Safety Science 205 (2027) 107406

Contents lists available at ScienceDirect

Safety Science

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

Maritime search and rescue operations planning: A hybrid approach for multi-asset deconfliction and efficient resource allocation ☆

Amirhossein Esmaeilpour *, Michael Morin, Irène Abi-Zeid

Department of Operations and Decision Systems, Faculty of Business Administration, Université Laval, 2325, rue de la Terrasse, Québec, QC, G1V 0A6, Canada

ARTICLE INFO

Keywords:
Search operations planning
Emergency response
Maritime search and rescue
Blackbox optimization and simulation

ABSTRACT

Decision support systems for maritime search and rescue search planning, such as those available in Canada and in the US, assist search mission coordinators in generating search plans for missing objects. However, because these systems rely on simulations to evaluate a plan's probability of success, they cannot be directly integrated into traditional mathematical programming frameworks. Furthermore, existing tools, such as SAR Optimizer, the planning module within the Canadian Advanced Search Planning Tool (CANSARP), lack the mechanisms to enforce some operational constraints and do not optimize the number of search resources and search durations. To address these limitations, we propose a hybrid optimization framework that treats the search simulator as an external blackbox objective function calculator. Within this framework, we develop models that enforce horizontal spatial separation between search units, an important operational safety constraint. Moreover, we introduce search duration as a decision variable, allowing for search plans that consume fewer search units or reduce search times. Numerical experiments on realistic scenarios demonstrate that this hybrid integration of simulation and optimization not only enhances safety but also significantly improves efficiency, potentially saving hundreds of nautical miles in flight time.

1. Introduction

The act of searching is an important part of many humanitarian operations, such as search and rescue (SAR), mine countermeasures, and of many surveillance operations for the purpose of protecting individuals, the environment, resources, or infrastructure.

As outlined in its mission statement, the Canadian Coast Guard (CCG) SAR program aims to achieve a 100 percent success rate in saving lives during maritime emergencies (Canadian Coast Guard, 2023). Now integrated within the Canadian Department of National Defence, the CCG directs and coordinates maritime SAR operations in Joint Rescue Coordination Centres and Maritime Subcentres (Canadian Coast Guard, 2023; National Defence and Fisheries and Oceans Canada, 2014). These centers are responsible for over 5.3 million square kilometers and are staffed by SAR mission coordinators who operate 24/7 (Fisheries and Oceans Canada, 2017). Managing such a vast operational area is inherently complex, as it encompasses a diverse range of maritime activities that each carry unique risk profiles (Cucinelli et al., 2023). While the overwhelming majority of day-to-day incidents are resolved quickly and routine assistance is common, formal search operations still arise once or twice every month, typically lasting between one and two days (Abi-Zeid, 2026a), with major operations such as the Swiss Air 111 catastrophe in 1998, which can last much longer. But how and where to search when time is of the essence? The answer lies in effective search planning that ensures the best use of scarce search resources, such as aircraft or vessels, to safely locate and aid those in distress and prevent injury or loss of life (Abi-Zeid et al., 2019).

The need for decision support systems to assist in the planning of search operations has been recognized in the scientific literature for many years (Abi-Zeid and Frost, 2005; Kratzke et al., 2010; Abi-Zeid et al., 2019). In 2014, the CCG stated a requirement for an optimization module to assist SAR mission coordinators in generating, evaluating, and selecting the best search plans for maritime searches (Abi-Zeid et al., 2019). Abi-Zeid et al. (2019) addressed this requirement by developing SAR Optimizer, a search plan optimization and evaluation software initially called Search

☆ This article is part of a Special issue entitled: 'Crisis and Emergency Management' published in Safety Science.

* Corresponding author.
E-mail addresses: amirhossein.esmaeilpour.1@ulaval.ca (A. Esmaeilpour), michael.morin@osd.ulaval.ca (M. Morin), irene.abi-zeid@osd.ulaval.ca (I. Abi-Zeid).

https://doi.org/10.1016/j.ssci.2026.107406
Received 15 February 2026; Received in revised form 23 July 2026; Accepted 3 August 2026
Available online 3 September 2026

Planner. SAR Optimizer was integrated, in 2020, into the CCG's Advanced Search Planning Tool (ASPT), a decision support system also known as CANSARP. SAR Optimizer assesses the quality of candidate search plans by simulating the trajectories of the search units[1] in relation to possible trajectories of a drifting search object. A key limitation of this simulation-based approach is that it does not easily allow for direct mathematical programming. Because the objective function, namely the probability of success, is estimated through simulation, the problem cannot be directly formulated and solved as a conventional mathematical program (also called an optimization model). This absence of an analytical form for the objective function is referred to as blackbox optimization (Audet and Hare, 2017).

Although simulation-based and blackbox optimization methods are well-established in the literature (Fu et al., 2015; Alarie et al., 2021), current operational maritime SAR decision support systems rely on problem-specific heuristics (e.g., (Kratzke et al., 2010; Abi-Zeid et al., 2019)). These heuristics are imperative by nature; they dictate *how* to solve a problem step-by-step rather than defining *what* an optimal solution should look like. In contrast, mathematical programming offers a declarative approach. Under this framework, a mathematical *model* defines the problem's constraints and objectives (the *what*), while a separate, general-purpose solver handles the optimization process (the *how*). A declarative *model-and-solve* approach to optimization offers significant advantages. For instance, it allows developers to benchmark and systematically evaluate various commercial and open source general-purpose solvers. These solvers are actively updated and can often surpass problem-specific heuristics. Solvers also facilitate hybrid strategies that combine exact and heuristic elements, often leading to better algorithms and can provide transparent and reproducible baselines. In addition, a model-and-solve approach to optimization makes it easier to address several limitations in the current system.

To illustrate these limitations, we highlight the three following operational gaps in SAR Optimizer, identified during consultations between the CCG team and the third author, who remains active in CANSARP's operational development and the ASPT working group: First, while altitude-based deconfliction is supported in SAR Optimizer, partitioning the search environment into safe, exclusive horizontal operating zones is a manual process when multiple search units operate simultaneously at the same altitude. Second, the number of search units and their available search effort are currently treated as fixed inputs rather than variables to be optimized. This can lead to an overutilization of resources that only marginally increases the probability of success of an operation. Third, the search units' trajectory angle variables are currently fixed a priori whereas rendering them variable can lead to better search plans.

With these gaps in mind, we formulate the following research question: How can we design mathematical models for the maritime search planning problem that enforce operational constraints, while using simulation to compute the objective function?

To address this question, this work provides three main contributions:

1. We propose a hybrid model-and-solve framework that integrates mathematical programming with simulation;
2. We develop five optimization models, each implementing a mathematical program that addresses one or more of the above identified limitations as explicit constraints;
3. We evaluate our approach on 18 realistic Canadian SAR cases generated by an experienced SAR mission coordinator.

While this research focuses on enhancing SAR Optimizer in CANSARP, the proposed framework can be adapted to other maritime SAR decision support systems.

The rest of the paper is structured as follows. In Section 2, we provide an overview of the relevant literature on maritime SAR and the necessary background. In Section 3, we present the problem formulation and the solution approach, including the details of the proposed optimization models. In Section 4, we compare the results of the numerical experiments conducted with the different models. The results are discussed in Section 5. We describe the limitations of our work in Section 6, and we conclude in Section 7.

2. Background and related work

Maritime SAR operations in joint rescue coordination centers can be modeled as an iterative decision-making process. We represent the workflow using the OODA loop (observe, orient, decide, act), a framework well-known in command and control operations (Fig. 1) (Abi-Zeid and Lamontagne, 2003).

The process starts with the *observe* phase with the notification and initial observation of the incident. It continues with the *orient* phase, where the available information is assessed and analyzed to build situational awareness. Then, the *decide* phase covers search planning, where the available information is used to choose a search plan. Finally, the *act* phase involves conducting the SAR operation, monitoring progress, and reducing the search. Note that not all incidents require undergoing the full process. In this work, we focus specifically on the search planning step within the *decide* phase, where we apply concepts from search theory.

2.1. Search theory concepts

Search theory is a specialized branch of operations research that provides a mathematical foundation for the allocation of search resources to locate lost, hidden, or evading search objects (Stone et al., 2016). The position of the search object is characterized by uncertainty, which is formally encoded as a *probability of containment*, a spatial probability distribution across a search environment abstracted as a discrete grid of cells. One critical distinction between stationary and moving objects is that while overland searches for wreckage may treat the probability of containment as static, maritime search objects are subject to drift due to wind and surface currents. As a result, the probability that a given search area contains the search object evolves over time. In classical search theory, the probability of containment of a search object is updated after an unsuccessful search through a Bayesian scheme (Stone, 1983).

Searching actions are modeled as a *search effort allocation* over the search environment. The effort can be characterized as either discrete or continuous. Discrete effort typically refers to a countable number of distinct observations or scans, whereas continuous effort represents a divisible resource, such as the total time or track length available for searching. Search effort is often allocated in a constrained fashion. An example of constrained search effort allocation occurs in optimal searcher path problems where the location of effort at a later time is constrained by the location of past effort (Trummel and Weisinger, 1986). Another example of constraints on the effort can be found in the multiple rectangular

[1] We use *search unit* to designate the *searcher* which is, in the context of maritime SAR operations, an aircraft or a vessel. We employ the term *search unit* instead of *search and rescue unit* to simplify the text, although some search units have rescue capabilities.

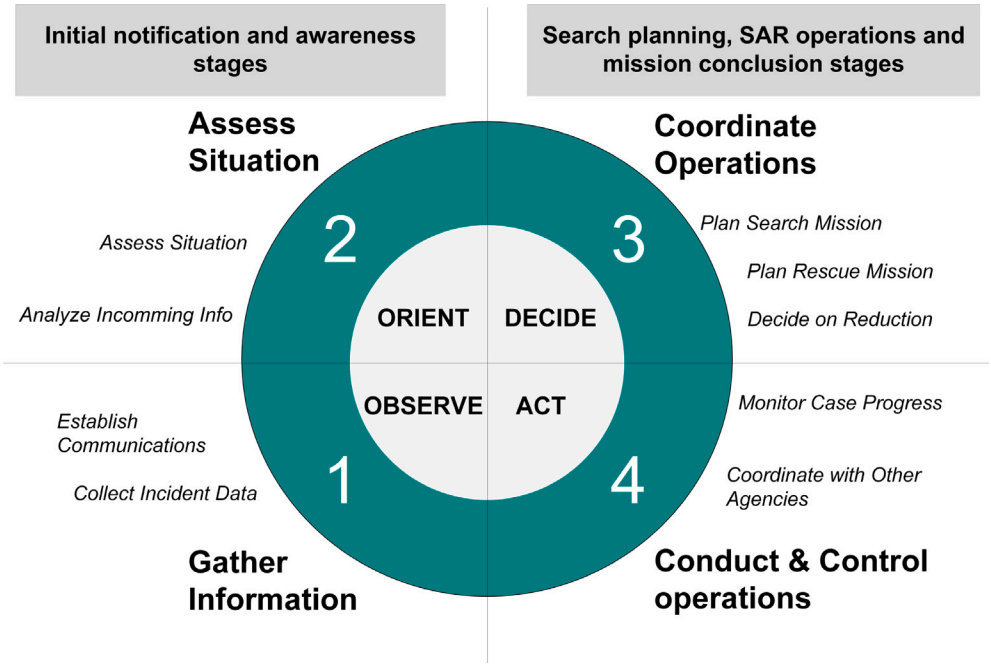

Fig. 1. Maritime SAR OODA loop (observe, orient, decide, act) (Abi-Zeid and Lamontagne, 2003).

search areas problem (MRSA) (Discenza, 1978). In the MRSA, each search unit has a total amount of effort, such as time, that needs to be allocated to a rectangular zone named the *search rectangle*. The MRSA problem is a high-level effort allocation problem, meaning that the trajectory of a search unit inside a rectangle follows an operational search pattern of a specific type, e.g., a parallel tracks pattern where the distance between two search legs is called track spacing (National Defence and Fisheries and Oceans Canada, 2014). The MRSA was applied, for instance, in the Canadian overland SARPlan decision support system designed for stationary objects (Abi-Zeid and Frost, 2005).

The performance of a search is defined by the *probability of detection*, the conditional probability that the search unit will detect the search object, given that the object is located within the unit's search rectangle. It is a function of the search effort expended by the search unit. Central to this calculation is the lateral range curve, $f_i(d)$, which represents the probability of detecting an object as a function of the distance at the closest point of approach (d) between the object and the search unit i along its search track. The lateral range curve depends on the object's type, environmental conditions, the search unit's type and characteristics such as the altitude, sensors, speed or the crew fatigue, and is estimated during extensive experiments in different contexts for different search objects (Stone et al., 2016).

A common figure of merit for evaluating the quality of a *search plan*, i.e., the assignment of rectangular search areas to search units, is its *probability of success*, roughly defined as the probability of finding the search object (Abi-Zeid et al., 2019). It is the product of the probability of containment and the conditional probability of detection.

2.2. Simulation-based maritime search and rescue planning

The United States Coast Guard Computer-Assisted Search Planning system (CASP), and later systems such as SAROPS, its successor, and SAR Optimizer in Canada (Richardson and Discenza, 1980; Kratzke et al., 2010; Abi-Zeid et al., 2019) use Monte Carlo simulations, which represent the search object as thousands of discrete *particles*. A plausible trajectory for each particle is predicted from real-time environmental variables, such as shifting wind and ocean currents (Kratzke et al., 2010). These simulated drifts are then used as an input to compute the probability of success. The simulation approach provides a continuous-space representation of the search object's likely drift and allows for a more precise positioning of search rectangles (Breivik and Allen, 2008).

In order to propose search rectangles, SAR Optimizer starts by generating a search environment based on the convex hull of the particles' positions during the on-scene time of the search assets. Heuristics are then applied to generate candidate search plans within this search environment (Abi-Zeid et al., 2019). These candidates are then evaluated through simulation: The search units' search patterns are simulated, and the probability of detecting a particle by a search unit is computed at the closest point of approach between a particle and a search leg, based on the lateral range function, as described below.

Consider a set P of particles. Let $O(p)$ represent the initial probability of particle $p \in P$ being the search object, i.e., 1 on the total number of particles. Let $pfail(p)$ indicate the probability of particle p being undetected. Before any search has been conducted, $pfail(p)$ is set to 1. The detection probability $f_i(p, d_l)$ is determined by the lateral range function based on the distance d_l at the closest point of approach of the l^{th} search leg of search unit i to particle p. The highest such value is retained for a search leg. The probability of search unit i failing to detect particle p on the l^{th} leg of its search pattern is expressed as $1 - f_i(p, d_l)$. Assuming independence of detection events across different infinitely long search legs (Abi-Zeid et al., 2019), the probability of search unit i not detecting particle p by crossing the n search legs of its search pattern can be formulated as:

$$pfail(p, i) = \prod_{l=1}^{n} (1 - f_i(p, d_l)) . \quad (1)$$

In SAR Optimizer, the detection calculation depends on the characteristics of the search unit, the search object, and environmental conditions. Different search units searching for the same search objects can therefore have different probabilities of detection and different values of $pfail(p, i)$. Suppose there are k search units involved in a maritime SAR operation. The probability of a particle p not being detected can be expressed as:

$$pfail(p) = \prod_{i=1}^{k} pfail(p, i)\,, \tag{2}$$

where $pfail(p, i)$ is the probability of particle p not being detected by the i^{th} search unit. The probability of a particle p being detected is then given by:

$$POD(p) = 1 - pfail(p)\,. \tag{3}$$

Finally, the probability of success of a search plan S for the k search units is defined as:

$$POS(S) = \sum_{p \in P} O(p) POD(p)\,. \tag{4}$$

Given a fixed set of particles, the search simulation performed in SAR Optimizer to evaluate a search plan is deterministic. That is, the same probability of success is calculated each time the same search plan is evaluated on the same set of particles.

2.3. Recent maritime SAR literature

Peer-reviewed journal articles published over the last decade on maritime SAR generally fall into two categories: strategic preparedness and operational response. Within this body of literature, a distinct subset of studies addresses the strategic elements before an incident occurs. This includes location–allocation studies that focus on the positioning of SAR assets (Karatas, 2021; Wang et al., 2025), capability assessment studies that evaluate the readiness of the SAR system (Zhou et al., 2020; Fernandes et al., 2026), and risk governance and human reliability studies that focus on system-level factors that affect SAR response (Cucinelli et al., 2023; Sezer and Akyuz, 2025).

The remaining literature focuses on operational and tactical responses, mapping directly onto the phases of the OODA loop Fig. 1. Once an incident is reported, the available information must be assessed. The work of Nordström et al. (2016) is an example of research on information assessment. They proposed a distress communication method that helps SAR operators and vessel crews assess and communicate the safety status of a distressed vessel (Orient phase).

Following orientation, SAR mission coordinators must transition from understanding the situation to allocating assets (Coordinate). A key input to the planning task is the identification of the search units to be used. Malyszko (2021) studied this resource selection problem by combining a multi-criteria decision analysis method with fuzzy logic to rank vessels according to their suitability for a SAR operation. Moving from the choice of resources to setting their number, Guo et al. (2019) used multi-objective optimization to decide how many aircraft and vessels to allocate and in what combinations.

Search planning in the Coordinate phase starts when the drift information and available assets are already known. At that point, the question shifts from "how to collect the incident data and assess the incident?" to "how to turn the available information into a search plan?" Search plans can be categorized by their level of abstraction. Low-level plans are highly specific, such as defining a precise path for a search unit (path planning), whereas high-level plans outline broader parameters, such as a rectangular area with an assumed search pattern type, like in the MRSA problem (effort allocation).

Wu et al. (2024) provide an example of a low-level planning problem. Given a search environment to cover, they compare paths obtained for multiple search units with reinforcement learning to other search pattern types. Their problem is framed such as reinforcement learning is used to reduce the number of search pattern steps needed to increase the probability of success rate of accumulation compared to other pattern types, such as parallel track patterns. Zhang et al. (2025) formulated a mixed-integer linear programming model for unmanned aerial vehicles path planning. The model generates search paths by maximizing the probability of success and penalizing repeated path selection in the objective function. It also enforces unmanned aerial vehicles path deconfliction.

Our approach aligns more closely with high-level effort allocation, where search patterns follow a predetermined type. For instance, Jeong et al. (2026) studied a search rectangle position problem based on heatmaps computed from simulated drifts. Their optimization framework uses a genetic algorithm for search unit deployment to search rectangles. It dynamically updates the search plan based on the relative motion between search units and drifting particles. The model does not explicitly consider rectangle deconfliction, flexible effort, nor the option to leave a search unit untasked. To support search planning in SAR Optimizer, Laperrière-Robillard et al. (2022) used machine learning to approximate the probability of success, computed otherwise through costly simulations. Furthermore, it provided SAR Optimizer with a heuristic to prioritize and simulate candidate solutions for a single search unit based on the highest predicted success. Parallel to these simulation-focused approaches, alternative methods prioritize iterative grid-based updates to account for motion. Xiong et al. (2020) proposed a time domain-based iterative method that accounts for the relative motion between search objects and search units to determine the search areas. The method creates a probability map for the search object and updates it iteratively. The search area is then determined by a procedure that starts from the cell with the highest probability of containing the search object. An agent-based simulation is used at the end of the optimization process to evaluate the search plan. Building further on this iterative approach, Xiong et al. (2021) proposed an iterative search area adjustment method based on minimum bounding rectangle, K-means clustering, and probability maps. The method generates search areas for a helicopter from probable drift trajectories fixed at the time of search initiation. These trajectories are used to build a static probability map, where each grid cell contains a probability of containment.

Once the search objects are found, the focus shifts to rescue missions. Zhao et al. (2024) developed a two-stage scheduling model for mass rescue operations, especially for cruise ship accidents. The model dispatches multiple rescue resources while minimizing both rescue time and the number of dispatched resources. Krajewska (2021) presents a real rescue operation in the Baltic Sea and compares actual Polish SAR actions with the recommendations of the IAMSAR manual (International Maritime Organization and International Civil Aviation Organization, 2019).

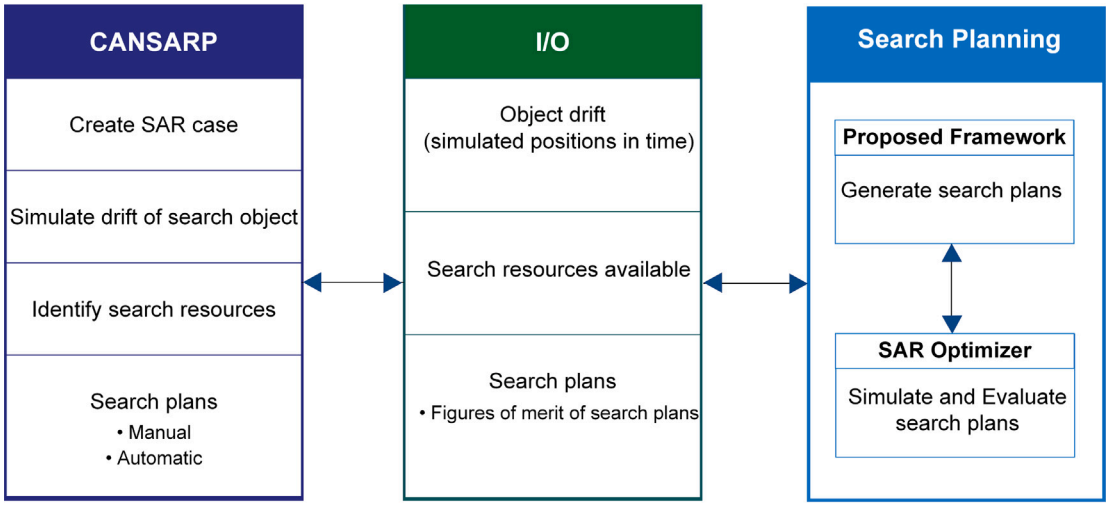

Fig. 2. Adapted CANSARP/SAR Optimizer logic when using the model-and-solve approach; the figure is based on Abi-Zeid et al. (2019).

2.4. Summary

In a maritime SAR context, multiple search assets, such as helicopters, fixed-wing aircraft, and vessels, are often deployed simultaneously. When these assets overlap in time in the same area, the SAR mission coordinator must manually manage altitude deconfliction. This asset mix is essential in search planning because fixed-wing aircraft cannot perform rescues; they can only locate the search object, requiring a nearby helicopter or vessel to complete the rescue. Given that Canadian operations can deploy up to six air assets and ten vessels at a time, there is a clear operational demand for a search planning model that integrates horizontal spatial deconfliction for safety considerations and handles variable effort allocation, thereby ensuring the most efficient use of resources.

Our review of the literature reveals, to the best of our knowledge, that the maritime SAR literature completely lacks a simulation–optimization blackbox framework. Furthermore, current optimization models fail to incorporate horizontal spatial deconfliction for aircraft, variable search angles, and the flexibility to reduce effort or leave a search unit untasked. This gap, identified jointly with the CCG, validates the focus of our research question and motivates our proposed framework.

3. A model-and-solve approach to optimizing maritime search planning

To answer our research question, we developed models that provide a search plan for k search units. This plan consists of at most k search rectangles R_i (for $i \in \{1, \ldots, k\}$) configured to maximize the total probability of success. When computing the probability of success using simulation, a search rectangle i induces a parallel search pattern for the i^{th} search unit. The length of this pattern (in nautical miles) is calculated from the amount of effort (in hours) made available to the corresponding search unit and to its speed (in knots). This length, in turn, dictates the number of search legs and the corresponding track spacing between them. Track spacing is constrained, for maneuverability reasons, by bounds that depend on the search unit's type. Furthermore, for a fixed amount of effort, it is directly proportional to the rectangle's area: Increasing or decreasing the rectangle's area increases or decreases the track spacing. In addition, our model is designed such that when multiple search units operate with overlapping on-scene time and at the same altitude, it enforces horizontal spatial deconfliction. This is achieved by treating the search rectangles as exclusive zones, i.e., the optimization algorithm ensures that the rectangles R_i and R_j of search units i and j do not overlap in space and time.

Fig. 2 depicts the maritime SAR planning process using CANSARP along with our proposed search planning model. Our approach fits in the search planning part. In the original optimizer presented in Abi-Zeid et al. (2019), a heuristic generates candidate search plans and passes them to the simulator and evaluator modules. In our approach, the heuristic is replaced by a pair of a model and a solver; the model describes the problem (possibly using blackbox external functions) and the solver generates the search rectangles. The generated search plans are then evaluated by calling the simulator and evaluator modules of SAR Optimizer, which are external to the solver.

3.1. Optimization models definition

We now describe the five mathematical models built to solve the problem of allocating k search units to k search rectangles. The models are presented from the simplest one to the most complete:

- `M` is a model solving the problem of allocating the k search units to k rectangles with fixed effort and angle. The objective function value is computed by SAR Optimizer. The constraints enforce rectangle geometry, horizontal spatial deconfliction, and operational constraints on track spacing. The decision variables control rectangle positioning and rectangle parameters.
- `MC` is an extension of model `M` with an additional constraint on particle coverage, i.e., minimum number of particles inside the generated rectangles. This particle coverage relates to particle density and is different from the geometric definition of coverage factor from search theory.
- `ME` is an extension of model `M` with effort as decision variables. The additional decision variables are combined with suitable constraints to add the option to untask one or more search units.
- `MEC` is an extension of model `M` with effort as decision variables and additional constraints on particle coverage.

Table 1
Variables for a search unit i, used in the models.

Decision variables	
LLX_i, LLY_i	The coordinate of the lower-left corner of the rectangle, bounded by the search environment with tolerance b, $LLX_i \in [x\_min_i - b, x\_max_i + b]$, $LLY_i \in [y\_min_i - b, y\_max_i + b]$.
$WIDTH_i$, $HEIGHT_i$	The rectangle's width and height, in nautical miles, $WIDTH_i \in [0, max\_width_i]$ and $HEIGHT_i \in [0, max\_height_i]$.
$ANGLE_i$	The orientation angle of the rectangle at the time on-scene in degrees, $ANGLE_i \in [0, 359]$.
TS_i	The track spacing in nautical miles, $TS_i \in [ts\_min_i, ts\_max_i]$.
N_i	The number of search legs, $N_i \in \{1, \ldots, 100000\}$.
T_i	The available on-scene time of the search unit in hours, $T_i \in [t\_min_i, t\_max_i]$.
Intermediate variables	
LRX_i, LRY_i	The coordinate of the lower-right corner of the rectangle.
ULX_i, ULY_i	The coordinate of the upper-left corner of the rectangle.
URX_i, URY_i	The coordinate of the upper-right corner of the rectangle.
θ_i	The angle in radians.

Table 2
Parameters for a search unit i, used in the models.

Case parameters	
$speed_i$	The search speed of the search unit in knots.
ts_min_i, ts_max_i	The lower and upper bounds for the track spacing of the search unit in nautical miles.
t_min_i, t_max_i	The minimum and maximum on-scene times allowed in hours.
$altitude_i$	The altitude at which the search unit operates, measured in feet, used in the ext_no_intersect external function that enforces deconfliction constraints.
$deconfliction_{ij}$	The preprocessed overlapping matrix of the on-scene times. $deconfliction_{ij} = 1$ if search units i and j overlap in time and operate at the same altitude, otherwise $deconfliction_{ij} = 0$.
Modeling-related parameters	
t_act	The activation threshold: a search unit is considered enabled if $T_i \geq t\_act$.
x_min_i, x_max_i	Minimum and maximum X coordinates of the search environment.
y_min_i, y_max_i	Minimum and maximum Y coordinates of the search environment.
max_height_i	Maximum height of the rectangle corresponding to the height of the search environment.
max_width_i	Maximum width of the rectangle corresponding to the width of the search environment.
b	A buffer parameter that relaxes the bounds of the maximum environment rectangle in nautical miles.
α	A unitless aspect ratio coefficient used in constraints (10) and (11).
φ_min	The minimum particle coverage density threshold as a percentage (%), used in constraint (17), (24), and (28).
ϵ	The tolerance used in effort constraints (13), (14), (20), and (21), in nautical miles.
$angle_i$	The orientation angle of the rectangle at the time on-scene in degrees, used as a parameter when the variable $ANGLE_i$ is not used.

- MECA is an extension of model M with effort and angle as decision variables and with additional particle coverage constraints. Adding the angle as a decision variable gives the solver the option to choose the angle of a search rectangle.

The models use logical constraints as well as blackbox external functions in their formulation. Table 1 summarizes variables used in all models along with their domains. Table 2 summarizes the parameters. Parameter names are in *snake_case*; variables are in *UPPERCASE*. The external function names start with ext_ in sans serif. X and Y coordinates correspond to longitude and latitude, respectively.

3.1.1. Model M

Model M is defined by objective function (5) and constraints (6)–(14).

$$\max \quad POS = \text{ext\_SAR}(\{LLX_i, LLY_i, WIDTH_i, HEIGHT_i, TS_i, N_i\}_{i=1}^{k}) \tag{5}$$

$$\text{s.t.} \quad (LRX_i, LRY_i) = (LLX_i, LLY_i) + WIDTH_i(\cos\theta_i, \sin\theta_i), \qquad \forall i \in \{1, \ldots, k\}. \tag{6}$$

$$(ULX_i, ULY_i) = (LLX_i, LLY_i) + HEIGHT_i(-\sin\theta_i, \cos\theta_i), \qquad \forall i \in \{1, \ldots, k\}. \tag{7}$$

$$(URX_i, URY_i) = (LRX_i, LRY_i) + HEIGHT_i(-\sin\theta_i, \cos\theta_i), \qquad \forall i \in \{1, \ldots, k\}. \tag{8}$$

$$x\_min_i \leq X \leq x\_max_i \ \wedge \ y\_min_i \leq Y \leq y\_max_i,$$
$$\forall (X, Y) \in \{(LRX_i, LRY_i), (ULX_i, ULY_i), (URX_i, URY_i)\}, \qquad \forall i \in \{1, \ldots, k\}. \tag{9}$$

$$WIDTH_i \geq \alpha \cdot HEIGHT_i, \qquad \forall i \in \{1, \ldots, k\}. \tag{10}$$

$$HEIGHT_i \geq \alpha \cdot WIDTH_i, \qquad \forall i \in \{1, \ldots, k\}. \tag{11}$$

$$\text{ext\_no\_intersect}(\{LLX_i, LLY_i, WIDTH_i, HEIGHT_i, TS_i, N_i\}_{i=1}^{k}). \tag{12}$$

$$(1-\epsilon)\, t\_max_i \cdot speed_i \leq N_i \cdot WIDTH_i \leq (1+\epsilon)\, t\_max_i \cdot speed_i, \qquad \forall i \in \{1, \ldots, k\}. \tag{13}$$

$$(1-\epsilon)\, N_i \cdot TS_i \leq HEIGHT_i \leq (1+\epsilon)\, N_i \cdot TS_i, \qquad \forall i \in \{1, \ldots, k\}. \tag{14}$$

The objective function (5) uses the ext_SAR external function that calls SAR Optimizer to calculate the total probability of success. SAR Optimizer uses case (problem instance) data, including environmental conditions, the characteristics of the search units, and the characteristics of the search object, to compute the probability of success with simulation, according to Eqs. (1)–(4). Each rectangle R_i is defined by the following decision

variables: its lower-left corner coordinates (LLX_i, LLY_i), its dimensions ($HEIGHT_i$ and $WIDTH_i$), the number of search legs N_i, and the track spacing TS_i.

Constraints (6)–(8) state the relation between the corners of rectangle i by applying a two-dimensional rotation matrix to its width and height vectors based on analytic geometry (Lengyel, 2011). In model M, θ_i is known and is computed from the $angle_i$ parameter for all search units:

$$\theta_i = angle_i \cdot \pi/180\,, \qquad \forall i \in \{1,\dots,k\}\,. \tag{15}$$

To calculate the $angle_i$ parameter value for a specific problem instance, we first apply principal component analysis (PCA) (Abdi and Williams, 2010) on the on-scene particles to find the direction of maximum variance in the drift. Then, we compute the minimum enclosing rectangle around the on-scene particles aligned with that direction. The $angle_i$ parameter is the orientation of that rectangle.

Constraints (9) force corner points $(X,Y) \in \{(LRX_i, LRY_i), (ULX_i, ULY_i), (URX_i, URY_i)\}$ to lie within the search environment. Constraints (10) and (11) ensure that the rectangle's size remains operationally feasible. By bounding the ratio of length to width, constraints (10) and (11) prevent the solver from converging to very long and thin rectangles. While such thin rectangles might be mathematically optimal, they are operationally unrealistic, as they would either result in inefficiently short search legs and excessive turning overhead that degrades the actual search effectiveness, or result in a single long search leg. Constraint (12) ensures that rectangles i and j do not overlap if $deconfliction_{ij} = 1$, i.e., when rectangles i and j have to be horizontally separated. The condition $R_i \cap R_j = \emptyset$ is evaluated using an external Python function ext_no_intersect. Constraint (13) and (14) enforce consistency between the rectangle's dimensions and the search effort. Constraints (13) ensure the total distance traversed by a search unit ($N_i \cdot WIDTH_i$) is consistent with its total available effort ($t\_max_i \cdot speed_i$). Constraints (14) ensure a rectangle height is consistent with the number of search legs and track spacing.

3.1.2. Model MC

Model MC objective function is (16), subject to constraints (6)–(14), and (17).

$$\max \quad POS = \text{ext\_SAR}(\{LLX_i, LLY_i, WIDTH_i, HEIGHT_i, TS_i, N_i\}_{i=1}^{k}) \tag{16}$$

$$\text{s.t.} \quad \text{Constraints (6)–(14)}\,.$$

$$\min_{i:T_i \geq t\_act} \text{ext\_coverage}(LLX_i, LLY_i, WIDTH_i, HEIGHT_i, TS_i, N_i) \geq \varphi\_min\,. \tag{17}$$

Constraint (17) ensures that each rectangle covers at least φ_min percent of the particles. To check whether a particle falls within the search rectangle, we use the ray-casting (ray-crossing) point-in-polygon detection algorithm. This method draws a horizontal ray from a particle to a direction and counts how many times it crosses the polygon's edges. If the number of crossings is odd, the particle is inside the polygon, and if not, the particle is outside it (O'Rourke, 1998). This constraint is implemented using this algorithm as an external function ext_coverage.

3.1.3. Model ME

In addition to decision variables for rectangle position and size, model ME decides the on-scene time T_i and can untask unnecessary search units through the activation threshold t_act. Model ME objective function is (18), subject to constraints (6)–(11), and (19)–(22).

$$\max \quad POS = \text{ext\_SAR}(\{LLX_i, LLY_i, WIDTH_i, HEIGHT_i, TS_i, N_i, T_i\}_{i=1}^{k}) \tag{18}$$

$$\text{s.t.} \quad \text{Constraints (6)–(11)}\,.$$

$$\text{ext\_no\_intersect}(\{LLX_i, LLY_i, WIDTH_i, HEIGHT_i, TS_i, N_i, T_i\}_{i=1}^{k})\,. \tag{19}$$

$$(1-\varepsilon)\,T_i \cdot speed_i \leq N_i \cdot WIDTH_i \leq (1+\varepsilon)\,T_i \cdot speed_i\,, \qquad \forall i \in \{1,\dots,k\}\,. \tag{20}$$

$$(1-\varepsilon)\,N_i \cdot TS_i \leq HEIGHT_i \leq (1+\varepsilon)\,N_i \cdot TS_i\,, \qquad \forall i \in \{1,\dots,k\}\,. \tag{21}$$

$$\max_{i=1,\dots,k} T_i \;\geq\; t\_act\,. \tag{22}$$

Constraint (19) redefines constraint (12) to take into account time decision variables T_i. Constraint (19) ensures that rectangles i and j do not overlap if $deconfliction_{ij} = 1$ and both search units i and j tasked. The external function ext_no_intersect implements the relation $R_i \cap R_j = \emptyset$ for the enabled search units. Constraints (20) and (21) enforce consistency between the rectangle's dimensions and the search effort. Constraints (20) ensure the total distance traversed by a search unit ($N_i \cdot WIDTH_i$) is consistent with its effort ($T_i \cdot speed_i$) and constraints (21) ensure the rectangle's height is consistent with the number of search legs and track spacing. Constraint (22) ensures that at least one search unit is enabled.

3.1.4. Model MEC

Model MEC objective function is (23), subject to constraints (6)–(11), (19)–(22), and (24).

$$\max \quad POS = \text{ext\_SAR}(\{LLX_i, LLY_i, WIDTH_i, HEIGHT_i, TS_i, N_i, T_i\}_{i=1}^{k}) \tag{23}$$

$$\text{s.t.} \quad \text{Constraints (6)–(11)}\,.$$

$$\text{Constraints (19)–(22)}\,.$$

$$\min_{i:T_i \geq t\_act} \text{ext\_coverage}(LLX_i, LLY_i, WIDTH_i, HEIGHT_i, TS_i, N_i, T_i) \geq \varphi\_min\,. \tag{24}$$

Constraint (24) redefines constraint (17) to take into account time decision variables T_i. Again, the external function ext_coverage implements ray-casting (ray-crossing) point-in-polygon detection (O'Rourke, 1998).

3.1.5. Model MECA

Model MECA objective function is (25), subject to constraints (6)–(11), (20)–(22), and (26)–(28). In addition to the LLX_i, LLY_i, $HEIGHT_i$, and $WIDTH_i$ variables defining a rectangle i, the model is based on $ANGLE_i$, a rotation variable. The other decision variables used in the model are

Table 3
Drift instance parameters.

Drift name	Drift start time	Drift end time
Person-in-water	2025-09-08 12:00:00	2025-09-10 04:00:00
Life Raft1	2025-09-08 12:00:00	2025-09-10 04:00:00
Fishing Vessel	2025-09-08 12:00:00	2025-09-10 04:00:00
Sailboat1	2025-09-08 12:00:00	2025-09-10 04:00:00
Sailboat2	2025-09-07 12:00:00	2025-09-10 04:00:00
Life Raft2	2025-09-07 00:00:00	2025-09-10 16:00:00

the number of search legs N_i, the track spacing TS_i, and the on-scene time T_i.

$$\max \quad POS = \text{ext\_SAR}(\{LLX_i, LLY_i, WIDTH_i, HEIGHT_i, TS_i, N_i, T_i, ANGLE_i\}_{i=1}^{k}) \tag{25}$$

s.t. Constraints (6)–(11).

Constraints (20)–(22).

$$\theta_i = ANGLE_i \cdot \pi/180, \qquad \forall i \in \{1, \ldots, k\}. \tag{26}$$

$$\text{ext\_no\_intersect}(\{LLX_i, LLY_i, WIDTH_i, HEIGHT_i, TS_i, N_i, T_i, ANGLE_i\}_{i=1}^{k}). \tag{27}$$

$$\min_{i:T_i \geq t\_act} \text{ext\_coverage}(LLX_i, LLY_i, WIDTH_i, HEIGHT_i, TS_i, N_i, T_i, ANGLE_i) \geq \varphi\_min. \tag{28}$$

Constraints (26) convert the angles from degrees to radians. Constraint (27) implements constraint (19) to also take into account the $ANGLE_i$ variables. As such, constraint (27) ensures that rectangles i and j do not overlap if they have to be deconflicted ($deconfliction_{ij} = 1$) and if both search units i and j are tasked. Constraint (27) uses the external function ext_no_intersect to verify the relation $R_i \cap R_j = \emptyset$ for the tasked search units. Constraint (28) implements constraint (24) for the case where we have both on-scene time and angle decision variables, i.e., T_i and $ANGLE_i$. The ext_coverage external function still implements ray-casting (ray-crossing) point-in-polygon detection (O'Rourke, 1998).

3.2. Solving the models

To solve the proposed models, the chosen solver needs to allow blackbox external functions as well as a modeling language enabling non-linear constraints and constraints based on constraint programming. For that purpose, we chose Hexaly (Hexaly, 2021), a hybrid commercial solver we selected for its ability to integrate heuristic search with exact mathematical programming. Hexaly supports external functions in both the constraints and the objective, a critical feature for our approach, as it allows the optimizer to call the external SAR simulation to evaluate candidate search plans.

Hexaly also supports the use of a surrogate to estimate the objective function, i.e., a model that approximates the response surface of the objective function instead of calling the simulation. The surrogate can help guide the solver, and a call to the surrogate is cheaper to perform than a call to the simulator. As described in Tenaud (2022), Hexaly uses Gutmann's radial basis function (RBF) as a surrogate function. The RBF guarantees convergence for the global optimization of a continuous nonconvex function and can be applied when derivatives are unavailable (Gutmann, 2001). Following each external function evaluation that calculates the objective value, the RBF surrogate is updated with the new objective function's value. The optimization process alternates between exploitation, which focuses on improving promising feasible solutions, and exploration, which searches new regions of the decision space. Enabling the surrogate for an external function introduces stochasticity in the output of the optimization process, since Hexaly uses random sampling when selecting evaluation points in its blackbox optimization process (Tenaud, 2024). This may lead to different optimal solutions across multiple runs of the same model.

Given that models M and ME do not implement a particle coverage constraint, they may generate search rectangles that do not cover any particles. To avoid such rectangles, we apply a post-optimization filtering step to remove rectangles with particle coverage below φ_min and recompute the probability of success on the remaining rectangles. This step only removes rectangles that do not contribute to detections in the simulation.

4. Experiments

To evaluate the proposed models, we conducted experiments using 18 realistic instances developed in collaboration with a SAR mission coordinator. They were constructed using six distinct drift files (Table 3) representing different search object types: one drift with a person-in-water, two drifts with a life raft, one drift with a fishing vessel, and two drifts with a sailboat. For each drift file, three distinct configurations of search units were defined, representing different combinations of available fixed-wing aircraft and helicopters: 4 fixed-wing aircraft (*4F*), 3 helicopters and 3 fixed-wing aircraft (*3H3F*), and 6 fixed-wing aircraft (*6F*). We also defined the meteorological visibility in which they operate, time on-scene, and altitudes (Table 4). Horizontal spatial deconfliction is only required if units operate at the same altitude during overlapping time periods.

Based on parameters provided by the CCG, we set *ts_min* to 0.2 nautical miles and *ts_max* to 10 nautical miles in all scenarios. In models M and MC, search effort is a parameter defined for each search unit, which means the available on-scene time is allocated in total (plus or minus an ϵ tolerance of $1e-6$ hours). In models MECA, MEC, and ME, the search effort is a decision variable defined for each search unit—it can therefore vary from 0 up to the available on-scene time. In MECA, MEC, and ME, the activation threshold *t_act* is set to 30 min (0.5 h), meaning a search unit is considered activated if it searches for at least 30 min. The 30-minute threshold is based on practice, where a shorter search time is not operationally realistic. For example, in scenario *4F*, all fixed-wing aircraft have a maximum of 4 h time on-scene. Therefore, a search unit is considered activated if its search time is between 30 min and 4 h, and untasked otherwise.

Models M and MC, i.e., the models without flexible effort, force all effort to lie within the search environment. Because our instances contain configurations with a large amount of available effort, a buffer parameter b is used in these models to slightly enlarge the search environment. This buffer parameter is set to 5% of the search environment's height in models M and MC.

Table 4
Search units involved in each search configuration.

	Speed (knots)	Visibility (NM)	Time or altitude deconflicted	Altitude (feet)	Time on-scene
4F search configuration					
4 fixed-wing aircraft	150	5	No	500	2025/09/08 (11:00-15:00)
6F search configuration					
6 fixed-wing aircraft	150	5	No	500	2025/09/08 (11:00-15:00)
3H3F search configuration					
1st fixed-wing aircraft	130	10	Yes	500	2025/09/09 (12:00-14:00)
2nd fixed-wing aircraft	130	10	Yes	500	2025/09/10 (00:00-02:00)
3rd fixed-wing aircraft	130	10	Yes	500	2025/09/09 (00:00-02:00)
1st helicopter	60	10	Yes	300	2025/09/09 (00:00-02:00)
2nd helicopter	60	10	Yes	300	2025/09/10 (00:00-02:00)
3rd helicopter	60	10	Yes	300	2025/09/09 (12:00-14:00)

The aspect ratio α is set to 5% (without units), meaning that the ratio between the height and width of a rectangle must be at least 5%, so the rectangles do not become excessively slender. The minimum particle coverage threshold φ_min is set to 1 percent of the particles, forcing active rectangles to cover at least 1% of the particles. For models MECA, MEC, and MC, this prevents the generation of rectangles that contain no particles.

Experiments were run on a GNU/Linux operating system with an AMD Ryzen 9 5900X 12-core processor and 64 GB of system memory. The code is in Python, with the exception of the ext_coverage external function that is implemented in C to achieve faster execution. Simulations were run in SAR Optimizer, developed in C++, and the modeling API (application programming interface) of the Hexaly solver was used. We configured an external function ext_SAR in the Hexaly solver to call SAR Optimizer's simulation module as the objective function. We used the *shapely* Python library for the external function ext_no_intersect that enforces the non-overlapping constraint. Because of the stochasticity introduced by surrogate modeling of the Hexaly solver, we performed 30 independent runs for each of the 18 instances for the main experiment.

Finally, in practice, based on our experience and private communications (Abi-Zeid, 2026b), a SAR mission coordinator needs a search plan with the highest possible probability of success within a short time. To simulate this context, we chose a solving time of 5 min as a near-operational experimental choice for the majority of the analysis. We also provide results for longer runtimes to evaluate the models.

5. Results and discussion

Fig. 3 shows the per-instance probability of success distribution across models for all instances, except *3H3F Fishing Vessel* and *3H3F Life Raft1*. We exclude the *3H3F Fishing Vessel* and *3H3F Life Raft1* instances to avoid overcrowding the figure and because the solver achieved feasible solutions with a probability of success close to 1 with all models on all runs. For each instance, models are ordered in increasing order of their mean probability of success across the 30 runs. For each run, these results represent the best solutions obtained in the allocated runtime, namely 5 min, with no guarantee of optimality. Each box shows the range between the 25th percentile (Q1) and the 75th percentile (Q3) of the data. The whiskers extend from the edges of the box to the smallest and largest values within 1.5 times the interquartile range. We consider data points that fall outside the whiskers as outliers. Cases where the solver failed to produce a feasible solution in 30 runs with a given model are indicated by an "X" at POS = 0. Such cases occur only with models M and MC, i.e., models without flexible effort.

As seen in Fig. 3, the solver can reach a probability of success close to 1 on several *Fishing Vessel* and *Life Raft1* instances, as long as the model leads to a feasible solution. Those instances are search cases where detection is easier, and the model choice has little effect on the probability of success. In contrast, the probability of success of the *Person-in-water* instances with search scenarios *4F* and *6F* is near POS = 0 for all models. This is not unusual because the detectability of a small object (a person's head) is much lower than that of a larger object, such as a life raft.

The remaining instances show clear differences across models. Feasibility separates the models more than their final probability of success values. The presence of the flexible effort is the defining factor in the clear separation into two groups. Only the MECA, MEC, and ME models (i.e., those with flexible effort) enable the solver to achieve feasible results on all instances. With model M, however, the solver reaches the maximum probability of success on a few instances. Nonetheless, it cannot find a feasible solution on 7 out of 18 instances. With model MC, the solver cannot find a feasible solution on half of the instances (within the time limit), mostly for *4F* and *6F* instances that involve fixed-wing aircraft. With fixed-wing aircraft, the fixed effort creates large rectangles within a bounded environment under deconfliction, making the instances infeasible when using model MC. Model ME is feasible on all instances because the solver can choose the on-scene time dynamically and can untask some search units when needed, i.e., the solver can find feasible solutions with fewer search rectangles. Models MEC and MECA, i.e., the ones with the minimum particle coverage and flexible angle, do not lead to a better probability of success than model ME. The addition of the minimum particle coverage or of the flexible angle tends to reduce the median probability of success. Adding additional decision variables and constraints makes the model more complex, which can lead to a lower median probability of success across runs when a 5-minute time limit is enforced. When adding variables, the solution space size increases. When adding constraints, the solver might have a harder time finding feasible solutions, or external constraints might consume solving time. This means that the solver might need more time on more complex models than on simpler ones to explore the solution space and improve its solution.

5.1. Impact of adding flexible effort

Because flexible effort is an operational requirement, we compare the results of models M and ME in Table 5, where we see the best probability of success across the 30 runs (max POS), mean probability of success over the 30 runs (mean POS), the 95% confidence intervals (95% CI) for the mean POS values computed over the 30 independent runs, and the coefficient of variation of the probability of success (CV). Also, for the model ME, the table includes the average number of activated search units across runs. Entries filled with "–" indicate that no feasible solution has been found across all runs. For the rest, the solver reached a feasible solution in all 30 runs.

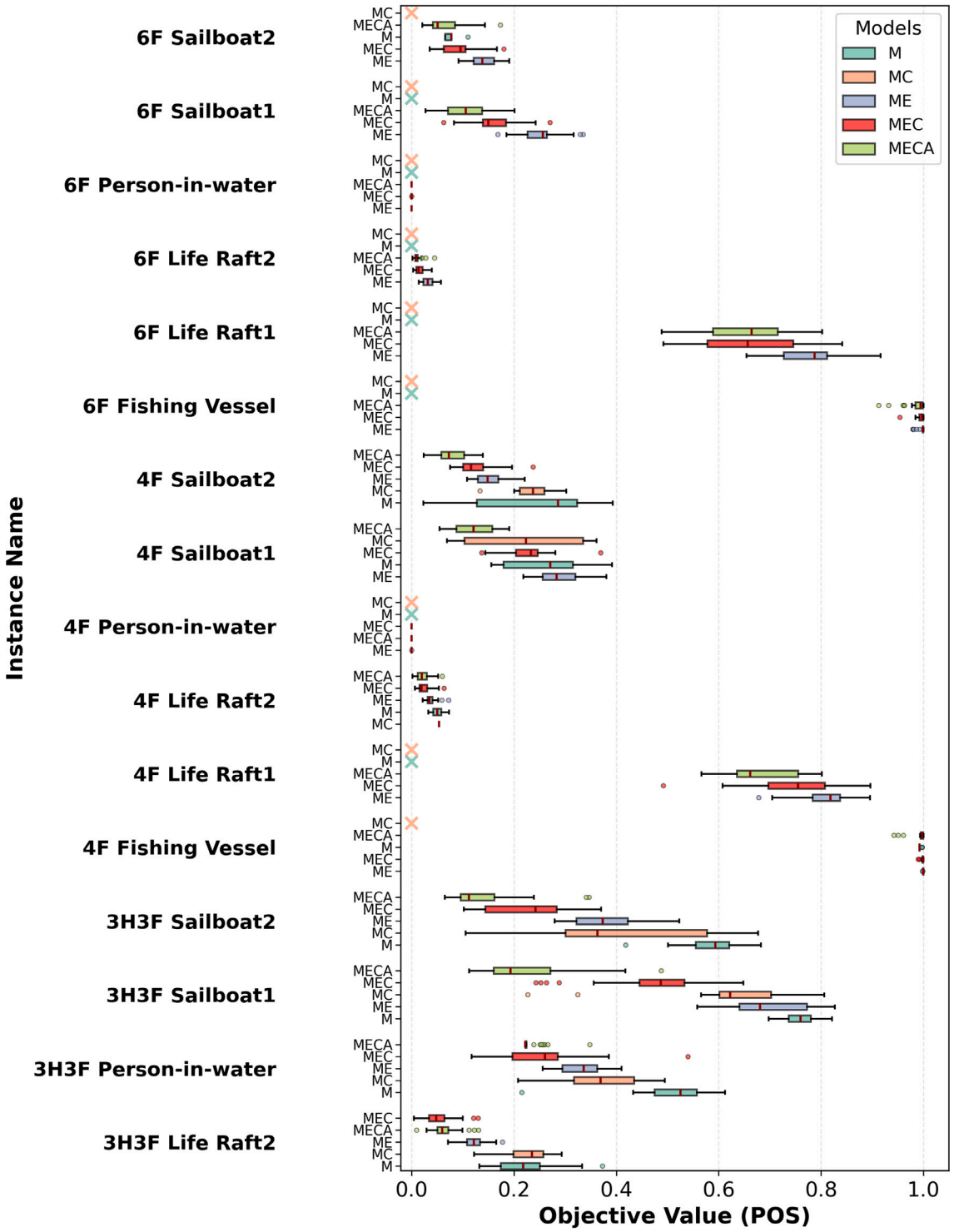

Fig. 3. The box plots of the optimization results of 30 runs with a 5-minute solving time for all five models (M, MC, ME, MEC, MECA) sorted by the mean probability of success (POS) for each instance. Instances *3H3F Fishing Vessel* and *3H3F Life Raft1* are omitted because the solver finds a feasible solution with a POS close to 1 with all models.

In terms of feasibility of model ME, the solver achieves a feasible solution on all 18 instances. In contrast, model M is feasible for the *3H3F* instances and for a subset of the *4F* and *6F* instances. With model M, the solver returns no feasible solution in *4F Life Raft1*, *4F Person-in-water*, *6F Fishing Vessel*, *6F Life Raft1*, *6F Life Raft2*, *6F Person-in-water*, and *6F Sailboat1*. This is due to horizontal spatial deconfliction and the effort assignment requirements. Unlike the fixed-wing aircraft from the *3H3F* instances, fixed-wing aircraft from the *4F* and *6F* instances operate with overlapping on-scene times and fly at the same altitude. Enforcing a fixed effort for all fixed-wing aircraft in model M violates horizontal spatial deconfliction constraints on many instances and prevents feasible solutions. Using model ME can help avoid this problem because the solver finds

Table 5
Results for 30 runs of 5 min of the solver with models M and ME grouped by instance: highest probability of success across runs (max POS), mean probability of success (mean POS), 95% confidence intervals around the mean (95% CI), and coefficient of variation in percentage (CV %) for both models. The average number of activated search units is also shown for model ME.

	Model M				Model ME				
Case	Max POS	Mean POS	95% CI	CV (%)	Max POS	Mean POS	95% CI	CV (%)	Avg. active units
3H3F Fishing Vessel	0.9999	0.9999	[0.9999, 0.9999]	0.0000	0.9999	0.9999	[0.9999, 0.9999]	0.0000	4.40/6
3H3F Life Raft1	0.9999	0.9999	[0.9999, 0.9999]	0.0002	0.9999	0.9996	[0.9995, 0.9998]	0.0398	4.90/6
3H3F Life Raft2	0.3729	0.2229	[0.2005, 0.2452]	26.4108	0.1774	0.1216	[0.1122, 0.1311]	20.4336	4.47/6
3H3F Person-in-water	0.6123	0.5128	[0.4848, 0.5408]	14.3631	0.4102	0.3319	[0.3158, 0.3480]	12.7895	5.33/6
3H3F Sailboat1	0.8213	0.7592	[0.7469, 0.7715]	4.2573	0.8269	0.6980	[0.6672, 0.7288]	11.6037	5.33/6
3H3F Sailboat2	0.6826	0.5887	[0.5665, 0.6109]	9.9249	0.5229	0.3797	[0.3555, 0.4038]	16.7569	4.57/6
4F Fishing Vessel	0.9975	0.9929	[0.9922, 0.9936]	0.1857	0.9999	0.9995	[0.9994, 0.9997]	0.0398	2.20/4
4F Life Raft1	–	–	–	–	0.8958	0.8104	[0.7928, 0.8281]	5.7238	2.40/4
4F Life Raft2	0.0729	0.0507	[0.0465, 0.0550]	22.2563	0.0719	0.0374	[0.0334, 0.0415]	28.3957	2.73/4
4F Person-in-water	–	–	–	–	0.00006	0.00004	[0.0000, 0.0000]	21.2474	1.60/4
4F Sailboat1	0.3916	0.2576	[0.2293, 0.2858]	28.8942	0.3803	0.2924	[0.2751, 0.3097]	15.5631	2.87/4
4F Sailboat2	0.3928	0.2366	[0.1926, 0.2807]	48.9952	0.2209	0.1528	[0.1418, 0.1639]	19.0769	3.00/4
6F Fishing Vessel	–	–	–	–	0.9999	0.9966	[0.9942, 0.9990]	0.6227	3.13/6
6F Life Raft1	–	–	–	–	0.9163	0.7752	[0.7514, 0.7990]	8.0694	2.43/6
6F Life Raft2	–	–	–	–	0.0569	0.0329	[0.0285, 0.0373]	35.0779	3.53/6
6F Person-in-water	–	–	–	–	0.00005	0.00003	[0.0000, 0.0000]	26.7425	2.00/6
6F Sailboat1	–	–	–	–	0.3352	0.2487	[0.2340, 0.2633]	15.5258	3.40/6
6F Sailboat2	0.1098	0.0745	[0.0712, 0.0777]	11.4747	0.1905	0.1402	[0.1309, 0.1495]	17.4977	3.50/6

solutions with a lower on-scene time or by deactivating unnecessary search units. Reducing the search effort reduces the rectangle size (and search pattern length) and the number of rectangles that must satisfy horizontal spatial deconfliction constraints.

For the *3H3F Fishing Vessel* and *3H3F Life Raft1* instances, the solver achieves a mean and maximum probability of success above 99% with both models. However, with model ME, the solver achieves these values while activating fewer search units (4.4 and 4.9, on average, out of 6, respectively). The CV values show the variability in the results. Both models M and ME show near-zero variability on the easy instances with a probability of success close to 100%. The hard instances show a different pattern. The *Life Raft2* and *Person-in-water* search instances produce large CV values in both models, which shows high run-to-run variability. Based on the confidence intervals, for cases with mean probability of success near 1, the intervals are very narrow (*3H3F Fishing Vessel*, *3H3F Life Raft 1*, *4F Fishing Vessel*, and *6F Fishing Vessel*). The widest confidence intervals occur in *4F Sailboat2* with model M (0.088), and *3H3F Sailboat1* with model ME (0.062), indicating that all interval widths remain below 0.09. This suggests 30 independent runs provide a reliable estimate of the mean probability of success.

When the solver returns feasible solutions with both models M and ME, it often gets a higher mean and maximum probability of success with M compared to ME within 5 min. This occurs in *3H3F Life Raft2*, *3H3F Person-in-water*, and *3H3F Sailboat2*. Because ME adds the on-scene time decision variables and their related constraints, the solver must search in a larger solution space within the same time budget. The solver then returns a solution with a lower probability of success value on some instances, even when it maintains feasibility.

The *4F* and *6F* instances differ in mean probability of success, maximum probability of success, and CV when both models M and ME lead to feasible solutions. In *4F Sailboat1*, ME enables the solver to increase the mean probability of success (29.24% vs. 25.76%) and reduces the CV value compared to M (15.56% vs. 28.89%). In *4F Sailboat2*, when using M, the solver achieves a higher mean and maximum probability of success and also shows a high CV (48.99%). In *6F Sailboat2*, ME enables the solver to improve both maximum probability of success (19.05% vs. 10.98%) and mean probability of success (14.02% vs. 7.45%) compared to M.

Fig. 4 compares the average used effort of models M and ME for all instances, across 30 runs. Cases where the solver failed to produce a feasible solution in 30 runs with a given model are indicated by an "X". Model ME uses less effort than model M on every instance where both models find a feasible solution. Model M uses its full available effort in all three scenarios (969 nautical miles on the *3H3F* instances, 2040 nautical miles on the feasible *4F* instances, and 3060 nautical miles on the feasible *6F* instance). The used effort gap between the two models is larger in the *4F* and *6F* scenarios than in the *3H3F* scenario. In the *3H3F* scenario, both models are feasible across all instances, but model ME still uses less effort. We observe that, across all instances, model ME, on average, saves hundreds of nautical miles of effort.

Fig. 5 presents an example of the solution achieved by the solver with model ME. The figure shows the particles during the time on-scene of instance *3H3F Sailboat1* and the search plan as search rectangles. *Sailboat1* drift forms an elongated particle cloud. The generated rectangles cover most of the on-scene particles, leading to an increase in the probability of success. Although this instance includes six available search units, the solver enabled only five of them. Also, during post-optimization filtering, we removed one rectangle that did not cover any particle. With three fixed-wing aircraft and one helicopter, a probability of success of 82.69% is achieved.

To check the sensitivity of our results based on the solving time, we ran the solver on models M and ME for 120 min with 5 runs on four instances (*3H3F Person-in-water, 4F Sailboat2, 4F Sailboat1, 6F Sailboat2*) that are feasible with both M and ME. The instances are cases with a different difficulty in terms of the probability of success achieved by the solver within a 5-minute solving time. From each run, we extracted the maximum probability of success reached by 5, 15, 30, 60, and 120 min solving time cutoffs.

Fig. 6 presents the average value of maximum probability of success across 5 runs for 5 min, 15 min, 30 min, 60 min, and 120 min of solving time cutoffs. The results show that increasing the solving time improves the probability of success for both M and ME in most cases.

When solving time extends to 120 min, the gap between the two models decreases on some instances. On the *3H3F Person-in-water* and *4F Sailboat2* instances, we observe that the gap between the results obtained with model ME and with model M is closing. On the *6F Sailboat2* instance, the solver obtains better results with model ME than with model M at all solving time cutoffs. On the *4F Sailboat1* instance, the relative ranking varies between the runs corresponding to different solving time cutoffs. On the 120-minute runs, the results are better with model ME than with model M.

We conclude that extending the solving time favors model ME, whereas M might be preferable for short runs on instances where the solver can quickly find good feasible solutions.

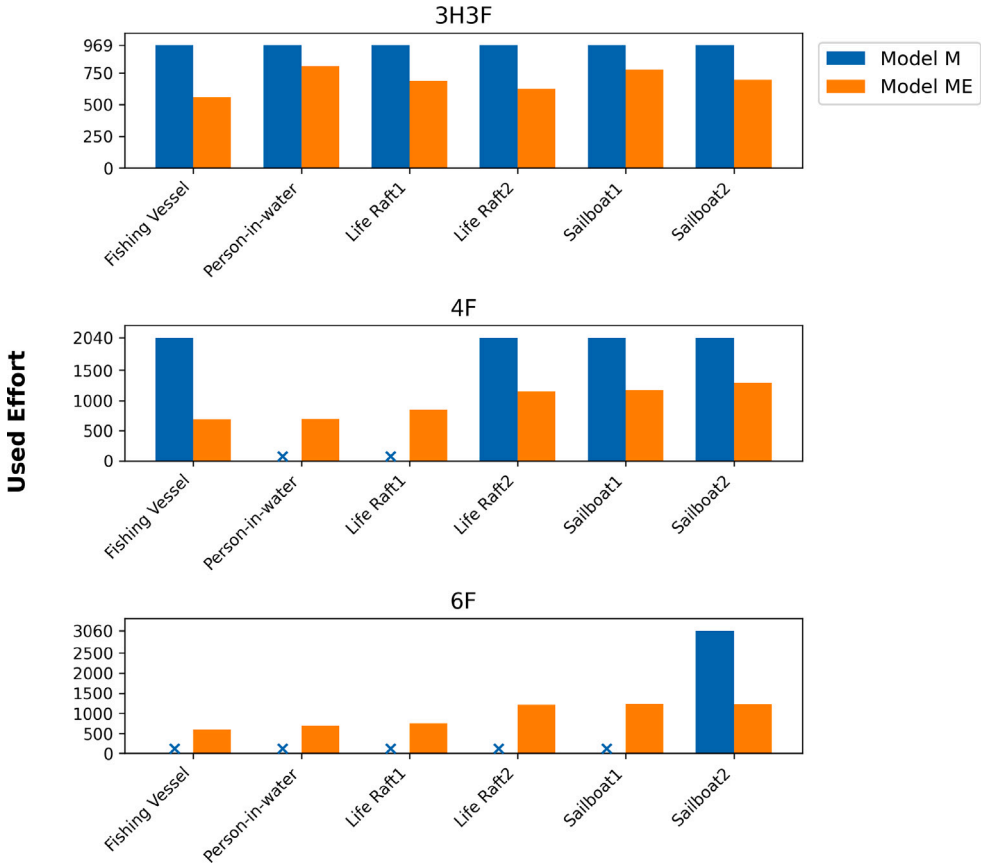

Fig. 4. The average used effort of models M and ME in nautical miles, for all instances, across 30 runs with a 5-minute solving time. "X" indicates no feasible solution across all runs.

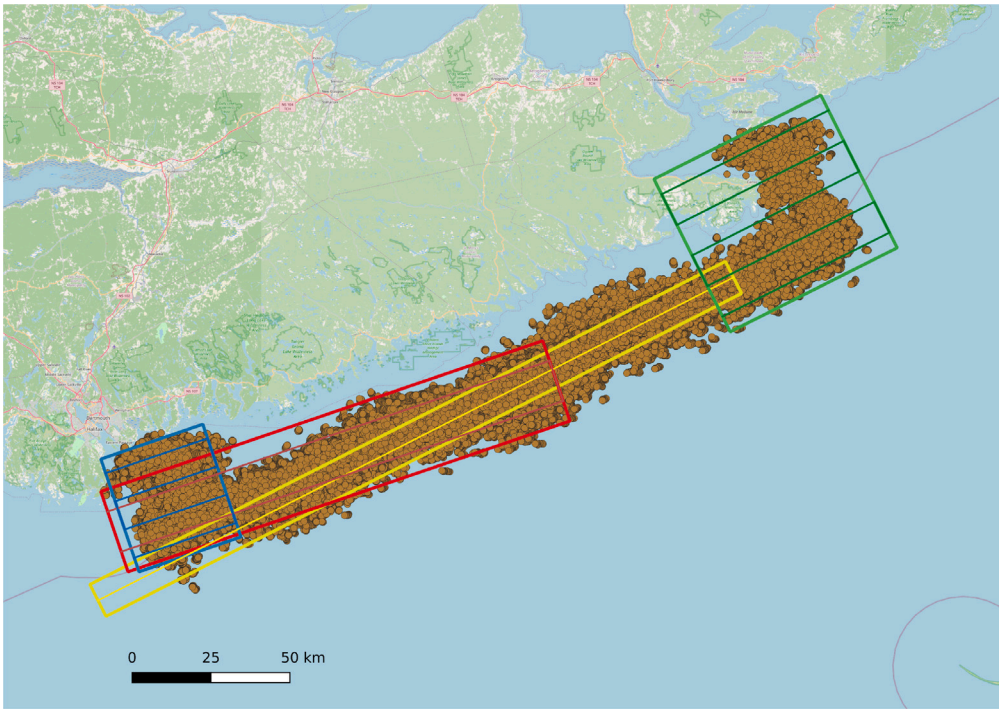

Fig. 5. The proposed search plan generated within a 5-minute solving time for instance *3H3F Sailboat1*, where green, yellow, and red search rectangles belong to fixed-wing aircraft, and the blue search rectangle belongs to a helicopter. The lines inside the rectangles show the search patterns that the search units will follow.

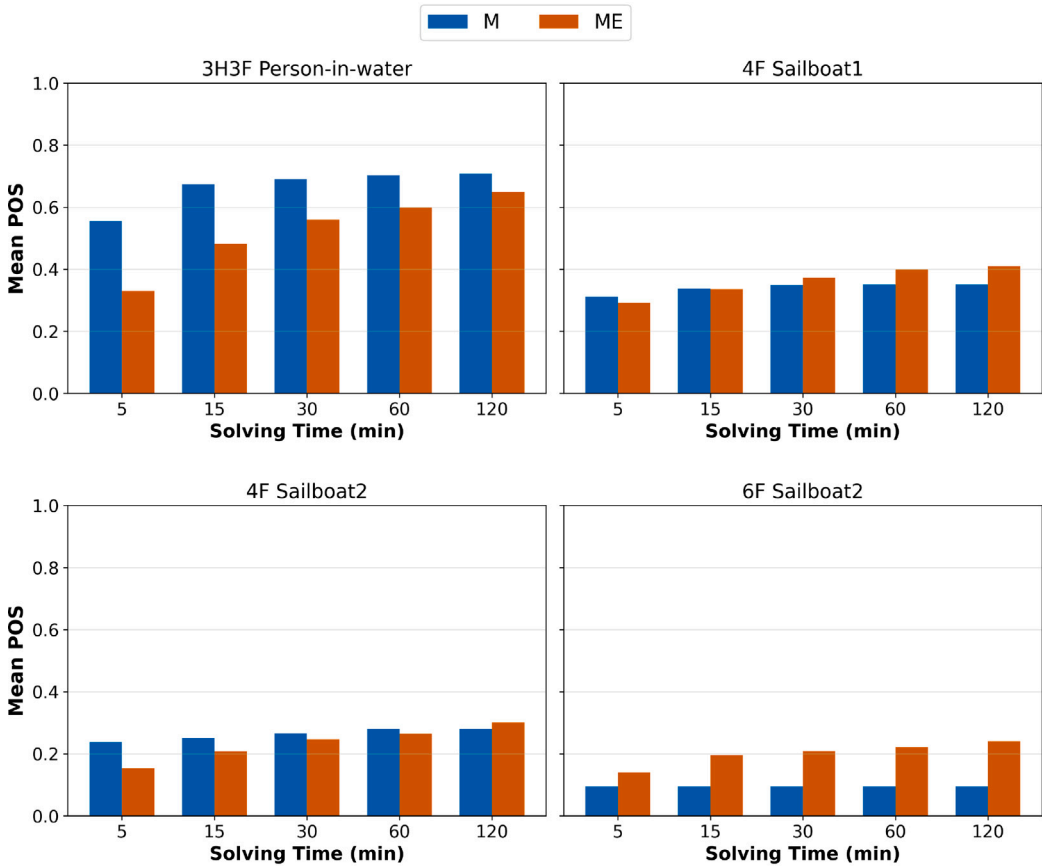

Fig. 6. Sensitivity analysis of solving time with different time cutoffs. The bars indicate the mean incumbent probability of success (POS) across 5 runs after 5, 15, 30, 60, and 120 min of solving time cutoffs.

5.2. Summary

Regarding which level of modeling detail is sufficient to generate operationally feasible search plans with a high probability of success within a sufficiently short time frame, our main criterion is to achieve the highest probability of success within a 5-minute runtime. The effort usage is our second operational criterion. Under this interpretation, we choose M as the best probability of success-oriented model and ME as the best effort-efficient model, which maintained a relatively higher probability of success than the other models. Model M, however, tends to be infeasible when search effort is high with respect to the drift spread. As a result, by enabling flexible effort, model ME provides the best trade-off between modeling detail and performance.

Based on Fig. 3, adding the flexible angle and the minimum particle coverage does not improve the probability of success over ME under the same 5-minute solving time and often lowers the median probability of success. Therefore, using a fixed drift-aligned angle parameter instead of a flexible angle decision variable, and using post-optimization filtering to remove empty rectangles instead of the particle coverage constraint, yields better search plans.

We can compare models M and ME across three aspects: feasibility, optimization progress, and effort usage. Fixed effort can make model M infeasible because search rectangles must satisfy horizontal deconfliction within a bounded area. Model ME avoids these infeasibilities by reducing on-scene time or leaving search units untasked. This extra flexibility can restore feasibility, but it comes at a cost, as the solver must search a larger decision space within the same 5-minute limit. At the operational level, after untasking the search units in model ME, they remain available for later assignments when the search environment changes or new information is gathered during the mission.

The horizontal spatial deconfliction constraint fulfills an operational requirement in our model-and-solve approach to optimization. It can, however, lead to infeasibility when combined with fixed effort, such as in the *4F* and *6F* configurations, where all fixed-wing aircraft share the same on-scene time window and altitude, and cannot overlap inside the search environment. Because model M forces full effort, it generates rectangles that cannot fit in the environment without overlap, so we get infeasible solutions. Model ME remedies this situation by reducing the on-scene time or effort of each search unit and deactivating some, making it possible to fit inside the bounded environment. Therefore, when we have too much effort available, it is better to use model ME, as it can adjust the effort and return a feasible search plan.

The results further indicate that within a 5-minute solving limit, model M achieves higher mean and maximum probability of success on certain *3H3F* instances, provided a feasible solution exists. Because ME includes additional decision variables compared to M, it leads to a larger search space that must be explored by the solver within the same time budget. This causes the mean probability of success to drop in some cases with ME as more time would be needed for the solver to reach a better solution. Although the solver untasks one to three search units when using model ME, the cases with high probability of success values (*Fishing Vessel* and *Life Raft1*) still reach a probability of success near 1 with fewer search units. These unassigned units represent surplus capacity, giving mission coordinators the flexibility to respond to evolving requirements, initiate new search tasks, or just save resources. In essence, the 5-minute solving time controls the solution quality and the observed trade-offs between

the probability of success and the saved effort. In the *Supplementary material*, we analyze and compare all five models by adding a convergence plot of the average incumbent probability of success over 5-minute time across all 30 runs to assess how the complexity of the model affects its performance within 5 min.

In all three configurations, the drifts with *Life Raft2* and *Person-in-water* search objects achieve the lowest probability of success values. The difference is more obvious on *4F* and *6F* instances because the meteorological visibility in these two configurations was set to 5 nautical miles, which is half of that in the *3H3F* instances.

6. Limitations

The first limitation of this work is related to the use of a closed-source commercial solver in our experiments. Despite the fact that we demonstrated the feasibility and benefits of a model-and-solve approach to optimization for maritime SAR planning, closed-source solvers remain opaque machinery. Compared to an open solver, whose mechanics and implementation can be understood by looking at the code, the behavior of closed-source solvers cannot be easily understood. This means that, even when a closed-source solver uses algorithms from the public domain, some of the results might be related to their particular implementation, and there is no easy way to show this. As a result, changing the solver or its version might unexpectedly impact the results, which implies that the performance reported might be at least partially solver-dependent. Another issue is related to the licensing costs when a commercial solver is used operationally. Nonetheless, commercial solvers are widely used as benchmark software because they are often considered state-of-the-art.

The second limitation of our work is the 5-minute runtime limit used in the main experiment. It is based on the practice of maritime SAR planning, where time is of the essence. We used a 5-minute solving time for the purpose of reproducing the context where a SAR mission coordinator has a short time to task search units, a realistic requirement in practice according to our experience and private communications (Abi-Zeid, 2026b). This is not to say, however, that the results between models will be the same if we allow for a longer solving time. A longer runtime can improve the probability of success. We checked it by extending the runtime to 120 min for model `M` and model `ME` on four representative instances. Extending the solving time favors model `ME`. Also, it should be noted that the amount of work performed by the solver within a fixed solving time is implementation (software) and hardware dependent. Changing the hardware might change the results. Similarly, Python was used for coding part of the code used to run external functions. Changing the programming language might change the results.

Third, we observed that simpler models can lead to better solutions compared to more complex models, e.g., adding the angle as a decision variable does not necessarily lead to better solutions compared to models where the angle is a fixed parameter. Although a model-and-solve approach to optimization can make it easier to implement complex constraints and to allow for more complex formulations, using complex models might not always be the right approach. This intuition is confirmed by our empirical results showing that more expressive (complex) models can necessitate more solving time. Notably, model `M` allows the solver to reach a higher objective value than model `ME` within 5 min on some instances. However, when we increase the allowed solving time from 5 min to 120 min, `ME` tends to outperform model `M`. This is expected, however, that `ME` would be harder to solve than `M` due to the presence of more decision variables. It shows that parsimony is the key, and developers need to resist the temptation to complexify their models by making a trade-off between complexity and solving time. Similarly, increasing the number of search units increases the model complexity and potentially the solving time. This limitation, however, is not particular to a model-and-solve approach to optimization. The performance of problem-specific algorithms will also degrade as we increase the number of search units or, more generally, the number of decisions to be made.

Fourth, regarding model-building, one of the perceived limitations might be the fact that mathematical modeling is not a skill acquired by most software developers. In practice, this could mean that a team might need additional expertise to implement a model-and-solve approach to optimization.

Fifth, although we took care in building 18 problem instances corresponding to realistic search scenarios, we cannot be sure that we covered a large enough sample of cases. The results might be influenced by the choice of problem instances.

Sixth, in SAR Optimizer, Monte Carlo particles trajectories represent the possible drift of the search object. For a fixed set of particles and search plan, the simulator evaluates the search plan deterministically and always returns the same probability of success. However, Hexaly surrogate modeling for the external function adds stochasticity to the solving process. This makes the solver non-deterministic. For legal liability purposes, this can be seen as a limitation that needs to be mitigated. This practical limitation, however, is not specific to our approach but to all non-deterministic solvers and algorithms. Nonetheless, non-deterministic solvers (or algorithms) might produce better solutions than deterministic solvers (or algorithms). As such, non-deterministic solvers (or algorithms) should not be overlooked when developing systems. In practice, if a non-deterministic solver or algorithm provides reproducibility controls via advanced parameters or seeding, it might be necessary to set them to ensure reproducibility in case of legal action.

Finally, the use of an external simulator to evaluate candidate search plans might be seen as a limitation since it could render the results more difficult to interpret compared to a closed-form objective function. Also, the reported performance gains are partially dependent on the assumptions and evaluation logic embedded in SAR Optimizer, including particle generation, detection, and the calculation of the probability of success. Nonetheless, since simulations are used in operational decision support systems for maritime SAR, of which SAROPS (Kratzke et al., 2010) and CANSARP (which includes SAR Optimizer) (Abi-Zeid et al., 2019) are two examples, we believe that it is important to provide more formal approaches in order to optimize search plans, such as the one we propose in this paper. The approach can be seen as complementary to problem-specific heuristics embedded in current operational SAR planning systems.

7. Conclusion

This paper introduced a model-and-solve approach to optimization for maritime search planning that integrates mathematical optimization with simulation. By developing five models, we demonstrated the ability to generate feasible search plans using mathematical programming for diverse search objects and search unit types, in addition to adding constraints that enhance operational, safety, and resource efficiency. This allowed us to address key requirements of the Canadian Coast Guard not currently available in the SAR Optimizer module of CANSARP. Our evaluation across 18 realistic Canadian SAR cases highlights the impact of model flexibility regarding variable search effort, asset untasking, and adjustable search rectangle angles.

The results confirm that maritime search planning can be successfully formulated as a mathematical model without sacrificing a more precise simulation-based objective function evaluation. However, our findings suggest a necessary trade-off involving the level of model detail and expressivity:

- Complexity vs. Performance: The most complex model (model MECA) did not consistently outperform simpler ones due to increased computational overhead.
- Infeasibility risks vs. Performance: Model M frequently failed to find feasible solutions when forced to reconcile fixed effort with strict horizontal spatial deconfliction.

Model ME, using a flexible effort, emerged as the most promising solution. When using model ME, the solver achieved feasibility across all 18 instances and allowed for the deactivation of unnecessary units. Ultimately, incorporating flexible effort significantly enhances resource efficiency by saving hundreds of nautical miles of search effort. Under 5 min of solving time, this flexibility sometimes leads to a lower mean probability of success than the model with fixed effort. However, the results for longer solving times indicate that incorporating flexible effort can yield better solutions on several instances.

Integrating general-purpose solvers into maritime decision support systems offers a double advantage. For developers, it provides a flexible modeling framework and a robust benchmark for comparing specialized algorithms against state-of-the-art optimization solvers. For SAR mission coordinators in an operational context, it provides high-quality search plans that respect complex safety constraints while eliminating resource waste. Ultimately, this approach ensures that every search effort unit is used effectively, maximizing the potential for life-saving operations.

CRediT authorship contribution statement

Amirhossein Esmaeilpour: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Michael Morin:** Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Funding acquisition, Formal analysis, Conceptualization. **Irène Abi-Zeid:** Writing – review & editing, Writing – original draft, Supervision, Resources, Project administration, Methodology, Funding acquisition, Formal analysis, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This project was funded by the Natural Sciences and Engineering Research Council of Canada (NSERC) [grants RGPIN-2021-03495 and DGECR-2021-00189].

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.ssci.2026.107406.

Data availability

The data that has been used is confidential.

References

Abdi, H., Williams, L.J., 2010. Principal component analysis. Wiley Interdiscip. Rev.: Comput. Stat. 2 (4), 433–459. http://dx.doi.org/10.1002/wics.101.

Abi-Zeid, I., 2026a. Private communication with an experienced search mission coordinator (JRCC, Halifax, Canada). Data obtained through private communication, Apr 22, 2026.

Abi-Zeid, I., 2026b. Private communication with an experienced search mission coordinator (JRCC, Halifax, Canada). Data obtained through private communication, Jul 21, 2026.

Abi-Zeid, I., Frost, J.R., 2005. SARPlan: A decision support system for Canadian search and rescue operations. European J. Oper. Res. 162 (3), 630–653. http://dx.doi.org/10.1016/j.ejor.2003.10.029.

Abi-Zeid, I., Lamontagne, L., 2003. A Functional Decomposition Model of Case Prosecution in Joint Rescue Coordination Centers and Related Projects: Technical Report TR 2000-150. Valcartier, Canada: Defence R&D Canada.

Abi-Zeid, I., Morin, M., Nilo, O., 2019. Decision Support for Planning Maritime Search and Rescue Operations in Canada. In: Filipe, J., Smialek, M., Brodsky, A., Hammoudi, S. (Eds.), Proceedings of International Conference on Enterprise Information Systems. vol. 1, SciTePress, Heraklion, Greece, pp. 316–327. http://dx.doi.org/10.5220/0007730303280339.

Alarie, S., Audet, C., Gheribi, A.E., Kokkolaras, M., Le Digabel, S., 2021. Two decades of blackbox optimization applications. EURO J. Comput. Optim. 9, 100011. http://dx.doi.org/10.1016/j.ejco.2021.100011.

Audet, C., Hare, W., 2017. Derivative-Free and Blackbox Optimization. Springer Cham, http://dx.doi.org/10.1007/978-3-319-68913-5.

Breivik, O., Allen, A.A., 2008. An operational search and rescue model for the Norwegian Sea and the North Sea. J. Mar. Syst. 69 (1–2), 99–113. http://dx.doi.org/10.1016/j.jmarsys.2007.02.010, Publisher: Elsevier.

Canadian Coast Guard, 2023. Maritime search and rescue (SAR) in Canada. URL: https://www.ccg-gcc.gc.ca/publications/search-rescue-recherche-sauvetage/sar-canada-res-eng.html, Last modified: May 12, 2023. (Accessed 22 May 2024).

Cucinelli, J., Goerlandt, F., Pelot, R., 2023. Exploring risk governance deficits of maritime Search and Rescue in Canada. Mar. Policy 149, 105511. http://dx.doi.org/10.1016/j.marpol.2023.105511.

Discenza, J.H., 1978. Optimal Search with Multiple Rectangular Search Areas. New York University, Graduate School of Business Administration.

Fernandes, I., Goerlandt, F., Mehr, M.Z., Brown, R., Pelot, R., 2026. A discrete event simulation model approach for probabilistic estimates of helicopter search and rescue times to maritime incidents in the Canadian Arctic. Saf. Sci. 197, 107113. http://dx.doi.org/10.1016/j.ssci.2026.107113.

Fisheries and Oceans Canada, 2017. Evaluation of the Search and Rescue Services Program. Technical Report, Government of Canada, URL: https://publications.gc.ca/collections/collection_2023/mpo-dfo/Fs23-654-2017-1-eng.pdf.

Fu, M.C., et al., 2015. Handbook of simulation optimization. vol. 216, Springer, http://dx.doi.org/10.1007/978-1-4939-1384-8.

Guo, Y., Ye, Y., Yang, Q., Yang, K., 2019. A multi-objective INLP model of sustainable resource allocation for long-range maritime search and rescue. Sustainability 11 (3), 929. http://dx.doi.org/10.3390/su11030929.

Gutmann, H.-M., 2001. A radial basis function method for global optimization. J. Global Optim. 19 (3), 201–227. http://dx.doi.org/10.1023/A:1011255519438.

Hexaly, 2021. Constrained black-box optimization. URL: https://www.hexaly.com/tutorial/solving-constrained-black-box-optimization-problems.

International Maritime Organization, International Civil Aviation Organization, 2019. International Aeronautical and Maritime Search and Rescue Manual (IAMSAR). London, UK and Montréal, Qc, Canada.
Jeong, H., Kim, C.-K., Hong, S.-Y., Yun, J.-H., Kim, D.-Y., Kim, Y.-H., 2026. Genetic-algorithm-based heatmap optimization of maritime search and rescue unit deployment considering time-step dynamics. Ocean Eng. 347, 124029. http://dx.doi.org/10.1016/j.oceaneng.2025.124029.
Karatas, M., 2021. A dynamic multi-objective location-allocation model for search and rescue assets. European J. Oper. Res. 288 (2), 620–633. http://dx.doi.org/10.1016/j.ejor.2020.06.003.
Krajewska, P., 2021. Modern equipment of the Polish SAR service during a real rescue operation in the Baltic Sea taking Into account the recommendations of the 3-volume IAMSAR manual. TransNav: Int. J. Mar. Navig. Saf. Sea Transp. 15, http://dx.doi.org/10.12716/1001.15.01.20.
Kratzke, T.M., Stone, L.D., Frost, J.R., 2010. Search and rescue optimal planning system. In: 2010 13th International Conference on Information Fusion. IEEE, pp. 1–8. http://dx.doi.org/10.1109/ICIF.2010.5712114.
Laperrière-Robillard, T., Morin, M., Abi-Zeid, I., 2022. Supervised learning for maritime search operations: An artificial intelligence approach to search efficiency evaluation. Expert Syst. Appl. 206, 117857. http://dx.doi.org/10.1016/j.eswa.2022.117857.
Lengyel, E., 2011. Mathematics for 3D Game Programming and Computer Graphics. Course Technology Press.
Malyszko, M., 2021. Fuzzy logic in selection of maritime search and rescue units. Appl. Sci. 12 (1), 21. http://dx.doi.org/10.3390/app12010021.
National Defence, Fisheries and Oceans Canada, 2014. Canadian Aeronautical and Maritime Search and Rescue Manual (CAMSAR)(B-GA-209-001/FP-001 DFO 5449, Combined Edition–Vol. I, II and III). URL: https://ccga-pacific.org/training/manuals/CAMSAR-2014-english-signed.pdf.
Nordström, J., Goerlandt, F., Sarsama, J., Leppänen, P., Nissilä, M., Ruponen, P., Lübcke, T., Sonninen, S., 2016. Vessel TRIAGE: A method for assessing and communicating the safety status of vessels in maritime distress situations. Saf. Sci. 85, 117–129. http://dx.doi.org/10.1016/j.ssci.2016.01.003.
O'Rourke, J., 1998. Search and intersection. In: Computational Geometry in C. Cambridge University Press, pp. 220–293. http://dx.doi.org/10.1017/CBO9780511804120.
Richardson, H.R., Discenza, J.H., 1980. The United States Coast Guard Computer-Assisted Search Planning system (CASP). Nav. Res. Logist. Q. 27 (4), 659–680. http://dx.doi.org/10.1002/nav.3800270413, (Accessed 2016-02-11).
Sezer, S.I., Akyuz, E., 2025. An integrated CREAM and Dempster–Shafer (DS) evidence theory for improving human reliability in maritime search and rescue (SAR). Ocean Eng. 340, 122241. http://dx.doi.org/10.1016/j.oceaneng.2025.122241.
Stone, L.D., 1983. The process of search planning: current approaches and continuing problems. Oper. Res. 31 (2), 207–233. http://dx.doi.org/10.1287/opre.31.2.207.
Stone, L.D., Royset, J.O., Washburn, A.R., et al., 2016. Optimal search for moving targets. vol. 237, Springer, http://dx.doi.org/10.1007/978-3-319-26899-6.
Tenaud, E., 2022. Optimisation de fonctions boîtes noires avec et sans contraintes. In: 23ème congrès annuel de la Société Française de Recherche Opérationnelle et d'Aide à la Décision.
Tenaud, E., 2024. A shallow dive into hexaly black-box optimization solver. URL: https://roadef.org/jfro/static/files/slides_446.pdf, Presentation at the 44th Greater Paris Operations Research Day (JFRO 44).
Trummel, K., Weisinger, J., 1986. The complexity of the optimal searcher path problem. Oper. Res. 34 (2), 324–327. http://dx.doi.org/10.1287/opre.34.2.324.
Wang, Y., Qu, X., Li, Z., 2025. Optimizing pre-occurrence maritime search and rescue system: A dynamic location-allocation model with considering spatiotemporal accessibility. Ocean Eng. 337, 121964. http://dx.doi.org/10.1016/j.oceaneng.2025.121964.
Wu, J., Cheng, L., Chu, S., Song, Y., 2024. An autonomous coverage path planning algorithm for maritime search and rescue of persons-in-water based on deep reinforcement learning. Ocean Eng. 291, 116403. http://dx.doi.org/10.1016/j.oceaneng.2023.116403.
Xiong, P., Hu, L., Yongliang, T., Zikun, C., Bin, W., Hao, Y., 2021. Helicopter maritime search area planning based on a minimum bounding rectangle and K-means clustering. Chin. J. Aeronaut. 34 (2), 554–562. http://dx.doi.org/10.1016/j.cja.2020.08.047.
Xiong, P., Liu, H., Tian, Y., Chen, Z., 2020. A time domain-based iterative method for helicopter maritime search area planning and construction of the simulation environment. IEEE Access 8, 191460–191471. http://dx.doi.org/10.1109/ACCESS.2020.3032583.
Zhang, C., Huang, N., Wu, C., 2025. Autonomous coverage path planning model for maritime search and rescue with UAV application. J. Mar. Sci. Eng. 13 (9), 1735. http://dx.doi.org/10.3390/jmse13091735.
Zhao, M., Pan, J., Wang, Q., Li, J., 2024. Collaborative scheduling of mass rescue operations at sea. J. Mar. Sci. Eng. 12 (11), 2060. http://dx.doi.org/10.3390/jmse12112060.
Zhou, X., Cheng, L., Min, K., Zuo, X., Yan, Z., Ruan, X., Chu, S., Li, M., 2020. A framework for assessing the capability of maritime search and rescue in the South China Sea. Int. J. Disaster Risk Reduct. 101568. http://dx.doi.org/10.1016/j.ijdrr.2020.101568.