act upon the incumbent solution whenever the algorithm has not improved the best solution for 500 iterations. Its objective is to introduce new solutions that can potentially lead to an improved overall solution, thereby enhancing the algorithm's robustness. This serves as an additional diversification component within the algorithm. This heuristic essentially forces q nodes to be added to the visiting path, in the best possible positions, and with a duration of stay of 0.

5. Computational experiments

In this section, we describe the conducted computational experiments in Section 5.1, and give their results in Section 5.2.

We use a time limit of 10000 seconds whenever we run the exact solver on an instance. We set the inputs of Algorithm 1 as follows. Maximum number of iterations $i_{max} = 10000$, segment iterations $i_s = 100$, warm-up iterations $i_{wu} = 100$, maximum non-improving iterations $i_{esc} = 500$, and initial heuristic weights $\mathcal{W}_{h,0} = 1/|\mathcal{H}|$ are uniform among all given heuristics. The parameters p, r, p_s, and T_f undergo a process of parameter tuning to determine appropriate values. The complete range of combinations, including $p \in \{5, 6, 7, 8\}$, $r \in \{0.1, 0.15, 0.2, 0.25, 0.3\}$, $p_s \in \{0.75, 0.8, 0.85\}$, and $T_f \in \{0.001, 0.01, 0.1\}$, is exhaustively tested using 30 validation instances. Through this analysis, the optimal parameter values are identified as $(p, r, p_s, T_f) = (6, 0.2, 0.8, 0.1)$.

The computational experiments consist of four parts, and we use three instance sets described in Section 5.1 to conduct them. Section 5.2.1 discusses the extent to which the varying coverage improves the solutions compared with the case of fixed coverage, with three values for $R > 0$. To that end, CTP-VC, CTP-FC, CSP-VC, and CSP-FC are solved for instances in the instance set 1. In Section 5.2.2 the performance of Adaptive Metaheuristic 1 is demonstrated by solving CTP-VC on instances in instance sets 1 to 3. Instances of size 20 from instance set 1 are solved in Section 5.2.3 and Section 5.2.4. In Section 5.2.3 the effect of parameters representing battery consumption and battery capacity, as well as the ones relevant to varying coverage, is studied. In Section 5.2.4 we discuss how much can be gained when the varying coverage is incorporated into the classical fashion in which the AUVs take measurements.

5.1. Instance sets

There are three sets of instances used in this study to look into the different features of the problem and demonstrate the performance of the proposed Adaptive Metaheuristic algorithm. The aim of studying the first set is to form a baseline for solving the problem and identify the limits of solving the problem with an exact method. However, in practice, we normally encounter instances with a very large number of nodes. Thus in the second set, we increase the number of nodes in instances without changing other parameters. Since we reach the limits of the exact solver in the first instance set, this set also acts as an indicator of the performance of the Adaptive Metaheuristic Algorithm 1 in terms of running time and solution quality when increasing the size of the problem. Finally, the third set includes instances that are derived from the application that has formed our inspiration, and we utilize the developed algorithm in a real-world case study.

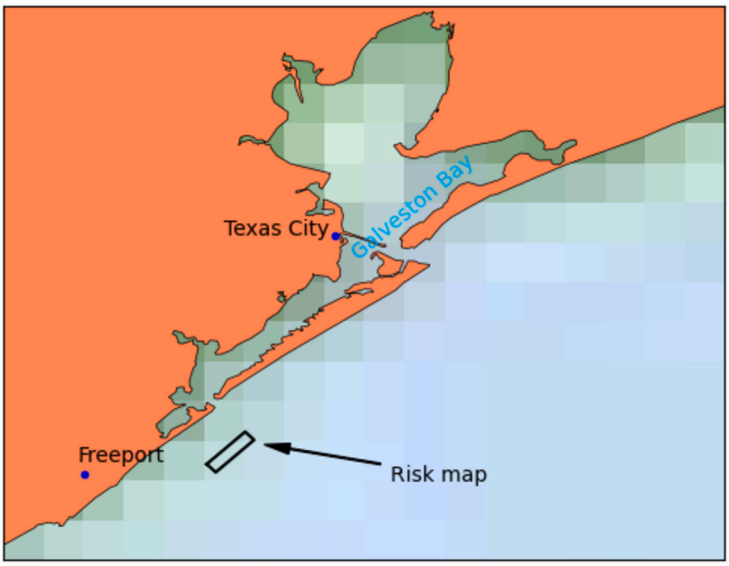


Fig. 2. Map of the monitoring area location.

For all the instance sets, the graph is considered to be complete, since the possibility of existing obstacles in the ocean is small. The end point is set to be the same as the start point for all instances.

Instance set 1 contains instances with size $n = 5, 6, \ldots, 20$. For each size, 5 different instances are generated by setting the origin as the start and end point of the vehicle's planned mission and choosing n locations uniformly at random in a 50×50 area. Further, $\hat{C}_i$ for each node is a random integer between 0 and 10, and W_i is a random integer between 1 and 100. The baseline values for the remaining parameters are $C_{ij} = D_{ij}$, $\alpha = 1.5$, $\beta = 0$, $R = 15$, $B = 400$, $sp = 1$, $\omega = 1$, $U = 2$, and $\hat{U} = 1$. This set comprises a total of 80 instances.

Instance set 2 is made up of instances with sizes $n = 50$, 100, 150, 200, 250. For each size 5 instances are generated. In total, this set consists of 25 instances. We generate instances in this set in the same way as instance set 1.

Instance set 3 contains 5 instances per each size, with $n = 50$, 100, 150, 200, 250. The area is located in the western Gulf of Mexico, just outside of Galveston Bay, see Fig. 2.

The probability for leakage in the area indicated in Fig. 2 has been obtained from geological risk mapping, which is a part of the ACTOM[1] toolbox, for details see [67,68]. The nodes are randomly chosen from this probability distribution, and their importance is taken to be proportional to the probability of leakage. The time step ΔT is set to 6 minutes, and distance is measured in meters. As for the AUV-related parameters, we use those of a double-hull Sabertooth.[2] A double-hull Sabertooth has the forward speed of 4 knots, which gives $sp = 740.8$ m$/\Delta T$. It has battery capacity of 30 kWh and the endurance of 14 h. From this, we estimate the battery usage while moving as $U = 2.89 \times 10^{-4}$kWh/m and while staying as $\hat{U} = 2.14 \times 10^{-2}kWh/\Delta T$. The latter is obtained as 10% of U recalculated in kWh$/\Delta T$ units with sp given above.

The varying cover, $r(t)$, is based on simulations of gas seep in the Gulf of Mexico [67,68], see Fig. 2. Transport and dilution of CO_2 from 75 seep locations were simulated, based on current conditions from a General Circulation Model (GCM) hindcast for the Gulf of Mexico [69]. We neglect the anisotropy in the resulting footprints and calculate the average increase in CO_2 concentration as a function of distance to the source for the seeps, $\Delta\mu(r)$. From standard power analysis, we can estimate how many measurements, $Nm(r)$, we need at a given distance r from the source, to reject the null hypothesis that there is no seep in the area, i.e., $H_0 : \Delta\mu(r) = 0$, i.e., no seep present, as opposed to the alternative $H_1 : \Delta\mu(r) \neq 0$, there is a seep nearby:

$$Nm(r) = \left(\frac{Z_{1-(\text{T-I})/2} + Z_{1-(\text{T-II})}}{\left(\frac{\Delta\mu(r)}{\sigma(r)} \right)} \right)^2, \tag{31}$$

where $Z_{1-(\text{T-I})/2} = 1.96$ and $Z_{1-(\text{T-II})} = 0.84$. Here we have used T-I and T-II instead of the commonly used α and β, representing Type I and Type II errors in statistical hypothesis testing, in order to distinguish them from the notation used in (1). Thus $Z_{1-(\text{T-I})/2}$ and $Z_{1-(\text{T-II})}$ are the values from the normal distribution reflecting the confidence levels for, respectively, avoiding type I (wrongfully rejecting H_0) and type II (failing to reject it) errors. We have used the frequently used T-I = 0.95 and T-II = 0.8.

[1] https://actom.w.uib.no.

[2] https://www.saabseaeye.com/solutions/underwater-vehicles/sabertooth-double-hull.

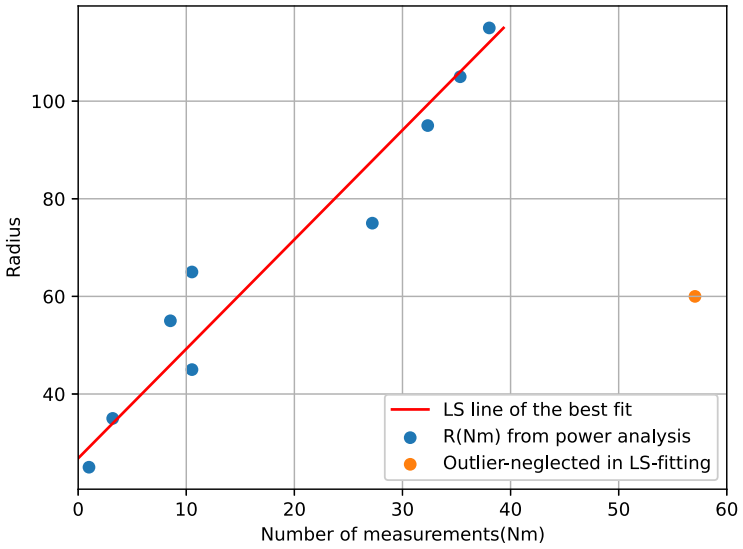


Fig. 3. Radius of coverage as a function of number of measurements, an indirect time.

For σ we have used the standard deviation in measured variability of CO_2 concentration from the GOMECC-1 cruise, $\sigma = 41\ \mu mol/kg_{sw}$ [70]. Since the transport simulations are non-dimensionalized, we scaled $Nm(r_0) = 1$, with $r_0 = 25$ m, i.e. the flux rate is high enough for a leak to be detected immediately if a measurement is taken within the 25 m radius distance from the source.

We set $R = 120$ m, and removed the outliers. Then the data was fitted to a linear function using the Least Squares method. See Fig. 3, in which the points indicated by dots are obtained from the GCM simulations and (31). The outlier, shown in orange, was neglected in the linear least square fitting of the straight line, $r(Nm) = 0.45Nm - 11.97$. We obtain $r(t)$ as in (1) with $\alpha = 2.24\,m/\Delta T$ and $\beta = 26.83$ m. The values are rounded off to two decimals. The other parameters are set as $C_{ij} = 740 \times D_{ij}, \hat{C} = 1$, and $\omega = 100$.

5.2. Computational results

Here the computational results are given for the four categories of experiments described in Section 5. For experiments in Section 5.2.1, all instances in instance set 1 are attempted to be solved using Gurobi 9.5.0, with the time limit set to 10000 seconds. For some instances, no feasible solution was found for CSP-VC using Gurobi. Thus, the Adaptive Metaheuristic was adjusted to this problem and run for the instances to report non-trivial solutions. The given results are obtained using Gurobi for CTP-FC and CSP-FC, and the best-known solution between Gurobi and Adaptive Metaheuristic for CTP-VC and CSP-VC.

In Section 5.2.2 all instances are attempted to be solved using Gurobi with a time limit of 10000 seconds. Some of these attempts result in reaching the time limit in instance set 1, and all of them in instance sets 2 and 3 terminate due to memory issues and do not return any feasible solutions. Therefore, we use a constructive heuristic as a baseline for instance sets 2 and 3. All instances are solved using the Adaptive Metaheuristic and the performance is compared to baseline values.

Section 5.2.3 describes the effect of changing a few parameters on solutions, and since the number of different experiments is very large, 410 experiments run in total, the Adaptive Metaheuristic 1 is used to run these experiments. In Section 5.2.4 we investigate the gained collected importance or saved costs, when in the process of data-gathering, we choose to have repeated measurements in a place, incorporating the varying coverage.

5.2.1. Effect of the varying coverage

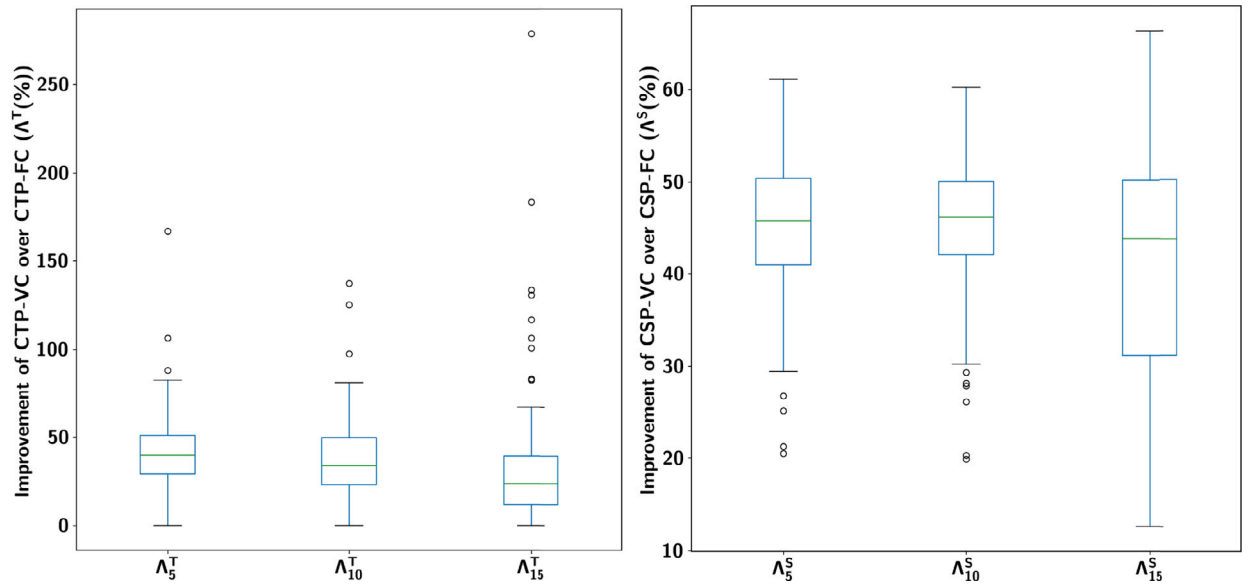
For any $R \leq 15$, an optimal solution of a problem (CTP or CSP) with fixed coverage is a feasible solution of the problem with varying coverage. Thus, the objective value for CTP-VC is always greater than or equal to that of CTP-FC. Similarly, the objective value for CSP-VC is less or equal to the objective value of CSP-FC.

Here we investigate how much more gain is obtained by CTP-VC and how much cost is saved by CSP-VC. We also touch upon the CPU time used by the exact solver for finding the solutions.

Let f_V^T be the objective value of the best-known solution of CTP-VC in (4) with $R = 15$, and $f_{F,R}^T$ the objective value of the solution of CTP-FC in (19) with fixed $R \in \{5, 10, 15\}$. Similarly, f_V^S is the objective value of the solution of CSP-VC defined in (17), and $f_{F,R}^S$ is the objective value of the solution of CSP-FC given in (29) with $R \in \{5, 10, 15\}$.

For each instance, the percentage improvement in objective function when CTP-VC is used instead of CTP-FC is defined as

$$\Lambda_R^T = \frac{f_V^T - f_{F,R}^T}{f_{F,R}^T} \times 100\%, \quad R = 5, 10, 15.$$



(a) Λ_R^T for $R = 5, 10, 15$.

(b) Λ_R^S for $R = 5, 10, 15$.

Fig. 4. Improvement of objective value, when using varying coverage instead of fixed coverage, with different values of R.

When comparing CSP-VC and CSP-FC with the minimizing objective function, the improvement in the objective value is defined as

$$\Lambda_R^S = -\frac{f_V^S - f_{F,R}^S}{f_{F,R}^S} \times 100\%, \quad R = 5, 10, 15.$$

As expected, both CTP and CSP with VC outperform the corresponding problem with FC for all instances, for any of the three tested values for R.

The Λ_R^T values for instances in instance set 1 are given in Fig. 4a. On average, CTP-VC yields objective values that are 43.1% more than CTP-FC with $R = 5$. The corresponding average improvement for the case of $R = 10$ and $R = 15$ are 39.3% and 35.3%, respectively. While we cannot make a general statement about the objective values of CTP-FC with different values of R for every individual instance, under the implicit assumption that the cost of staying is relatively low and there is a greater emphasis on the node coverage than the cost, it is expected that increasing R will lead to an increase in the objective value. This trend is observed when averaging over all instances in instance set 1, as the average improvement resulted from the varying coverage decreases with increasing R values, indicating that with fixed coverage, the larger R results in better objective values.

The values of Λ_R^S are shown in Fig. 4b. Taking the average over all instances in instance set 1, CSP-VC gives objective values that are 45.4%, 45.4%, and 41.6% less than the ones obtained from CSP-FC with $R = 5, 10, 15$, respectively. The CSP-FC performs slightly better with the larger value of R when considering average improvements.

It is worth noting the average CPU time taken for Gurobi to solve the instances of CTP and CSP with both varying coverage and fixed coverage, for the same value of R, given in Table 4. As expected, the varying coverage immensely affects the complexity of the problem and thus the CPU time. This confirms the necessity of approaching this problem with heuristics, see Section 4. We discuss the effectiveness of Algorithm 1 in Section 5.2.2.

5.2.2. *Effectiveness of the adaptive metaheuristic*

Here we focus on solving our original problem, namely CTP-VC. The exact solver is run to solve this problem using the mathematical model with Gurobi for all instances in all instance sets. While the exact solver yielded solutions for the first instance set, optimal or otherwise, it ran out of memory for all instances in instance sets 2 and 3 and was not able to find any feasible solutions. Thus, we use a constructive heuristic as the baseline for these two sets and bring the results separately. We call the constructive heuristic *LKH & Radius Improvement.*

The Adaptive Metaheuristic Algorithm 1 was run 10 times with different random seeds for each problem instance. The objective value from each run and average CPU time over the 10 runs have been reported. We calculate two measures for each of the instances. The best-known objective value and the Gap to Best Known. The best-known objective value is the maximum of all the 10 objective values and the objective value from the exact solver if existing. Gap to Best Known is defined as $\text{GBK} = -(f(s) - f(bk))/f(bk)$, where f is the objective function evaluated in the solution s of a single run of Adaptive Metaheuristic, and $f(bk)$ is the best-known objective value for the instance. The averages over 5 instances per size of the minimum and average of GBK for instance set 1 as well as the results of the exact solver are given in Table 5.

Table 4
Average CPU time for exact solver on CSP and CTP for each size.

Problem Size	CTP		CSP	
	Fixed Coverage seconds	Varying Coverage seconds	Fixed Coverage seconds	Varying Coverage seconds
5	0.04	29.68	0.23	32.63
6	0.05	67.98	0.04	57.51
7	0.05	99.37	0.04	110.80
8	0.07	270.86	0.09	176.48
9	0.09	275.19	0.10	480.92
10	0.19	690.52	0.18	531.14
11	0.30	699.80	0.29	749.85
12	0.49	2149.85	0.81	1849.92
13	1.05	2515.60	1.18	3139.95
14	3.18	3878.06	4.04	7420.64
15	9.42	5641.09	8.93	7012.44
16	22.18	6659.05	30.17	8605.37
17	48.01	4781.51	58.77	6761.19
18	183.63	8571.05	122.95	6505.32
19	229.65	9813.56	255.04	8736.40
20	1060.36	9266.68	580.06	8963.39

Table 5
Average results for instance set 1.

Problem Set	#nodes	Exact Solver			Adaptive Metaheuristic		
		Duality Gap (%)	Gap to Best Known (%)	Seconds	Minimim Gap to Best Known (%)	Average Gap to Best Known (%)	Seconds
1	5	0	0	50.4	0	0	9.61
	6	0.86	0	218.1	0	0	9.96
	7	0	0	93.9	0	0	10.82
	8	0	0	395.8	0	0	11.32
	9	0	0	569.2	0	0	12.98
	10	0	0	498.2	0	0	12.18
	11	0	0	931.1	0	0	11.69
	12	0	0	2822.4	0	0	13.82
	13	0	0	3796.9	0	0	14.71
	14	0	0	2434.5	0	0	14.95
	15	1.66	0.56	6258.3	0	0	14.03
	16	17.87	8.36	5928.6	0	0	18.78
	17	0	0	5528.0	0	0.03	16.36
	18	4.45	2.27	8566.2	0	0.83	16.50
	19	0.53	0.26	7031.7	0	0.33	17.35
	20	3.41	0.66	7501.0	0	0.04	18.74

Attempting to solve the problem using the exact solver, we see that for the first set, the optimal solution is found in small sizes, but with increasing size, it tends to reach the time limit without reaching the optimal solution. In particular, of the 5 instance of each size, one instance with size 6, one instance with size 15, two instances with size 16, 3 instances with size 18, one instance with size 19, and three instances with size 20 are not solved to optimality. From the results, we report the average Duality Gap for 5 instances per size.

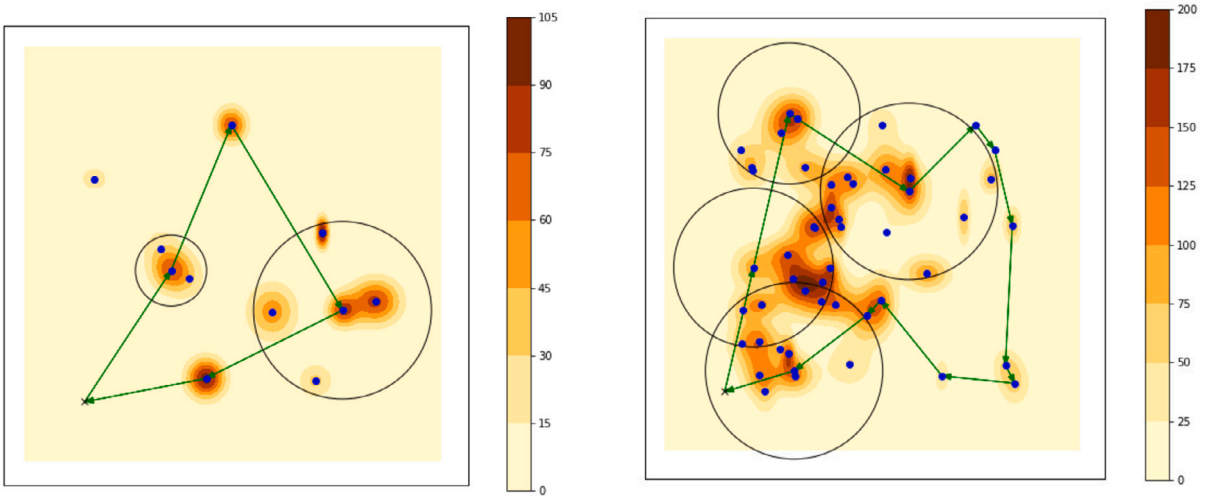
For the instances of size 5 through 14 and size 17, the exact solver reaches the best-known solution, but beyond this size, the Adaptive Metaheuristic Algorithm 1 outperforms the exact solver. The average results for this set are given in Table 5.

As mentioned at the beginning of this section, we are unable to get solutions from the exact solver for instance sets 2 and 3. Instead, we use a constructive heuristic called LKH & Radius Improvement as a baseline for our heuristic. This heuristic makes decisions on the tour and the duration of stay in each node consecutively. We first solve a TSP using Lin-Kernighan heuristic [71] implemented by Keld Helsgaun (LKH) [72]. Next, we optimize the duration of stay in each node by running a loop through all the nodes in the TSP tour. This step aims to remove some nodes from the tour and cover them with the least possible cost. The algorithm takes care of the feasibility of solutions, considering battery capacity and unnecessary node coverage. While this algorithm is not expected to reach an optimal solution, it can provide a lower bound. The averages over 5 instances per size of the minimum and average of GBK for instance sets 2 and 3 are given in Table 6.

The solutions of two instances in instance sets 1 and 2 are shown in Fig. 5. The background color is an indicator of the importance of each point, particularly the nodes. The circles show the coverage of each node. The solution for one instance in instance set 3 with 100 nodes is shown in Fig. 6. Since the area is very large, we have zoomed in on one part of the tour. Moreover, to distinguish the different coverage radius values, the coverage area for the nodes in which the duration of stay is non-zero is shown with a thicker line-width.

Table 6
Average results for instance sets 2 and 3.

Problem Set	#nodes	LKH & Radius Improvement		Adaptive Metaheuristic		
		Gap to Best Known (%)	Seconds	Minimim Gap to Best Known (%)	Average Gap to Best Known (%)	Seconds
2	50	10.01	13.90	0	1.18	19.32
	100	12.03	32.52	0	1.55	28.34
	150	7.95	43.57	0	1.53	30.60
	200	11.63	48.47	0	1.71	46.94
	250	14.29	103.67	0	1.81	40.65
3	50	65.98	144.64	0	0.24	69.09
	100	82.20	154.65	0	0.18	127.89
	150	84.52	171.35	0	0.30	183.89
	200	79.73	176.67	0	0.80	227.81
	250	87.30	206.24	0	1.12	302.02



(a) The optimal solution for an instance with 11 nodes in instance set 1.

(b) Best-known solution for an instance with 50 nodes in instance set 2.

Fig. 5. The solution shown in Figure (a) traverses four nodes with a duration of stay [4,0,10,0]. The solution shown in Figure (b) traverses the 12 nodes with a duration of stay [9,8,10,0,0,0,0,0,0,0,0,10]. The color bar represents the importance of the nodes.

5.2.3. Impacts of parameter changes on the solutions

Here we investigate how different parameters impact the solution of CTP-VC. In particular, we focus on the covering radius, that is, α and R, and the battery consumption $U/\hat{U}$ and B. We use the instances with size 20 from instance set 1, and the baseline values $\alpha = 1.5$, $R = 15$, $U = 2$, $\hat{U} = 1$, $B = 400$ which we mark in boldface further on.

First, we consider the parameters associated with the radius. The R parameter represents the maximum coverage radius of the moving vehicle, in our application the maximum coverage of the sensors. It directly affects the solution in the sense of determining the maximum duration of stay in each node, t_{max}. It is obvious that an increase in the value of this parameter can only improve the objective value, since CTP-VC provides flexibility for the duration of stay. Therefore, if given more freedom to choose the value for this variable, i.e. given a larger R value, in the worst-case scenario, it will obtain a solution with the duration of stay less than t_{max} in all the nodes.

When considering the linear function (1) to describe R, we can look into the effect of α on the solution. However, changing α when keeping the R value constant implies changing t_{max} indirectly. Thus it is more convenient to consider the independent values, α and t_{max}, and determine R based on them for each experiment using (1).

We consider the values $\{1, 1.2, 1.4, \mathbf{1.5}, 1.6, 1.8, 2\}$ for α and $\{\mathbf{10}, 12, 14, 16, 18, 20\}$ for t_{max} and solve the problem for all the combinations of these values. The average improvement of the objective value over baseline for 5 instances is given in Fig. 7. It can be seen that for any fixed α, with increasing t_{max} by two time steps, the objective value is improved by 1% on average. This is equivalent to an increment in R value, keeping α constant, discussed above. Similarly, when α is increased by 0.2 with a fixed value of t_{max}, the objective value is improved by 1% on average. The improvement is expected, since the radius of coverage increases without imposing any additional costs for a longer duration of stay.

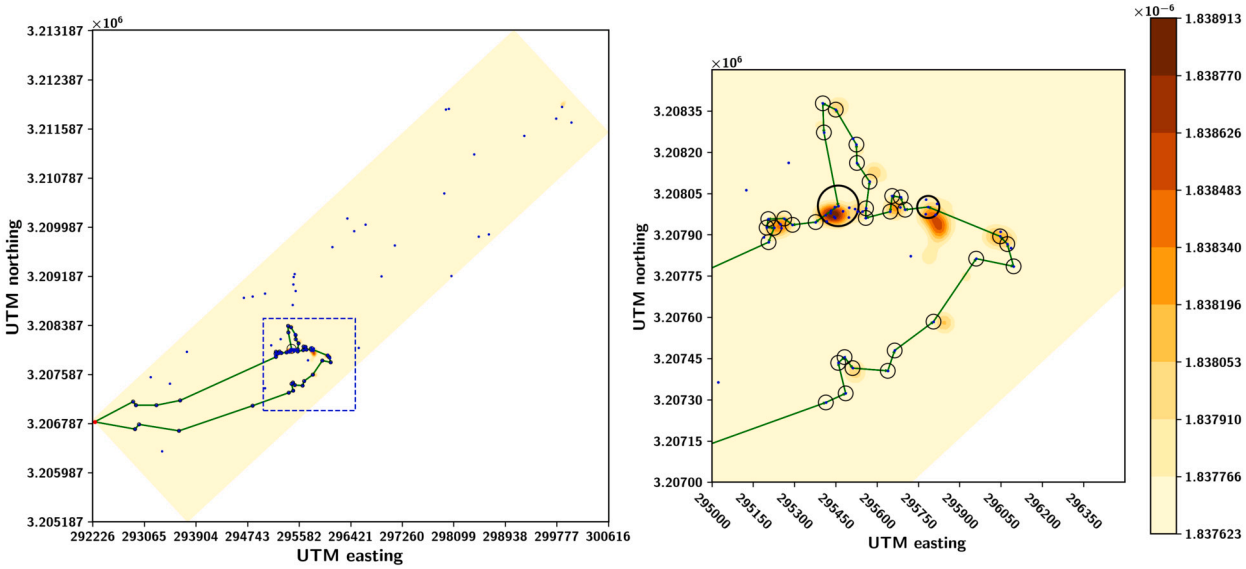


Fig. 6. The solution of an instance with 100 nodes, in the area shown in Fig. 2. Circles with larger line-width indicate a non-zero duration of stay.

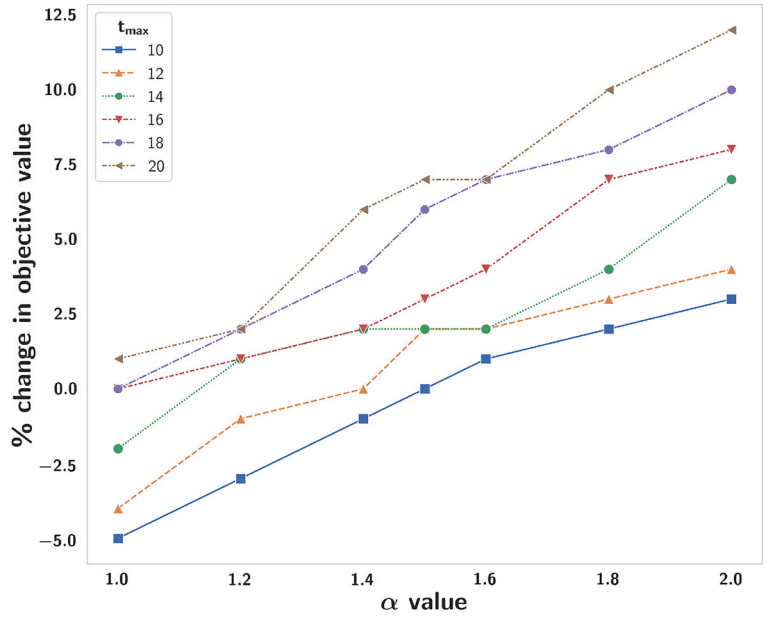


Fig. 7. Average percentage change in the objective function compared to baseline values, when varying α.

Next, we investigate the parameters related to the battery. Here we focus on two parameters, namely the battery capacity of the vehicle, B, and the battery consumption while moving proportioned to when the vehicle stays in a place, $U/\hat{U}$. Different parameter values taken into account are $\{360, 370, 380, 390, \mathbf{400}, 410, 420, 430\}$ for B and $\{1, 1.5, \mathbf{2}, 2.5, 3\}$ for U when keeping $\hat{U} = 1$. In Fig. 8 can be seen that if U is decreased from the baseline value by 0.5 a unit, the solution is independent of the B value which indicates that for these values in the tested instances, the constraint on battery limitation is non-binding. When increasing U and keeping B constant, the objective function clearly declines, since it becomes increasingly more costly, in terms of battery usage, to move among nodes. On average, by increasing U by 0.5 a unit, the objective value declines by 4.6%. Changing the battery capacity of the vehicle, the improvement in the objective value highly relies on the value for the battery usage while moving and is on average 0.62% for 10 units increase in the capacity.

5.2.4. *Implications for monitoring: insights from the study*

Here we demonstrate the improvement that the varying coverage can make on the monitoring, both in terms of cost and coverage. We show the results on instances of size 20 from instance set 1. To this end, we consider CTP-FC, with $R = 0$, i.e. when the AUV

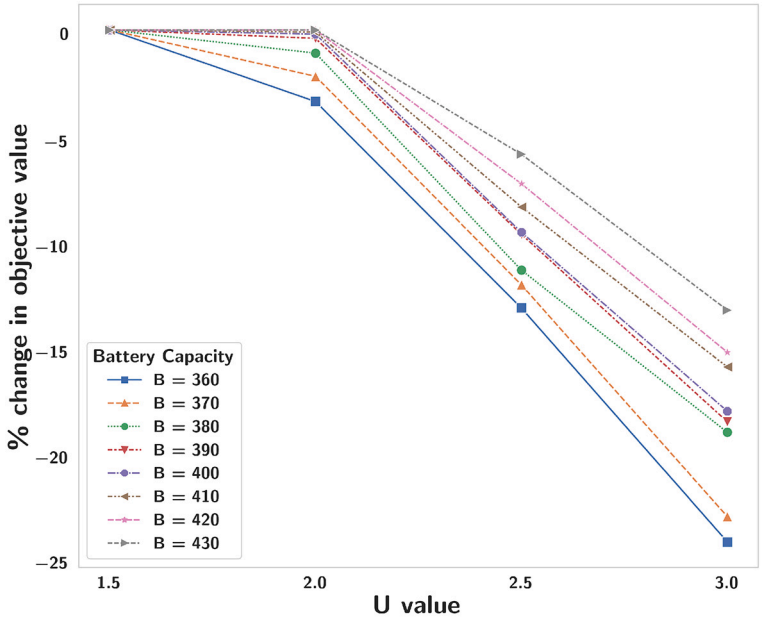


Fig. 8. Average percentage change in the objective function compared to baseline values, when varying U.

Table 7
Average percent improvements of CTP-VC over L-PTP.

Relative improvement	∞-Battery	CTP-VC, $B = B^{\text{PTP}}_{\text{max}}$	L-PTP, $B = B^{\text{VC}}_{\text{max}}$
Objective-value	2.71	2.71	19.78
Cost of moving and staying	9.22	9.22	-7.23
Importance coverage of nodes	0.00	0.00	16.93
Battery consumption	13.92	—	—

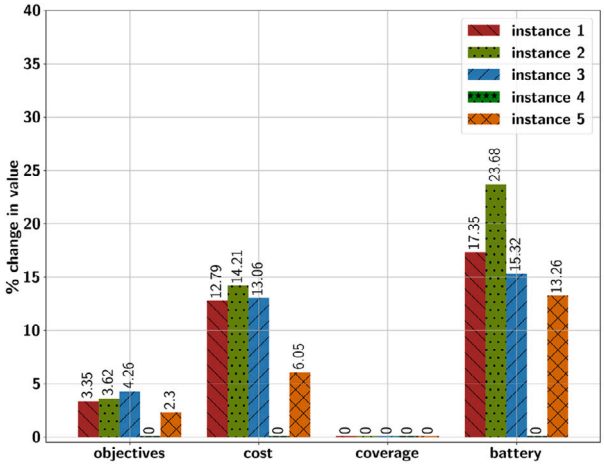
moves to pre-specified points and takes measurements that can only inform the monitoring operators about that specific location. This problem is essentially a Profitable Tour Problem with limited tour length. We call this problem *L-PTP* and compare its performance with CTP-VC.

We make this comparison under different assumptions. First, we remove the battery consumption constraint for both L-PTP and CTP-VC. In this case, the objective value for the CTP-VC solution improves on average by 2.71% compared to the L-PTP solution. The cost of moving and staying decreases by 9.22% and battery consumption by 13.92%. The total importance of covered or visited nodes remains the same for both cases. We notice that although both solutions covered the same monitoring area, the CTP-VC solution saves battery and requires less operational cost. An interesting observation is that both CTP-VC and L-PTP have partial coverage in the best-known solution for three of the tested instances. On average, 99.23% of the total sum of importance was collected. This might indicate an area too far away from the other points of interest, or with negligible importance that can be overlooked for the benefit of the objective. In Fig. 9a we display the improvement for each of the five instances. In Table 7, see the first column, we provide a summary of these, taking the average over instances.

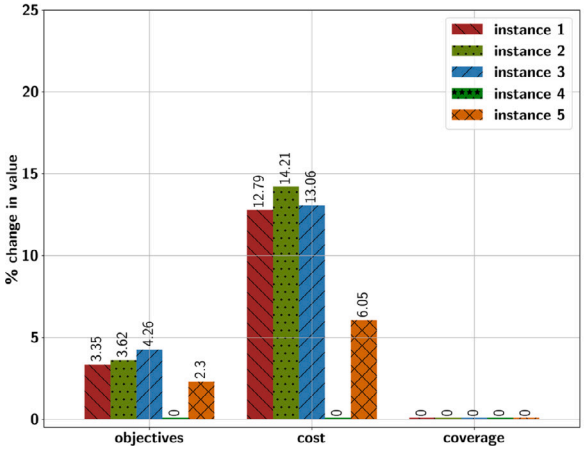
In the setting above, the battery consumed by the solution of CTP-VC is denoted by $B^{\text{VC}}_{\text{max}}$ and the battery consumed by the solution of L-PTP is denoted by $B^{\text{PTP}}_{\text{max}}$. We set the CTP-VC battery limit to $B^{\text{PTP}}_{\text{max}}$. We observe that the improvements are the same as the first column, which is due to the fact that in the previous case, for all instances the battery consumption of CTP-VC was less than that of L-PTP. This makes this constraint non-binding for CTP-VC, and we get the exact same results as the first case. The results of this case are displayed in Fig. 9b. The average improvements of CTP-VC over L-PTP are reported in the second column of Table 7.

Finally, we restrict the battery consumption of L-PTP to $B^{\text{VC}}_{\text{max}}$. We observe that the objective value and the total importance are improved by 19.78%, and 16.93%, respectively. At the same time, the cost of moving and staying is increased by 7.23%, see the third column in Table 7. This is expected as L-PTP now visits fewer nodes and thereby saves on cost. The improvement in the objective value here is more substantial compared to the previous two cases. However, this improvement is gained by covering larger areas rather than saving on costs. For detailed results of this case, see Fig. 9c.

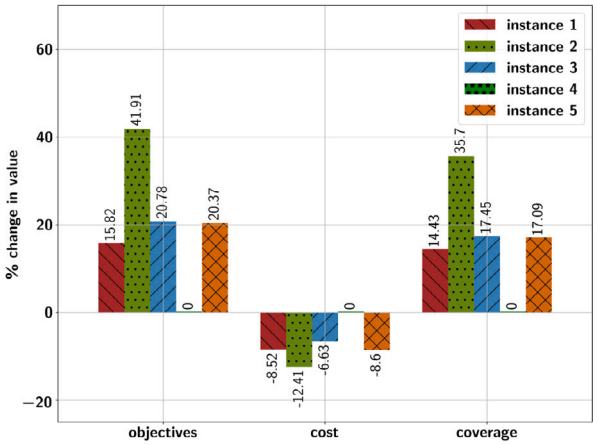
This suggests that in the case the area required to be monitored is large, i.e., an AUV is unable to visit all nodes with the given battery capacity, CTP-VC gives a better monitoring plan than a Profitable Tour Problem.



(a) Both problems solved with unlimited battery capacity.



(b) CTP-VC solved with $B = B^{\text{PTP}}_{\max}$.



(c) L-PTP solved with $B = B^{\text{VC}}_{\max}$.

Fig. 9. Improvement of CTP-VC over L-PTP in various battery capacity scenarios.

6. Conclusion

We have presented a generalized Covering Tour Problem by introducing stay-dependent varying coverage. This was motivated by offshore monitoring, in particular CCS, where varying coverage naturally appears due to the dispersion of pollutants in the ocean environment. Improvement of monitoring and sampling strategies is necessary in order to support CCS as a climate mitigation technology and to ensure the health of the ocean, and the CTP-CV is a step in that direction.

In this paper, we have formulated CTP-VC as an Integer Programming model and conducted numerical experiments. First, we investigated the impact of the varying coverage on a set of small-size instances, for which an exact solution was available in almost all cases. The flexibility provided by the varying coverage resulted in 41.7% improvement in the objective function, on average, compared to the case of fixed coverage. The cost of such an improvement was clearly indicated in the significant increase in the solving time.

Next, we considered large instance sizes for which the exact solver was unable to produce a feasible solution in a given time limit. In order to establish a baseline for such instances, we introduced a constructive heuristic, named LKH & Radius Improvement.

Finally, all the instances were solved using the Adaptive Metaheuristic, the method that we have developed inspired by ALNS [41]. The Adaptive Metaheuristic has shown promising results in terms of both the objective function and the CPU time. In particular, it reaches or surpasses the baseline objective values for all instances and shows robustness to embedded randomness. Moreover, the Adaptive Metaheuristic has shown to be a rather flexible method, meaning that it can be used not only for solving CTP-VC, but also for special cases, i.e. CTP-FC, CSP-VC, CSP-FC, and L-PTP, which we considered in our experiments in the paper.

It is worth mentioning that one set of large instances was based on the data from a CCS storage site in the Gulf of Mexico. For these instances, we modeled the expansion radius using a simplified transport model and the least squares fitting. This modeling has been done assuming isotropic properties of CO_2 dispersion. However, storage sites typically have quite different currents and

naturally occurring CO_2 baseline characteristics. Thus, the CO_2 dispersion, and consequently the varying coverage, could be modeled in a more realistic manner. The isotropic assumption could be lifted in further studies. This would complicate the mathematical formulation, but will not affect the metaheuristic significantly.

We have explored the impacts of parameter change, particularly the parameters related to the varying coverage and battery. The results show that the objective value improves with an increase in the maximum coverage and with the rate of coverage expansion, which is rather expected. In particular, when the rate of change in the coverage is fixed, 1% increment in the maximum covering distance results in 0.05% improvement in the objective function. The corresponding value when increasing the rate of change by 1%, with a fixed maximum duration of stay, is 0.1%. A similar analysis is done on the battery capacity and battery usage while moving. Not surprisingly, it indicates that decreasing battery capacity or increasing the battery usage rate both result in declined objective value. Particularly, an improvement of 0.06% can be seen in the objective value when decreasing the battery usage rate while moving by 1%, with a fixed battery capacity. When the battery capacity is improved by 1%, the objective value has 0.25% improvement.

The implications of varying coverage have been analyzed with respect to monitoring cost and the total area coverage. Most importantly, we have shown that a vehicle uses less battery to cover the same total importance area when the varying coverage is added to the model.

The model and metaheuristic could be developed even further by including multiple AUVs with different characteristics, introducing stationary or moving re-charging stations, or AUVs-carrying ship paths. These developments should however be done in close cooperation with operators and stakeholders. Finally, the applications of CTP-VC are not limited to marine monitoring. The tour design for mobile healthcare delivery systems or disaster relief teams could benefit from incorporating varying coverage in the planning process.

Declaration of competing interest

None of the authors have any financial or non-financial interests that are directly or indirectly related to this work.

Data availability

Data will be made available on request.

Acknowledgements

This work is part of ACTOM, a project that is funded through the ACT program (Accelerating CCS Technologies, Horizon2020 Project No. 294766). Financial contributions made from; The Research Council of Norway, (RCN), Norway, grant number 305202, Ministry of Economic Affairs and Climate Policy, the Netherlands, Department for Business, Energy & Industrial Strategy (BEIS) together with extra funding from NERC and EPSRC research councils, United Kingdom, US-Department of Energy (US-DOE), USA. In-kind contributions from the University of Bergen are gratefully acknowledged. Anna Oleynik is funded through the Academia agreement between Equinor and the University of Bergen.

The authors would like to thank Stefan Carpentier, TNO, for kindly providing the geological risk map, used in this study.

References

[1] E. Domínguez-Tejo, G. Metternicht, E. Johnston, L. Hedge, Marine spatial planning advancing the ecosystem-based approach to coastal zone management: a review, Mar. Policy 72 (2016) 115–130, https://doi.org/10.1016/j.marpol.2016.06.023.

[2] B.S. Halpern, C. Longo, D. Hardy, et al., An index to assess the health and benefits of the global ocean, Nature 488 (2012) 615–620, https://doi.org/10.1038/nature11397.

[3] P. Buhl-Mortensen, M.F.J. Dolan, R.E. Ross, et al., Classification and mapping of benthic biotopes in Arctic and sub-Arctic Norwegian waters, Front. Mar. Sci. 7 (2020), https://www.frontiersin.org/articles/10.3389/fmars.2020.00271. (Accessed 21 November 2022).

[4] L.M. Smith, T.J. Cowles, R.D. Vaillancourt, S. Yelisetti, From the guest editors: introduction to the special issue on the ocean observatories initiative, Oceanography 31 (2018) 12–15.

[5] A.D. Beaton, C.L. Cardwell, R.S. Thomas, et al., Lab-on-chip measurement of nitrate and nitrite for in-situ analysis of natural waters, Environ. Sci. Technol. 46 (2012) 9548–9556.

[6] R.B. Wynn, V.A.I. Huvenne, T.P.L. Bas, et al., Autonomous underwater vehicles (AUVs): their past, present and future contributions to the advancement of marine geoscience, Mar. Geol. 352 (2014) 451–468, https://doi.org/10.1016/j.margeo.2014.03.012.

[7] B. Metz, O. Davidson, H. De Coninck, et al., IPCC Special Report on Carbon Dioxide Capture and Storage, Cambridge University Press, Cambridge, 2005.

[8] E.K. Halland, F. Riis, C. Magnus, et al., CO2 storage atlas of the Norwegian part of the North Sea, Energy Proc. 37 (2013) 4919–4926, https://doi.org/10.1016/j.egypro.2013.06.403.

[9] F. Riis, E. Halland, CO2 storage atlas of the Norwegian continental shelf: methods used to evaluate capacity and maturity of the CO2 storage potential, Energy Proc. 63 (2014) 5258–5265, https://doi.org/10.1016/j.egypro.2014.11.557.

[10] C. Oldenburg, J. Lewicki, On leakage and seepage of CO2 from geologic storage sites into surface water, Environ. Geol. 50 (2006) 691–705.

[11] J. Blackford, J.M. Bull, M. Cevatoglu, et al., Marine baseline and monitoring strategies for carbon dioxide capture and storage (CCS), Int. J. Greenh. Gas Control 38 (2015) 221–229, https://doi.org/10.1016/j.ijggc.2014.10.004.

[12] J. Blackford, Y. Artioli, J. Clark, L. de Mora, Monitoring of offshore geological carbon storage integrity: implications of natural variability in the marine system and the assessment of anomaly detection criteria, Int. J. Greenh. Gas Control 64 (2017) 99–112.

[13] A. Flohr, A. Schaap, E.P. Achterberg, et al., Towards improved monitoring of offshore carbon storage: a real-world field experiment detecting a controlled sub-seafloor CO2 release, Int. J. Greenh. Gas Control 106 (2021) 103237, https://doi.org/10.1016/j.ijggc.2020.103237.

[14] J. Blackford, K. Romanak, V.A.I. Huvenne, et al., Efficient marine environmental characterisation to support monitoring of geological CO2 storage, Int. J. Greenh. Gas Control 109 (2021) 103388, https://doi.org/10.1016/j.ijggc.2021.103388.

[15] T. Dixon, K.D. Romanak, Improving monitoring protocols for CO2 geological storage with technical advances in CO2 attribution monitoring, Int. J. Greenh. Gas Control 41 (2015) 29–40, https://doi.org/10.1016/j.ijggc.2015.05.029.
[16] L. Mabon, S. Shackley, N. Bower-Bir, Perceptions of sub-seabed carbon dioxide storage in Scotland and implications for policy: a qualitative study, Mar. Policy 45 (2014) 9–15.
[17] L. Mabon, S. Shackley, J.C. Blackford, et al., Local perceptions of the QICS experimental offshore CO2 release: results from social science research, Int. J. Greenh. Gas Control 38 (2015) 18–25.
[18] L. Mabon, J. Kita, Z. Xue, Challenges for social impact assessment in coastal regions: a case study of the tomakomai CCS demonstration project, Mar. Policy 83 (2017) 243–251.
[19] C. Jenkins, The state of the art in monitoring and verification: an update five years on, Int. J. Greenh. Gas Control 100 (2020) 103118, https://doi.org/10.1016/j.ijggc.2020.103118.
[20] P.M. Haugan, H. Drange, Sequestration of CO2 in the deep ocean by shallow injection, Nature 357 (1992) 318–320, https://doi.org/10.1038/357318a0.
[21] G. Alendal, H. Drange Two-phase, Near-field modeling of purposefully released CO2 in the ocean, J. Geophys. Res., Oceans 106 (2001) 1085–1096, https://doi.org/10.1029/1999JC000290.
[22] M. Dewar, N. Sellami, B. Chen, Dynamics of rising CO2 bubble plumes in the QICS field experiment, Int. J. Greenh. Gas Control 38 (2015) 52–63, https://doi.org/10.1016/j.ijggc.2014.11.003.
[23] P. Taylor, H. Stahl, M.E. Vardy, et al., A novel sub-seabed CO2 release experiment informing monitoring and impact assessment for geological carbon storage, Int. J. Greenh. Gas Control 38 (2015) 3–17, https://doi.org/10.1016/j.ijggc.2014.09.007.
[24] D.P. Connelly, J.M. Bull, A. Flohr, et al., Assuring the integrity of offshore carbon dioxide storage, Renew. Sustain. Energy Rev. 166 (2022) 112670, https://doi.org/10.1016/j.rser.2022.112670.
[25] A.E.A. Blomberg, I.-K. Waarum, C. Totland, E. Eek, Marine monitoring for offshore geological carbon storage—a review of strategies, technologies and trends, Geosciences 11 (2021), https://doi.org/10.3390/geosciences11090383.
[26] H.K. Hvidevold, G. Alendal, T. Johannessen, et al., Layout of CCS monitoring infrastructure with highest probability of detecting a footprint of a CO2 leak in a varying marine environment, Int. J. Greenh. Gas Control 37 (2015) 274–279, https://doi.org/10.1016/j.ijggc.2015.03.013.
[27] A. Oleynik, M.I. García-Ibáñez, N. Blaser, et al., Optimal sensors placement for detecting CO2 discharges from unknown locations on the seafloor, Int. J. Greenh. Gas Control 95 (2020) 102951, https://doi.org/10.1016/j.ijggc.2019.102951.
[28] G. Alendal, Cost efficient environmental survey paths for detecting continuous tracer discharges, J. Geophys. Res., Oceans 122 (2017) 5458–5467, https://doi.org/10.1002/2016JC012655.
[29] A. Sahoo, S.K. Dwivedy, P. Robi, Advancements in the field of autonomous underwater vehicle, Ocean Eng. 181 (2019) 145–160.
[30] E. Desa, R. Madhan, P. Maurya, et al., The Small Maya AUV–Initial Field Results, 2007.
[31] A. Alvarez, A. Caffaz, A. Caiti, et al., Folaga: a very low-cost autonomous underwater vehicle for coastal oceanography, IFAC Proc. 38 (2005) 31–36.
[32] R. Kimura, M. Choyekh, N. Kato, et al., Guidance and control of an autonomous underwater robot for tracking and monitoring spilled plumes of oil and gas from seabed, in: Twenty-Third Int. Offshore Polar Eng. Conf., OnePetro, 2013.
[33] H. Niu, S. Adams, K. Lee, et al., Applications of autonomous underwater vehicles in offshore petroleum industry environmental effects monitoring, J. Can. Pet. Technol. 48 (2009) 12–16.
[34] D. Kruger, R. Stolkin, A. Blum, J. Briganti, Optimal AUV path planning for extended missions in complex, fast-flowing estuarine environments, in: Proc. 2007 IEEE Int. Conf. Robot. Autom., 2007, pp. 4265–4270.
[35] J. Witt, M. Dunbabin, Go with the Flow: Optimal AUV Path Planning in Coastal Environments, Aust. Conf. Robot. Autom., 2008.
[36] M. Gendreau, G. Laporte, F. Semet, The covering tour problem, Oper. Res. 45 (1997) 568–576.
[37] L. Bach, G. Hasle, C. Schulz, Adaptive large neighborhood search on the graphics processing unit, Eur. J. Oper. Res. 275 (2019) 53–66, https://doi.org/10.1016/j.ejor.2018.11.035.
[38] S. Mirzaei, Optimization Algorithms for Multi–Commodity Routing and Inventory Routing Problems, 2016.
[39] P. Laborie, D. Godard, Self-adapting large neighborhood search: application to single-mode scheduling problems, in: Proc. MISTA-07 Paris 8, 2007.
[40] D. Pisinger, S. Ropke, Large Neighborhood Search, Handb. Metaheuristics, Springer, 2019, pp. 99–127.
[41] D.P. Stefan Ropke, An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows, Transp. Sci. 40 (2006) 455–472, https://doi.org/10.1287/trsc.1050.0135.
[42] E. Arnoff, S. Sengupta, The Traveling Salesman Problem, Prog. Oper. Res., 1961.
[43] K. Ilavarasi, K.S. Joseph, Variants of travelling salesman problem: a survey, in: Int. Conf. Inf. Commun. Embed. Syst. ICICES2014, IEEE, 2014, pp. 1–7.
[44] V. Raman, N.S. Gill, Review of different heuristic algorithms for solving travelling Salesman problem, Int. J. Adv. Res. Comput. Sci. 8 (2017).
[45] D. Feillet, P. Dejax, M. Gendreau, Traveling Salesman problems with profits, Transp. Sci. 39 (2005) 188–205.
[46] H. Ouelmokhtar, Y. Benmoussa, D. Benazzouz, et al., Energy-based USV maritime monitoring using multi-objective evolutionary algorithms, Ocean Eng. 253 (2022) 111182.
[47] H. Ouelmokhtar, Y. Benmoussa, J.-P. Diguet, et al., Near-optimal covering solution for USV coastal monitoring using PAES, J. Intell. Robot. Syst. 106 (2022) 24.
[48] K. Deb, K. Deb, Multi-Objective Optimization, Search Methodol. Introd. Tutor. Optim. Decis. Support Tech., Springer, 2013, pp. 403–449.
[49] M. Dell'Amico, F. Maffioli, P. Värbrand, On prize-collecting tours and the asymmetric travelling Salesman problem, Int. Trans. Oper. Res. 2 (1995) 297–308.
[50] J.R. Current, D.A. Schilling, The covering Salesman problem, Transp. Sci. 23 (1989) 208–213.
[51] B. Golden, Z. Naji-Azimi, S. Raghavan, et al., The generalized covering salesman problem, Inf. J. Comput. 24 (2012) 534–553.
[52] M.H. Shaelaie, M. Salari, Z. Naji-Azimi, The generalized covering traveling Salesman problem, Appl. Soft Comput. 24 (2014) 867–878.
[53] G. Ozbaygin, H. Yaman, O.E. Karasan, Time constrained maximal covering Salesman problem with weighted demands and partial coverage, Comput. Oper. Res. 76 (2016) 226–237.
[54] R. Bowerman, B. Hall, P. Calamai, A multi-objective optimization approach to urban school bus routing: formulation and solution method, Transp. Res. Part Policy Pract. 29 (1995) 107–123.
[55] M. Ma, Y. Yang, Data gathering in wireless sensor networks with mobile collectors, in: 2008 IEEE Int. Symp. Parallel Distrib. Process., IEEE, 2008, pp. 1–9.
[56] S.P. Tripathy, A. Biswas, T. Pal, A multi-objective covering Salesman problem with 2-coverage, Appl. Soft Comput. 113 (2021) 108024.
[57] M. Salari, M. Reihaneh, M.S. Sabbagh, Combining ant colony optimization algorithm and dynamic programming technique for solving the covering Salesman problem, Comput. Ind. Eng. 83 (2015) 244–251, https://doi.org/10.1016/j.cie.2015.02.019.
[58] K. Li, T. Zhang, R.W.Y. Wang, Y. Han, Deep reinforcement learning for combinatorial optimization: covering Salesman problems, arXiv:2102.05875 [cs.NE], 2021.
[59] Y. Lu, U. Benlic, Q. Wu, A highly effective hybrid evolutionary algorithm for the covering Salesman problem, Inf. Sci. 564 (2021) 144–162.
[60] L.P. Maziero, F.L. Usberti, C. Cavellucci, Branch-and-cut algorithms for the covering Salesman problem, arXiv:2104.01173 [cs.DS], 2021.
[61] P.J. Rousseeuw, Silhouettes: a graphical aid to the interpretation and validation of cluster analysis, J. Comput. Appl. Math. 20 (1987) 53–65.
[62] P. Shaw, A New Local Search Algorithm Providing High Quality Solutions to Vehicle Routing Problems, APES Group Dept Comput. Sci. Univ., Strathclyde Glasg. Scotl. UK, 1997, p. 46.
[63] P. Shaw, Using constraint programming and local search methods to solve vehicle routing problems, in: Int. Conf. Princ. Pract. Constraint Program., Springer, 1998, pp. 417–431.

[64] D.E. Goldberg, Genetic Algorithms, Pearson Education India, 2013.
[65] S. Kirkpatrick, C.D. Gelatt Jr, M.P. Vecchi, Optimization by simulated annealing, Science 220 (1983) 671–680.
[66] Y. Crama, M. Schyns, Simulated annealing for complex portfolio selection problems, Eur. J. Oper. Res. 150 (2003) 546–571.
[67] M. Dewar, G. Alendal, J. Blackford, et al., A semi-automated toolbox to aid operations in the design of efficient environmental offshore monitoring programs for CO2 storage sites, in: Proc. 16th Greenh. Gas Control Technol. Conf., 2022.
[68] G. Alendal, S.E. Gasda, K. Romanak, et al., The impact of pre-project data quality and quantity on developing environmental monitoring strategies for offshore carbon storage: case studies from the Gulf of Mexico and the North Sea, in: Greenh. Gas Control Technol. Conf., 2022.
[69] X. Zhang, M. Marta-Almeida, R.D. Hetland, A high-resolution pre-operational forecast model of circulation on the Texas-Louisiana continental shelf and slope, J. Oper. Oceanogr. 5 (2012) 19–34, https://doi.org/10.1080/1755876X.2012.11020129.
[70] Z.A. Wang, R. Wanninkhof, W.-J. Cai, et al., The marine inorganic carbon system along the Gulf of Mexico and Atlantic coasts of the United States: insights from a transregional coastal carbon study, Limnol. Oceanogr. 58 (2013) 325–342, https://doi.org/10.4319/lo.2013.58.1.0325.
[71] S. Lin, B.W. Kernighan, An effective heuristic algorithm for the traveling-Salesman problem, Oper. Res. 21 (1973) 498–516.
[72] K. Helsgaun, An effective implementation of the Lin–Kernighan traveling Salesman heuristic, Eur. J. Oper. Res. 126 (2000) 106–130.