Context-Aware Dynamic Asset Allocation for Maritime Surveillance Operations

LINGYI ZHANG
DAVID SIDOTI
GOPI VINOD AVVARI
DIEGO F. M. AYALA
MANISHA MISHRA
DAVID L. KELLMEYER
JAMES A. HANSEN
KRISHNA R. PATTIPATI

This paper formulates and solves a maritime surveillance problem involving the allocation of *multiple* heterogeneous assets over a large area of responsibility to detect *multiple* drug smugglers using *heterogeneous* types of transportation on the sea with varying contraband weights. The asset allocation is based on a probability of activity surface, which represents spatiotemporal target activity obtained by integrating intelligence data on drug smugglers' whereabouts/waypoints for contraband transportation, their behavior models, and meteorological and oceanographic information. A number of algorithmic concepts based on branch-and-cut with limited search and approximate dynamic programming (ADP) were investigated. We validate the proposed algorithmic concepts via realistic mission scenarios. We conduct scalability analyses of the algorithms and conclude that effective asset allocations can be obtained within seconds using rollout-based ADP. The contributions of this paper have been transitioned to and are currently being tested by Joint Interagency Task Force—South, an organization tasked with providing the initial line of defense against drug trafficking in the East Pacific Ocean and the Caribbean Sea.

Manuscript received February 23, 2019; revised June 11, 2019 and September 26, 2019; released for publication June 30, 2020.

Refereeing of this contribution was handled by Ramona Georgescu.

L. Zhang, D. F. M. Ayala, and K. R. Pattipati are with the Department of Electrical and Computer Engineering, University of Connecticut, Storrs, CT 06269, USA (E-mail: lingyi.zhang@uconn.edu, krishna.pattipati@uconn.edu).
D. Sidoti and J. A. Hansen are with the U.S. Naval Research Laboratory (NRL-MRY), Monterey, CA 93943, USA (E-mail: david.sidoti@nrlmry.navy.mil, james.hansen@nrlmry.navy.mil).
G. V. Avvari and M. Mishra are with Delphi Technologies, Kokomo, IN 46902, USA.
D. L. Kellmeyer is with the Naval Information Warfare Center (NIWC)—Pacific, San Diego, CA 92152, USA (E-mail: dave@spawar.navy.mil).
This work was supported in part by the U.S. Office of Naval Research and Naval Research Laboratory under Grant N00014-18-1-2838, Grant N00014-16-1-2036, Grant N00173-16-1-G905, and Grant HPCM034125HQU.

1557-6418/20/$17.00 © 2020 JAIF

I. INTRODUCTION

A. Motivation

The illicit drug trade is an extremely profitable industry and it is estimated that the consumers in the United States alone spend as much as 150 billion USD per year on black market drugs. Of this, it is estimated that 37 billion USD is spent on cocaine alone. It is a problem of national, and increasingly international, concern [1], [2]. This problem increased exponentially with the advent of narco-terrorism and the prospect of terrorists using narcotics smuggling techniques to transport terrorists or weapons of mass destruction into the country. Given the reduction in the national resources allocated to the counter-narcotics threat, it is of paramount importance that smarter and faster decision support tools that integrate a wide variety of information are developed to assist in this challenge of using less to accomplish more. To do so requires effective hybrid human–machine systems.

The U.S. Navy has shown a growing interest in mixed-initiative human–machine systems and mastering information dominance for effective context-driven operations [3]. To do so requires the transfer of the right data from the right sources in the right context to the right decision maker (DM) at the right time for the right purpose—a concept known as 6R [4]. If a dynamically developing operational context can be understood by the DM, appropriate courses of action (COAs) can be carried out, given the unfolding events. In the context of maritime operations, DMs must assimilate information from a multitude of sources before making decisions on the strategy to be followed each day. If the DMs are better informed about what to expect given the currently accessible data, as well as what they might expect in the case of unforeseen events, effective decisions can be made on the COAs.

Currently, much planning for narcotics seizures is performed by humans interpreting large amounts of data, including weather forecasts, intelligence, and recently reported contacts of interest. Each day, the targeting analysts must process and interpret all of these data and agree upon a COA amounting to where limited detection aircraft and interdiction vessels should be allocated to disrupt the maximum amount of shipments over a multiday planning cycle. The consolidation of large amounts of data and possible strategies into a single asset allocation optimizer is beneficial for both algorithmic purposes and human understanding. To support this transition to a human–machine

collaborative mode of operation, we have developed an optimization-based modeling framework and the associated decision support software tool for dynamic surveillance and interdiction resource management in counter-smuggling operations. This tool, named COAST or Courses Of Action Simulation Tool [5], and the corresponding algorithms are intended to support targeting analysts in identifying high-probability areas of smuggler presence and to proactively develop a set of high-value COAs.

The counter-smuggling problem presented in this paper is viewed as a moving horizon stochastic control problem, as illustrated in Fig. 1, specifically from a strategic operations standpoint, i.e., decision making with regard to a schedule to follow for the upcoming time horizon. Here, each block is an entity, such as a DM, sensor, or asset, and the link from each block represents the outcome of the block and its impact or influence on the next block. The problem can be decomposed into surveillance and interdiction asset allocation subproblems. This paper focuses on the surveillance component, where DMs (also termed targeteers) are responsible for allocating multiple aircraft (namely, P-3 Orions (manned)) over a finite time horizon in an effort to detect the transportation of contraband. The interdiction component, detailed in [6], involves the allocation of multiple heterogeneous surface assets (namely, Navy ships, Coast Guard cutters), to disrupt multiple drug smugglers of varying types, similar to that which is addressed in this paper. The DMs in Fig. 1 choose which surveillance assets to allocate to which target(s) (smugglers) based on the target type and intelligence forecasting the target's trajectory (specified in the form of probability of activity (PoA) surfaces [7], [8]). After allocated assets attempt to search for potential targets, the mission environment changes due to any target detection that may occur or due to weather changes. These environment changes are recorded by sensors and operators, processed, and sent back to the DMs in the form of target types and tracks, and are combined into an updated PoA surface, providing a new forecast for the remainder of the planning time horizon. The process then repeats. Ideally, the results of this paper feed that of [6] for *coordinated* smuggler detection and interdiction.

B. Related Research

The surveillance mission involves the search, detection, tracking, and identification of potential smugglers within a large geographic region, which plays an essential role in the counter-smuggling operation. Airborne surveillance assets (e.g., helicopters, maritime patrol aircraft) are highly efficient at determining the sea surface traffic information. However, in a real-world scenario, there are typically a limited number of surveillance assets and a large sea surface area that needs to be surveilled. The study of how to most effectively employ limited resources to locate an object, whose location is not precisely known, falls under the rubric of search theory.

The earliest foundations of search theory were built by Koopman [9] to aid the U.S. Navy in efficiently locating enemy submarines during World War II, which was further generalized in [10]. There are two major categories of search theory: 1) the optimal allocation of effort problem and 2) the best track problem [11]. For the optimal effort allocation problem, Blachman and Proschan [12] derived an optimum search pattern for a generalized problem of finding an object in one of the n boxes. Pollock [13] introduced a Bayesian approach to the optimal allocation problem, where allocation decisions are made sequentially based on observations up to the current time in order to minimize the expected cost of searching to satisfy a specified probability of detection (PD). Charnes and Cooper [14] applied convex programming, along with the Kuhn–Tucker conditions, for the optimum distribution of effort computation. In this paper, we adopt Charnes and Cooper's method to compute the effort required for the optimal search in a discretized map.

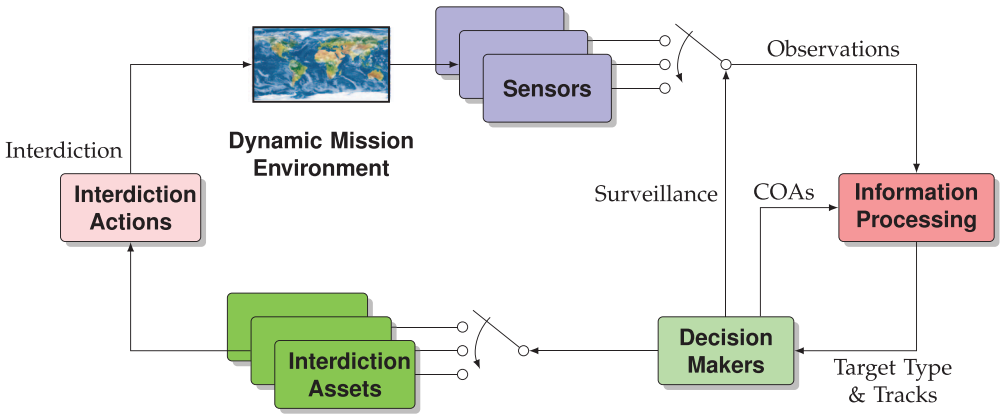


Fig. 1. The counter-smuggling problem viewed from a stochastic control standpoint. Targeteers (DMs) choose from a set of available surveillance assets and finalize a search schedule to allocate the asset(s) over a near-time planning horizon, typically 72 h. Similar to the planning/decision process presented in [6], [7], [32], and [40], after the action is carried out, information is gathered, processed, and fed back to the targeteer.

Stone [15] made use of the calculus of variations, convexity properties, and generalized Lagrange multiplier techniques to formulate a systematic treatment of search theory. For the best track problem, Lukka [16] worked out the theory of optimal search for stationary targets, targets whose motion is known, and targets whose motion is almost known. The method relies on the theory of optimal control. Mangel [17] extended Lukka's algorithms with the option of incorporating a detection rate that is either independent of or dependent on velocity.

In recent years, the problem of drug surveillance has been formulated from a variety of viewpoints. For example, Washburn and Wood [18] formulated the surveillance problem as a two-person zero-sum game and Pfeiff [19] applied search theory to a defender–attacker optimization model that maximizes the defender's probability of success. Royset and Wood approach the problem as a network flow problem, wherein an interdictor must destroy a set of arcs on a network to minimize both the interdiction cost and the maximum flow of smugglers [20]. Jacobson et al. [21] formulate the problem as a multiple traveling salesman problem with the objective of minimizing the overall search route cost for multiple platforms that visit every search location. Ng and Sancho [22] developed a dynamic programming method to solve the surveillance problem. However, the dynamic programming approach suffers from the curse of dimensionality for large problems and, consequently, near-optimal approximations are needed. A common way to overcome this curse is by approaching the problem via approximate dynamic programming (ADP) with policy iteration as in [23], where they frame the problem in terms of stochastic control with partially observable Markov decision processes. Kress et al. [24] examine a discrete-time and discrete-space stochastic dynamic programming approach to coordinate the efforts of a single aerial search asset and a single surface interdiction asset. Other approaches, including the formulation of the surveillance problem as a resource-dependent orienteering problem [25]–[27], wherein reward depends on the resource expended at each visited node, have been investigated.

Optimal search problem formulations have become versatile in their ability to account for multiple cooperating searchers, multiple targets with different characteristics, and environmental effects on the search [28]–[31]. For example, arc inspection is based on the inverse of the probabilities of detection as opposed to PoA surfaces accounting for weather and intelligence in [7], [8], and [32]. Byers [33] extended the network modeling approach to drug interdiction by including Bayesian updating of the PoA surface. He considered a scenario with one unmanned aerial vehicle and one ground-based interceptor to interdict multiple targets with different deadlines. Bessman [34] developed a defender–attacker optimization model that uses the PoA surfaces as the basis for asset allocation against smugglers. He formulated a stochastic shortest path problem and represented smuggler behavior as the output of an all-to-one label-correcting temporal dependence instead of one-step dependence. Three different sensor types (one interdiction and two surveillance) are considered for allocation to prosecute one type of target (among three possible). In this defender–attacker model, smugglers are assumed to have imperfect knowledge of possible sensor locations and are given the ability to modify their behavior in response to this information.

C. Paper Organization

Similar to Pietz and Royset [25], we also discretized our maritime map. We adopt Charnes and Cooper's method [14] to compute the effort required for optimal search in a discretized map. Our novel algorithmic contributions are the following:

1) Fast one- and two-step lookahead ADP (1SLADP and 2SLADP) algorithms for maritime surveillance composed of heterogeneous assets and heterogeneous targets, each of which is carrying not necessarily the same amount of contraband. Our algorithms exploit the fusion of intelligence and weather information available in the PoA surfaces.
2) We measure the utility of our approach by way of comparison with more traditional branch-and-cut (B&C) algorithms to solve the surveillance problems. We develop two variations of the ADP-based surveillance asset allocation algorithms, wherein real-world constraints on the assets (e.g., endurance and rest time) are explicitly considered.

The paper is organized as follows. Section II describes the problem and the technical challenges addressed in the development of allocation algorithms underlying our decision support tool. In Section III, we discuss solution approaches, including exhaustive and greedy B&C and ADP. In Section IV, we present simulation results as applied to a benchmark scenario that has multiple targets, multiple surveillance assets, and parameters that have multiple levels of uncertainty. We additionally conduct and present results from our sensitivity analysis relating to the scalability and performance of our solution approaches in a realistic mission scenario. We conclude the paper in Section V with a summary of our findings and future work.

II. PROBLEM MODEL AND FORMULATION

A. Problem Definition and Solution Architecture

The complete maritime surveillance and interdiction problem is one of maritime drug trafficking disruption in the East Pacific Ocean and the Caribbean Sea. The general mission consists of two components: 1) surveillance (the detection, tracking, and identification of contacts of interest) and 2) interdiction (the interception, investigation, and potential apprehension and repatri-

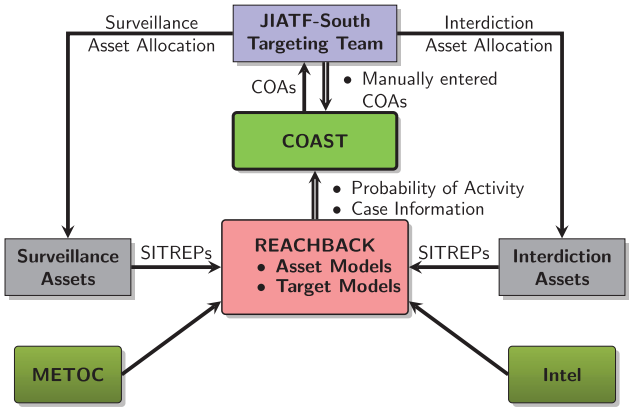


Fig. 2. Information flow and decisions (controls) in the counter-smuggling problem. The decision support tool, COAST, provides COAs to the JIATF-South Targeting Team who then modify them as they see fit. The manually entered COAs can then be fed back into the tool where the simulation is rerun providing new outcomes to the targeting team, who can then provide further feedback and modifications, if necessary.

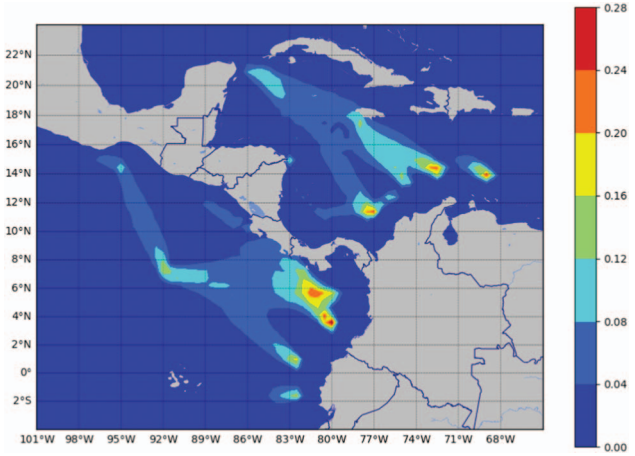


Fig. 3. PoA surface $\mathrm{PoA}(q, k, j)$ summed over all k.

ation of smugglers). In response to the need for information fusion, we proposed a decision support system (DSS) in [5], named COAST, to host and utilize algorithms to provide auxiliary support to Joint Interagency Task Force—South (JIATF-South) targeteers. We proposed different forms of visualizations to enable DMs to understand the behavior of our algorithms and the presently evolving context, while also providing functionality for human input and interaction in order to effectively integrate both humans and decision support algorithms for mixed-initiative planning. The information flow for the complete maritime interdiction problem is illustrated in Fig. 2.

In COAST, we solve a moving horizon dynamic resource management problem for both surveillance and interdiction operations based on user-defined mission parameters. We then provide suggested COAs that the DMs can interact with, adjust, and fine-tune to analyze various "what–if" scenarios and to obtain a satisfactory allocation. Visual and computational analytics are provided to communicate the reasons behind our algorithm's behavior. From Fig. 2, continuously updated PoA surfaces (see Fig. 3 for an example), representing the posterior probabilities of smugglers' presence, constitute the sufficient statistics for decision making [35]—that is, COAST does not need to know how specific Intel or meteorology and oceanography (METOC) features, for example, uncertainty associated with a drug trafficker, wave heights, currents, etc., and how these two inputs, along with asset and target models, are combined to produce the PoA surface. A targeteer can fine-tune the allocations, the resulting COAs are executed, and observations from surveillance and interdiction assets are sent back to the reachback cell in the form of situational reports or SITREPs (e.g., detections or nondetections) that are used to update the PoAs. The targeteer can specify multiple objective functions. The objectives considered and analyzed in this paper are as follows:

O1: Maximize the normalized weight of the contraband detected (normalized by the total possible amount of contraband).

O2: Maximize the normalized number of detections (normalized by the total possible number of cases).

O3: Maximize the normalized number of smugglers detected (normalized by the total possible number of smugglers).

Let α_j and ρ_j denote the expected contraband weight and expected number of smugglers for case j. Let C be the total number of cases (i.e., predicted smuggler tracks) to be searched. Then, the normalized priority weights for objectives O1–O3, respectively, are as follows:

$$\lambda_j = \frac{\alpha_j}{\sum_{g=1}^{C} \alpha_g}, \tag{1}$$

$$\lambda_j = \frac{1}{C}, \tag{2}$$

$$\lambda_j = \frac{\rho_j}{\sum_{g=1}^{C} \rho_g}. \tag{3}$$

B. Problem Formulation

The notation used in this paper is listed in Table I.

1) PoA Surface: The foundation for each asset allocation solution is the PoA surface over multiple time epochs. The PoA surface is the result of combining METOC information with actionable intelligence with regard to uncertain smuggler departure point(s), departure times, waypoint(s), destination(s), and their behavior on the ocean. The spatiotemporal probability surface, PoA, is calculated as the joint probability of two discrete random events: 1) the case j, with a corresponding binary random variable C_j, i.e., how trustworthy the intelligence source is regarding a target, and 2) the target correspond-

TABLE I
Summary of Notations

A	Total number of surveillance assets
$\mathcal{A}_j$	Total area to be searched for case j
B_{ij}	Great Circle distance from the base of asset i to the centroid of case j
C	Total number of cases
CPoSD(i, j)	Cumulative probability of successful detection for a given asset i allocated to case j
$d_{i\ell}$	Landing time for asset i's ℓth flight
i	Surveillance asset index
j	Case index
k	Time epoch index
K	End of planning time horizon
L_i	Endurance of asset i
PoA(q, k, j)	Likelihood that a smuggler belonging to case j is located in a cell q at time k
R_i	Downtime of asset i
$s_{i\ell}$	Remaining search time available within the current sortie for asset i
S_{ij}	Sweep width of asset i searching for target j
t_{ij}	Travel time for traversing the distance B_{ij}
v_i^{a}	Travel speed of asset i
v_i^{s}	Search speed of asset i
w_{ijk}	Reward of allocating asset i to case j at time k
x_{ijk}	Binary decision variable of allocating asset i to case j at time epoch k
λ_j	Priority weight of case j
$\tau_{i\ell}$	Departure time for asset i's ℓth flight
$\gamma(i, j, \ell)$	The set of search time indices for asset i assigned to case j for the ℓth flight

ing to case j at a location q at time epoch k, with a corresponding binary random variable $\mathcal{X}(q, k, j)$, i.e., given that the case j exists, the probability that the target exists at a location q at time k. The probability surface PoA is indexed by a location q, time k, and case j, and is defined in (4)–(7):

$$\mathrm{PoA}(q, k, j) = P\left(\mathcal{C}_j = 1 \cap \mathcal{X}\left(q, k, j\right) = 1\right) \tag{4}$$

$$= E\left\{\mathcal{C}_j \cdot \mathcal{X}\left(q, k, j\right)\right\} \tag{5}$$

$$= E_{\mathcal{C}_j}\left\{\mathcal{C}_j \cdot E_{\mathcal{X}(q,k,j)|\mathcal{C}_j}\left(\mathcal{X}\left(q, k, j\right)|\mathcal{C}_j\right)\right\} \tag{6}$$

$$= \sum_{c_j=\{0,1\}} c_j \cdot P(\mathcal{C}_j = c_j) \cdot \left(\sum_{h=\{0,1\}} h \cdot P(\mathcal{X}\left(q, k, j\right) = h|\mathcal{C}_j = c)\right), \tag{7}$$

where we separate the expectation in (6) based on the law of total expectation/iterated expectations.

We assume that $P(\mathcal{C}_j = 1) = 1$, that is, the intelligence sources are always correct with 100% certainty. Then, (7) reduces to

$$\mathrm{PoA}(q, k, j) = P\left(\mathcal{X}\left(q, k, j\right) = 1\right). \tag{8}$$

Therefore, PoA(q, k, j) is a number that refers to the likelihood that a smuggler, belonging to case j, is located in a cell q at time k. The PoA surfaces are computed as detailed in [8] and represent all the relevant information for effective asset allocation. The DM can specify how many planning epochs to optimize over based on these PoA surfaces and the objective function to be optimized. A typical PoA surface PoA(q, k, j), summed over all k, is shown in Fig. 3.

2) Optimal Search Effort Calculation: We assume the optimum distribution of search effort is known based on the model in [14]. Let p_{jkq} denote the PoA of target j in cell q at time k. We first rank the nonzero PoA cells in decreasing order such that $p_{jk[1]} \geq p_{jk[2]} \geq \cdots$, where $[\kappa]$ denotes the κth largest nonzero PoA cell. Let the total available effort to be expended by asset i to search case j be Φ_{ij}. A critical threshold is then calculated to narrow the problem space and eliminate PoA cells not worth searching, by first finding an n that satisfies the following inequality [14]:

$$\sum_{v=1}^{n} v\left[\ln p_{jk[v]} - \ln p_{jk[v+1]}\right] > \Phi_{ij}. \tag{9}$$

Then, the critical probability, ρ_{ijk}, corresponding to the search of case j by asset i at time k, is as in (10):

$$\rho_{ijk} = \frac{1}{n}\left(\sum_{v=1}^{n} v\left(\ln p_{jk[v]} - \ln p_{jk[v+1]}\right) - \Phi_{ij}\right) + \ln p_{jk[n+1]}. \tag{10}$$

We then select all the cells corresponding to case j that have a PoA greater than the critical probability found in (10). This reduces the number of potential cells that need to be searched for each case j by asset i. We then compute the patrol box that maximally covers the high-probability cells for each case. The allocation of assets to patrol boxes is the subject of the optimization problem discussed next.

3) Optimization Problem: The case regions are labeled by aggregating the PoA surfaces over a discrete planning time period of length K (e.g., 72 h). Let us assume a moving horizon frame of reference, where $k = 0$ corresponds to the current time period of unit length ($\Delta = 1$ h), $k = 1$ corresponds to the first planning period, and $k = K$ corresponds to the final period to be planned for. Let A be the total number of surveillance assets, C be the total number of cases, and $q \in Q(j)$ be the set of cells in the patrol box for case j as determined by the optimal search effort calculation algorithm. The size of the patrol box depends on the concept of operations and is assumed known. Let w_{ijk} be the probability of successful detection (PoSD), which is the product of the PoA surface and the PD when asset i is assigned to

search for case j at time k. That is,

$$w_{ijk} = \sum_{q \in Q(j)} \text{PoA}(q, k, j)\text{PD}(i, j, k), \tag{11}$$

where

$$\text{PD}(i, j, k) = 1 - \exp\left(-\frac{S_{ijk} v_i^{\text{s}} \Delta}{\mathcal{A}_j}\right) \tag{12}$$

is the probability that asset i detects case j during the kth time epoch interval (PD(i, j, k) can only be collected at the end of the kth time epoch interval). Let us assume that each asset travels to the search region at a speed v_i^{a} and searches in the search region at a speed v_i^{s}. The PD equation is adopted from Koopman's random search formula [9], and offers a lower bound on the PD; advanced models may be used in place of (12) as in [15]. Here, S_{ijk} is the sweep width of asset i searching for case j at time epoch k, and Δ is the interepoch interval (=1 h in this paper).

Let B_{ij} represent the geodesic[1] distance that asset i must traverse from its base to the centroid of case j. The time it takes to traverse B_{ij}, denoted by t_{ij}, is given by

$$t_{ij} = \left\lceil \frac{B_{ij}}{v_i^{\text{a}}} \right\rceil, \tag{13}$$

where $\lceil \cdot \rceil$ denotes the ceiling or rounding up to the nearest integer. Let $\tau_{i\ell}$ denote the departure time if an asset i is allocated to a case for flight ℓ, and $d_{i\ell}$ as the landing time upon its return from the corresponding search box. The index ℓ increments with each flight that asset i is scheduled to fly over the planning time horizon. Formally,

$$\tau_{i\ell} = \begin{cases} k, 0 < k \leq K, & \text{if i is assigned to a case during} \\ & \quad \text{the } \ell^{\text{th}} \text{ flight,} \\ \infty, & \text{otherwise.} \end{cases} \tag{14}$$

A similar definition applies to $d_{i\ell}$. For each flight, the total search and travel time for each asset from its corresponding base to each case must not exceed the asset's endurance, L_i (in hours), and, upon flight completion, it must rest for R_i consecutive hours before it can be scheduled to depart for the next search box. The assets are assumed to be manned aircraft with an associated rest time for the pilot; additionally, each aircraft requires periodic maintenance and refueling. The minimum time it may take for an asset to become available again for search is $L_i + R_i$. Note that there is no feasible asset allocation for a case j and asset i if $2t_{ij} \geq L_i$; i.e., the total round trip travel time for a search region is greater than the maximum aloft time L_i. With PoSD defined as in (11), the cumulative probability of successful detection (CPoSD) for a given asset i is

$$\text{CPoSD}(i, j) = 1 - \prod_{k=1}^{K} \left(1 - w_{ijk} x_{ijk}\right), \tag{15}$$

where x_{ijk} is a binary decision variable such that $x_{ijk} = 1$ if asset i is assigned to case j at time epoch k, and 0, otherwise. The total reward that asset i can collect over the planning time horizon is then

$$r_i = \sum_j \lambda_j \text{CPoSD}(i, j), \tag{16}$$

where λ_j is the normalized priority weight of case j. We wish to solve the following problem:

$$\max_{x_{ijk}, \tau_{i\ell}, d_{i\ell}} J = \max \sum_{i=1}^{A} r_i, \tag{17}$$

$$\text{s.t.} \sum_i x_{ijk} \leq 1 \quad \forall j, k, \tag{18}$$

$$\sum_j x_{ijk} \leq 1 \quad \forall i, k, \tag{19}$$

$$d_{i\ell} - \tau_{i\ell} \leq L_i \quad \forall i, \ell, \tag{20}$$

$$\tau_{i\ell+1} - d_{i\ell} \geq R_i \quad \forall i, \ell, \tag{21}$$

$$\tau_{i\ell}, d_{i\ell} \in \{0, \ldots, K\} \cup \{\infty\} \quad \forall i, \ell, \tag{22}$$

$$x_{ijk} \in \{0, 1\}. \tag{23}$$

In (17), we assume that the surveillance asset cannot detect targets while it is en route to the patrol box. Constraints (18) and (19) ensure that no more than one case is allocated to an asset at one time. Constraint (20) indicates that the maximum asset aloft time must not exceed L_i. Constraint (21) ensures that there must be a minimum downtime of R_i between asset allocations for a particular asset i and that subsequent allocations must have a departure time later than the previous one(s), if any. The problem posed in (17)–(23) is NP-hard [36].

III. SOLUTION APPROACH

A. Exhaustive Branch-and-Cut

The first solution approach we investigated is the exhaustive B&C method, herein referred to as E-B&C. This method involves the enumeration and evaluation of all feasible solutions and is illustrated in Fig. 4 with respect to asset i, where each completed branch is a feasible solution with the corresponding asset–case assignments $\langle i, j^* \rangle$, given the departure times $\tau_{i\ell}$, $\ell = 1, 2, \ldots$, for each flight of asset i. We enumerate a complete feasible flight schedule over all flights for each asset i and calculate the total reward r_i for a given asset i using (15) and (16). The schedule with the highest r_i is selected to

[1]The geodesic distance is the shortest distance between two points on the surface of a sphere.

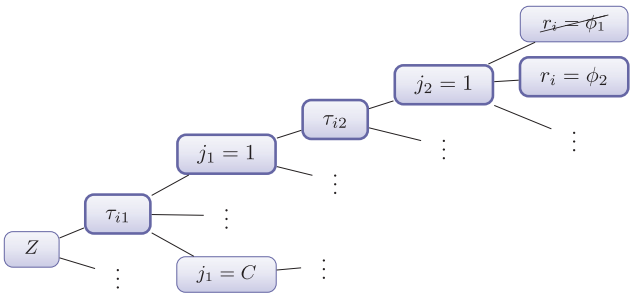


Fig. 4. Branching method with τ_{i1} and τ_{i2} being the departure time for the first and second flights and the corresponding case assignments j_1 and j_2. r_i is evaluated using (15) and (16) for each completed branch. The highest r_i is then saved as the best assignment for asset i.

be the best assignment for asset i. In order to find the optimal allocation, we repeat the process mentioned earlier with the full permutation of asset–case combinations. The pseudocode is shown in Algorithm 1. In Algorithm 1's pseudocode, line 1 generates the permutation of the ordering of assets for which to start the allocation. Lines 2–5 compute the best assignment for the selected asset i using B&C and updates the PoA surface accordingly to avoid duplicate assignments (this is done by setting the allocated grid cells in the PoA surface to have no reward during the assigned search time(s)). Line 7 saves all the assignment for each asset sequence generated by the permutation function. Line 8 resets the PoA surface to the originally initialized surface prior to any updates in order to compute the next sequence generated by the permutation function.

ALGORITHM 1 Exhaustive Branch-and-Cut (E-B&C)

1: PermSeq = Perm$(1, \ldots, A)$ ▷ Permutation of ordering of assets for which to start allocation
2: **for each** AssetSeq in PermSeq **do**
3: **for each** $i \in$ AssetSeq **do**
4: assign(i) = B&C(i)
5: updatePoA(assign(i)) ▷ Prevent overlap of patrol box assignments
6: **end for**
7: PotentialAssign ← PotentialAssign + assign ▷ Save potential assignment given we allocated in order AssetSequence
8: resetPoA ▷ Set PoA to originally initialized surface prior to any updates
9: **end for**
10: BestAssign = MaxReward(PotentialAssign) ▷ For all potential assignments found, search and find that which resulted in the maximum reward

B. Greedy Branch-and-Cut I

Similar to E-B&C, we repeat the asset allocation process for all the available assets and fix the assignment for an asset i^* with the highest r_i. After the asset–case–time epoch assignment is fixed, we update the PoA surface to ensure that the assigned cases are no longer available for additional scheduling during the assigned search hours. The same process is then repeated until either no more assets are available or all cases are fully allocated. We refer to this method as GB&C-I. The pseudocode is shown in Algorithm 2. In this pseudocode, line 1 states that while there are any unassigned assets, continue on to lines 2–7, where the best assignment for each unassigned asset is found using B&C. The best asset assignment is then selected in line 8 (i.e., i^* becomes known among the explored potential assignments). In lines 9–11, the PoA surface is updated given the asset assignment found.

ALGORITHM 2 Greedy Branch-and-Cut I (GB&C-I)

1: **while** length(AssignedAsset) $\leq A$ **do**
2: assign = $\emptyset$
3: **for each** $i \in \{1, \ldots, A\}$ **do**
4: **if** $i \notin$ AssignedAsset **then**
5: assign(i) = B&C(i)
6: **end if**
7: **end for**
8: assignment(i^*) = MaxReward(assign)
9: AssignedAsset ← AssignedAsset + i^*
10: BestAssign ← BestAssign + assignment(i^*)
11: updatePoA(assignment(i^*))
12: **end while**

C. Greedy Branch-and-Cut II

To reduce the runtime and problem complexity, we propose a second greedy B&C method, referred to as GB&C-II. This method is similar to the E-B&C method, except that we put an additional constraint on assets. Once we enumerate all the possible departure times and find the best assignment $\{j^*\}$ corresponding to each departure time for an asset i, we fix the corresponding schedule. That is, we reduce the complexity of search with more than one asset from permutation ordering to a linear ordering. The same process is then repeated until all cases are fully allocated or there are no more assets available. The pseudocode is shown in Algorithm 3. Here, line 2 finds the best assignment for asset i found in line 1. Line 3 updates the PoA surface and line 4 saves the best assignment found in line 2.

ALGORITHM 3 Greedy Branch-and-Cut II (GB&C-II)

1: **for each** $i \in \{1, \ldots, A\}$ **do**
2: assign(i) = B&C(i)
3: updatePoA(assignment(i))
4: BestAssign ← BestAssign + assign(i)
5: **end for**

D. Parallelized Greedy Branch-and-Cut II

To further improve the computation time, we develop a parallelized version of the GB&C-II algorithm. Parallelization involves dividing a large problem into multiple independent subproblems, where each subproblem is assigned to a processor. This substantially reduces the computation time and therefore can rapidly generate allocation solutions. We use a master–slave architecture for our parallelization with the following functionalities:

Master process

- Pools subproblems for the slave processors to run.
- Spawns the subproblems on multiple slave processors.
- Collects the results from the slave processors.

Slave process

- Receives the subproblem from the master processor.
- Executes the subproblem.
- Returns the solution to the master processor.

The serial GB&C-II algorithm, executed on a single processor, searches the B&C tree by expanding live nodes one at a time. In order to parallelize this problem on M processors, we set each τ_{i1} to each processor and let each processor execute the subproblem. All processes share the same memory for the PoA and other read-only data. Lastly, the master processor collects all value returns from the slave processors to evaluate the best assignment for asset i.

E. Approximate Dynamic Programming

Another approach to solve the problem is via ADP, more specifically, a one-step lookahead rollout algorithm. Note that the following formulation is for a single arbitrary asset i and is thus assumed given throughout. Let j_k be the asset–case assignment at time epoch k and z_k be the remaining aloft time for an asset at time epoch k. We have the state equation for z_{k+1} as

$$z_{k+1} = f(z_k, j_k), \tag{24}$$

where j_k is the state-based control variable that selects a case j at time epoch k as

$$j_k = \mu_k(z_k, j_{k-1}), \qquad j_k = 0, 1, \ldots, C. \tag{25}$$

Here, $z_k = L_i$ and $j_k = 0$ implies that there is no asset–case assignment made and the asset is in the rest state at time epoch k. When $z_k \leq L_i$ and $j_k = 1, \ldots, C$, an asset–case assignment has been made at time epoch $k-1$ and the asset is currently in a flight state. The detailed control options are described in this section later (see (31) and (32)).

The ADP equation for the problem is defined as follows:

$$g_k(z_k, j_k) = \lambda_{j_k}\left(1 - \prod_{k \in s_{j_k} | z_k} \left(1 - w_{ij_k k}\right)\right), \tag{26}$$

$$J_k(j_k) = \max_{j_k} E\left\{g_k(z_k, j_k) + \bar{J}_{k+1}\left(f(z_k, j_k, \Lambda(k))\right)\right\}, \tag{27}$$

where s_{j_k} is the set of remaining search time indices available within the current sortie for asset i assigned to case j and $\Lambda(k)$ is a function that indicates that the asset is currently flying its ℓth flight at time k. The variable λ_j is the normalized priority weight for case j. Here, $\bar{J}_{k+1}$ is the heuristic cost-to-go and is estimated based on the following assumptions:

H1: The asset will fly out for its maximum aloft time.
H2: Each asset will stay on just one case for each flight.
H3: Each asset will fly out immediately after it is fully rested.
H4: The case with the highest total reward will be selected for the ℓth flight interval, as in (28):

$$j^* = \arg\max_j \lambda_j \left(1 - \prod_{k \in \gamma(i,j,\ell)} \left(1 - w_{ijk}\right)\right), \tag{28}$$

where $\gamma(i, j, \ell)$ is the set of search time indices for asset i assigned to case j for the ℓth flight. If the planning time horizon allows multiple flights, then we first compute the best case for the next flight time defined by H1 to H3 using (28). The future cost-to-go for the ℓth flight is as follows:

$$H(\ell) = \lambda_{j^*}\left(1 - \prod_{k \in \gamma(i,j^*,\ell)} \left(1 - w_{ij^*k}\right)\right), \tag{29}$$

where j^* is computed from (28). The heuristic cost-to-go given the current flight at time k is $\Lambda(k)$ and is given by

$$\bar{J}_{k+1}\left(f(z_k, j_k, \ell_k)\right) = \sum_{n=\Lambda(k)+1}^{\lceil K/(L_i+R_i)\rceil} H(n). \tag{30}$$

As mentioned earlier, the control variable j_k is state-dependent. When an asset is at rest state at time k, the control variable j_k comprises the actions of launching the asset or not with the intent of obtaining better reward at a later time epoch. That is,

$$j_k = \begin{cases} 0, & \text{do not launch the asset,} \\ \beta, \quad \beta = 1, \ldots, C, & \text{launch the asset.} \end{cases} \tag{31}$$

A comparison of expected reward between launching the asset at the current hour versus the next hour is

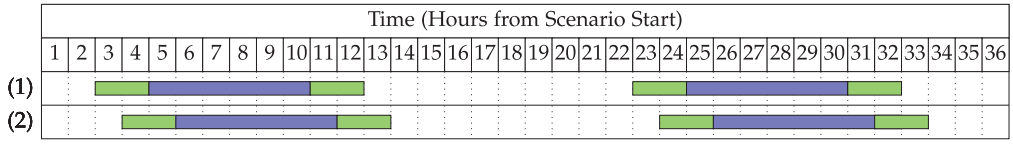


Fig. 5. Illustration of rollout for deciding when to fly with the traveling time (green) and search time (blue).

performed using rollout with the heuristic defined earlier. If launching the asset during the current time epoch results in a higher reward, then the asset will be assigned to the case with the highest total reward r_i in (28) and assigned for the first search hour to the selected case. If launching the asset during the next time epoch results in a higher reward, then we simply increment the time epoch and repeat the process. Fig. 5 illustrates this rollout heuristic for determining the expected reward for launching at a different hour.

When the asset is in flight, for the second through final hour of the search, the control variable j_k takes on a different set of values, detailed as follows:

$$j_k = \begin{cases} j_{k-1}, & \text{stay on the current case,} \\ \tilde{j} \neq j_{k-1}, & \text{switch to a different case with the cost of additional travel time.} \end{cases} \tag{32}$$

We illustrate the computation of the heuristic for the one-step lookahead rollout in Fig. 6. The first example illustrates the situation when the surveillance asset is searching for case j and chooses to stay on case j for the remaining search interval. The second example illustrates the situation, wherein the asset currently searching for case j switches to a new case $\tilde{j} \neq j$, while considering the cost of additional travel time from case j to case $\tilde{j}$. The travel time between the new case $\tilde{j}$ to the asset's home base is then the new return travel time for the asset. The optimal control action is selected based on the maximum expected reward, as in (27). This process is repeated for each time epoch k to obtain a feasible asset–case assignment over the planning horizon.

F. Multistep Lookahead Approximate Dynamic Programming I

We propose two multistep lookahead ADP strategies to obtain near-optimal assignments for all assets. The first method begins with an m-length permutation of the asset order for which to start the allocation. That is, $m = 1$ corresponds to searching over each asset; $m = 2$ corresponds to searching over all possible pairs of assets; and so on. The PoA is then updated with respect to each asset–case–time assignment to ensure that there are no duplicate asset–case–time tuples. The difference between the two methods lies in how the remaining assets are allocated. In the first proposed method, we exhaustively compute the feasible asset assignment for all the available assets and fix the allocation corresponding to the asset with the highest r_i. The PoA is then updated and the process is then repeated until either no more assets are available or all cases are fully allocated. Once all the assets are allocated, we reset the PoA surface to its original state and repeat the process from the beginning with the next possible m-length subset of assets to start the initial asset assignment over the time horizon. We refer to this method as mSLADP-I. The pseudocode is shown in Algorithm 4. In Algorithm 4's pseudocode, line 1 generates the m-length permutation of asset order, where m specifies the size of the subset permutation to be used for the initial asset allocation. Lines 4–7 find the best allocation given each asset i in a specific asset order from line 1 and update the PoA surface, accordingly. Then, lines 12–18 compute the best assignment for the remaining unassigned assets. Line 12 finds the best assignment for each unassigned asset and line 17 selects the best asset i^* for allocation. The PoA surface is subsequently updated in line 18. Lines 20–24 save the complete assignment and reset the parameters for the next asset permutation sequence generated in line 1.

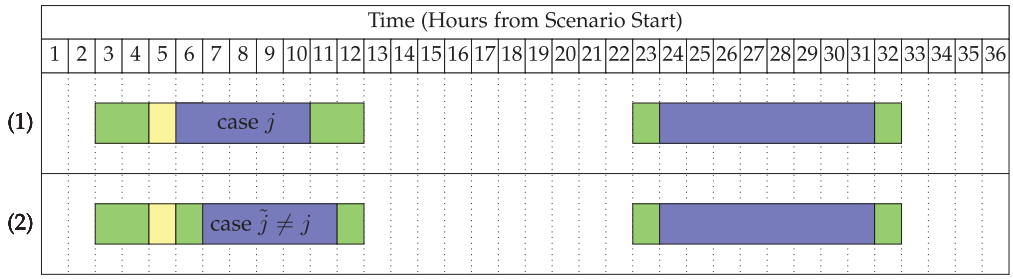


Fig. 6. Illustration of one-step lookahead. 1) Stay at current case; 2) switch to a different case with the cost of additional traveling time; and 3) return to the asset's base.

ALGORITHM 4 *m*SLADP-I

1: PermSeq = Perm($\{1, \ldots, A\}, m$) ▷ m-length permutation of asset order, where m specifies the *size* of the subset permutation to be used for initial asset allocation
2: **for each** AssetSeq in PermSeq **do**
3: **for each** $i \in$ AssetSeq **do**
4: assign(i) = ADP(i)
5: BestAssign ← BestAssign + assign(i)
6: updatePoA(assign(i))
7: AssignedAsset ← AssignedAsset + i
8: **end for**
9: **while** length(AssignedAsset) ≤ A **do**
10: **for each** $i \in \{1, \ldots, A\}$ **do**
11: **if** $i \notin$ AssignedAsset **then**
12: assignTemp(i) = ADP(i)
13: **end if**
14: **end for**
15: b_assign(i^*) = MaxReward(assignTemp) ▷ Given the previous allocations, select the asset assignment with the highest r_i among the remaining available assets
16: AssignedAsset ← AssignedAsset + i^*
17: BestAssign ← BestAssign + b_assign(i^*)
18: updatePoA(assignment(i^*))
19: **end while**
20: PotentialAssign ← PotentialAssign + BestAssign
21: assign = ∅
22: AssignedAsset = ∅
23: BestAssign = ∅
24: resetPoA
25: **end for**
26: BestAssign = MaxReward(PotentialAssign)

G. Multistep Lookahead Approximate Dynamic Programming II

The second multistep lookahead method (referred to as *m*SLADP-II), as in the first method, begins with an m-length permutation of asset ordering. The difference between *m*SLADP-I and *m*SLADP-II is how the algorithm computes the asset allocation for the remaining assets. In *m*SLADP-II, we iteratively compute the best asset allocation for each i. Once the best assignment is found for asset i, we immediately fix the corresponding schedule and update the PoA surface. There is no additional loop to find the best asset–case assignment among all the remaining assets. Hence, *m*SLADP-II is faster and less complex than *m*SLADP-I. The same process is then repeated until all cases are fully allocated or there are no more assets available. The pseudocode is shown in Algorithm 5. In this pseudocode, lines 1–7 remain the same as Algorithm 4. The difference lies in lines 10–17, where, for each unassigned asset, we find the best assignment corresponding to an asset i^* and update the PoA surface accordingly. Once all assets are assigned, we reset all the parameters for the next asset permutation sequence generated from line 1.

ALGORITHM 5 *m*SLADP-II

1: PermSeq = Perm($\{1, \ldots, A\}, m$)
2: **for each** AssetSeq in PermSeq **do**
3: **for each** $i \in$ AssetSeq **do**
4: assign(i) = ADP(i)
5: BestAssign ← BestAssign + assign(i)
6: updatePoA(assign(i))
7: AssignedAsset ← AssignedAsset + i
8: **end for**
9: **for each** $i \in$ {1,..., A} **do**
10: **if** $i \notin$ AssignedAsset **then**
11: assign(i) = ADP(i)
12: **end if**
13: **end for**
14: AssignedAsset ← AssignedAsset + i^*
15: BestAssign ← BestAssign + assign(i)
16: updatePoA(assignment(i^*))
17: PotentialAssign ← PotentialAssign + BestAssign
18: assign = ∅
19: AssignedAsset = ∅
20: BestAssign = ∅
21: resetPoA
22: **end for**
23: BestAssign = MaxReward(PotentialAssign)

IV. SIMULATION AND COMPUTATIONAL RESULTS

The proposed algorithms were implemented in Python 2.7 on an Intel® Core™ i7-6600U CPU @ 2.60 GHz × 4 with 32 GB RAM. Our computational results are organized as follows: We first describe the mission scenario. Then, we discuss the solution quality of various algorithms with respect to objectives O1–O3 and their runtimes. Additionally, we conduct scalability analyses of the algorithms by varying the number of assets and cases, as well as robustness of the various algorithms using a signal-to-noise ratio (SNR) metric from robust design [37].

A. Scenario Description(s)

There are two main areas of operation in the simulated scenario: the East Pacific Ocean and the Caribbean Sea. The PoA surfaces corresponding to this area of responsibility (AOR) were partitioned into a grid of 90 × 138 cells, where each cell is a square with a side length of 30 nautical miles. The total area of the AOR was ≈11 million square nautical miles. The lower left corner of the rectangular AOR had a latitude and longitude of 10°S and 110°W, respectively.

TABLE II
Smuggler Cases

Case	Case ID	Vessel type	Speed (kts)	Payload (kg)	No. of smugglers
1	GF1	Go fast	30	1000	3
2	PG1	Panga	20	450	2
3	GF2	Go fast	30	1000	3
4	PG2	Panga	20	450	2
5	PG3	Panga	20	450	2
6	PG4	Panga	20	450	2
7	SP1	SPSS[a]	8	2500	3
8	FSV1	FSV[b]	4	5000	2
9	PG5	Panga	20	450	2
10	PG6	Panga	20	450	2

[a]Self-propelled semi-submersible.

[b]Fully submerged vessel.

The PoA surfaces forecasted 10 smuggler cases, of which 5 were located in the East Pacific Ocean and the remaining 5 were located in the Caribbean Sea. The details for each case can be found in Table II and Fig. 7. These cases are generated based on Navy intelligence, which typically comprises estimates of the expected number of smugglers on board and the size of the contraband shipment. Often there are few "active" cases, i.e., cases that targeteers deem to have sufficient actionable intelligence to allocate assets to. We assume the PoA surfaces reflect the spatiotemporal probabilities pertaining to such "active" cases. Four different types of smuggler vessels were considered: 1) Go fast—small, fast boats capable of reaching high speeds; 2) Panga—modest-sized, fast boats that are easy to build by the smugglers; 3) self-propelled semi-submersible (SPSS)—narco-submarines capable of shifting heavy loads long distances while almost submerged under the ocean's surface [38]; and 4) fully submerged vessel—makeshift submarine-like vessels that can remain submerged with large quantities of cocaine aboard. Each case had a unique departure, destination, and waypoint combination. Waypoints are defined as possible areas in the ocean where the cargo is transferred to another vessel or a change in trajectory of the smuggler is predicted. Additionally, each case also had an associated payload measured in kg of cocaine. This is relevant when we run the algorithm with objective O2. An important fact to note is that each case had different start and end times. Fig. 8 details the time epochs when each smuggler case is deemed active. Cases with high uncertainty had wide bands of PoA. The amount of uncertainty is dependent on the type of smuggler vessel (e.g., SPSSs can be extremely difficult to detect, and thus the corresponding PoA surfaces reflect this in long and broad bands of probability reflecting spatial and temporal uncertainty) and/or departure time uncertainty.

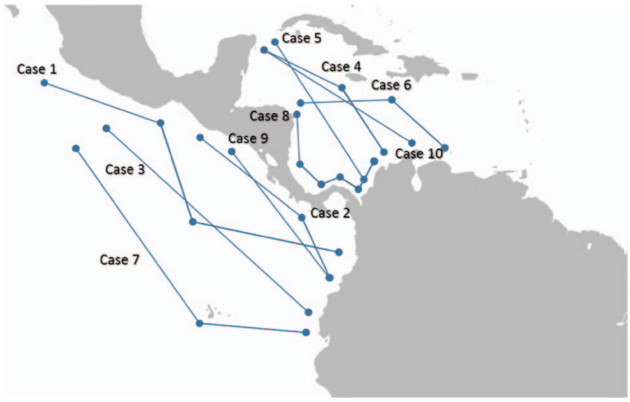


Fig. 7. Experiment scenario.

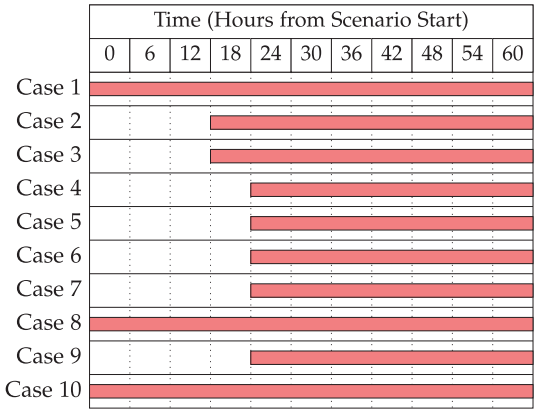


Fig. 8. Chart displaying when each smuggler case is active over the 72 h time horizon. Cases are active up through time $K = 72$ and do not necessarily end at that time, but rather, due to the time horizon of the forecast data, are truncated.

In the scenario, 10 P-3 surveillance assets were considered as available for allocation during the planning horizon. The home bases of individual surveillance assets are detailed in Table III. Each asset carries two different types of sensors with performance parameters detailed in Table IV.

We simulated the scenario with a granularity of 1 h (i.e., the forecasted surfaces were for each hour, on the hour; thus $\Delta = 1$ h). The forecasts extended to 72 h out from the current time (i.e., $K = 72$) and an asset allocation solution (e.g., $x_{ijk} = 0$ or 1) was required for each time epoch, k, in order for the algorithm to terminate.

Note that we omit E-B&C for large-sized scenarios in our results due to an exponential increase in computation times. Therefore, for E-B&C, we compute the solution for scenarios involving only up to 5 assets and 10 cases.

TABLE III
Asset Home Base Location (Longitude, Latitude)

1, 6	(−69.7617, 18.5036)
2, 7	(−79.3833, 9.07111)
3, 8	(−85.5442, 10.5931)
4, 9	(−89.0558, 13.4406)
5, 10	(−92.37, 14.7942)

TABLE IV
Sensor-to-Target Sweep Width (nm)

Sensor type	FSV[a]	SPSS[b]	FV[c]	Panga	GF[d]	MV[e]	SV[f]	UNK[g]
APS 115	5	7.5	7.5	9	7.5	7.5	7.5	2.5
APS 137	10	15	15	18	15	15	15	5

[a]Fully submerged vessel.
[b]Self-propelled semi-submersible.
[c]Fishing vessel.
[d]Go fast.
[e]Merchant vessel.
[f]Supply vessel.
[g]Unknown (other).

B. Solution Quality with Different Objective Functions

Using the aforementioned values for the parameters, we ran the simulation for all the approaches to schedule the 10 specified assets over the 72 h planning horizon. Tables V–VII show the CPoSD for the GB&C-II method for objectives O1, O2, and O3, respectively. Parallel GB&C-II has the same result as the sequential GB&C-II. Therefore, we omit the parallel GB&C-II from the quality comparison.

We refer to Tables V–VII as COA matrices [5]. The COA matrices aid the DM in understanding the reasoning behind the algorithm's behavior and its output by giving metrics for both individual asset–case pairs and, overall, the probability an asset detects at least one case (PDC) and the probability that a case is detected by at least one asset (PDA). These matrices may be generated to assess the allocation performance at a particular time epoch, or, as shown in Tables V–VII, the cumulative asset allocation performance up to that point in time (in Tables V–VII, through $K = 72$).

Solving with respect to objective O1 (Table V) resulted in an asset allocation with the highest expected weight of contraband detected, totaling 7828 kg of cocaine compared to objectives O2 and O3 (Tables VI and VII). This implies that we have a 64% success rate of detecting the transport of contraband with respect to the total possible for the experimental scenario of 12,200 kg of contraband. The asset allocations with respect to objective O1 have 15.5% and 10.1% more contraband disrupted when compared to objectives O2 and O3, respectively. In Table V, case 8 has the most amount of contraband (5000 kg) with a CPoSD = 0.95. Solving with respect to objective O3 resulted in the detection of a higher expected weight of contraband (5.9%), expected number of detections (6.8%), and expected number of smugglers (7.5%) compared to objective O2. This could be caused by the uniform priority weight vector used in objective O2.

For the sake of compactness, we omit the COA matrices used in demonstrating the performance of the other approaches implemented and, instead, quantify the goodness of the allocation by comparing the algorithms with that of GB&C-II algorithm as measured by the expected weight of the contraband detected, expected number of detections, and expected number of smugglers detected.

The sums of the totals for each objective for each algorithm are shown in Table VIII. Fig. 9 shows a normalized representation of the results detailed in Table VIII, where the largest possible number of detections and contraband detected was utilized as a basis for normalization of both metrics, respectively, to compare the expected number of detections and contraband weight detected. Note that Fig. 9 only contains the results for 1SLADP-I and 1SLADP-II; the detailed solutions of mSLADP with $m > 1$ are shown later in Section IV-C.

We illustrate in Table VIII and Fig. 9 that all B&C-based algorithms optimizing objective O2 are outperformed by the same algorithms optimizing objective O3 in terms of both the expected number of detections and expected number of smugglers. When comparing GB&C-I and GB&C-II, optimizing with respect to objective O2 resulted in 4% less expected number of detec-

TABLE V
Objective O1: Maximize Weight of Contraband Detected

Asset	Case 1	Case 2	Case 3	Case 4	Case 5	Case 6	Case 7	Case 8	Case 9	Case 10	PDC
1	–	–	–	–	–	–	–	0.76	–	–	0.76
2	–	–	–	–	–	–	0.43	0.29	–	–	0.60
3	0.09	–	–	–	–	–	0.25	0.29	–	–	052
4	0.15	–	–	–	–	0.28	–	–	–	0.14	0.47
5	0.21	0.14	–	–	–	–	–	–	–	–	0.33
6	–	–	–	–	–	–	–	0.61	–	–	0.61
7	–	–	–	–	–	–	0.40	–	–	–	0.40
8	–	–	0.09	—	–	–	0.21	–	–	–	0.28
9	0.17	–	–	–	–	0.30	–	–	–	–	0.42
10	–	–	–	0.20	–	0.11	–	–	–	0.06	0.33
PDA	0.49	0.14	0.09	0.20	–	0.55	0.80	0.95	–	0.19	[a,b,c]

[a]Expected weight of contraband disrupted: 7828 kg.
[b]Expected number of detections: 3.41.
[c]Expected number of smugglers: 8.21.

TABLE VI
Objective O2: Maximize Number of Detections

Asset	Case 1	Case 2	Case 3	Case 4	Case 5	Case 6	Case 7	Case 8	Case 9	Case 10	PDC
1	–	–	–	–	–	–	–	0.76	–	–	0.76
2	–	0.27	–	–	–	0.38	–	–	–	0.15	0.61
3	0.20	–	–	–	–	–	0.28	–	–	–	0.43
4	0.15	–	–	–	–	–	–	0.10	–	0.15	0.35
5	0.19	–	–	0.17	–	–	–	–	–	–	0.33
6	–	–	–	0.13	–	–	–	–	–	0.20	0.31
7	–	0.29	–	–	–	–	–	0.37	–	–	0.55
8	0.13	–	–	–	–	–	–	–	0.23	–	0.33
9	–	–	0.10	–	0.18	–	–	–	–	–	0.25
10	0.17	–	–	–	–	–	–	–	–	0.05	0.21
PDA	0.61	0.48	0.10	0.28	0.18	0.38	0.28	0.86	0.23	0.46	[a,b,c]

[a]Expected weight of contraband detected: 6619 kg.
[b]Expected number of detections: 3.85.
[c]Expected number of smugglers: 8.67.

TABLE VII
Objective O3: Maximize Number of Smugglers Detected

Asset	Case 1	Case 2	Case 3	Case 4	Case 5	Case 6	Case 7	Case 8	Case 9	Case 10	PDC
1	–	–	–	–	–	0.21	–	0.61	–	–	0.69
2	–	–	–	–	–	0.34	0.26	–	–	0.15	0.59
3	0.09	0.23	–	–	–	–	0.25	–	–	–	0.48
4	0.11	–	–	0.21	–	–	0.10	–	–	–	0.37
5	0.18	–	0.10	–	–	–	–	–	–	–	0.25
6	–	–	–	–	–	–	–	0.58	–	0.15	0.64
7	–	0.27	–	–	–	–	–	–	–	0.11	0.35
8	–	–	–	–	0.18	–	–	–	0.22	–	0.36
9	0.16	–	–	–	–	0.30	–	–	–	–	0.41
10	–	–	0.09	0.16	–	–	–	–	–	–	0.23
PDA	0.44	0.44	0.17	0.34	0.18	0.63	0.51	0.84	0.22	0.35	[a,b,c]

[a]Expected weight of contraband detected: 7036 kg.
[b]Expected number of detections: 4.13.
[c]Expected number of smugglers: 9.37.

tions and 1.2% less expected number of smugglers than when optimizing with respect to objective O3.

In terms of the amount of contraband detected, using the GB&C-I algorithm resulted in an allocation that obtained the highest expected amount of contraband detected when solving for objective O1; however, its solutions for maximizing the expected number of detections or expected number of smugglers were inferior to the ADP-based algorithms. In general, we see that the B&C-based methods are able to obtain more contraband when

TABLE VIII
Algorithm comparison

Contraband disrupted (kg)				
Objective	GB&C-I	GB&C-II	1SLADP-I	1SLADP-II
O1	7869	7828	7520	7821
O2	6658	6619	7185	7610
O3	7188	7036	7594	7591
No. of detections				
Objective	GB&C-I	GB&C-II	1SLADP-I	1SLADP-II
O1	3.47	3.41	3.57	3.68
O2	3.87	3.84	3.72	4.08
O3	4.12	4.13	3.80	4.04
No. of smugglers				
Objective	GB&C-I	GB&C-II	1SLADP-I	1SLADP-II
O1	8.35	8.21	8.37	8.72
O2	8.56	8.67	8.40	9.21
O3	9.50	9.37	8.70	9.12

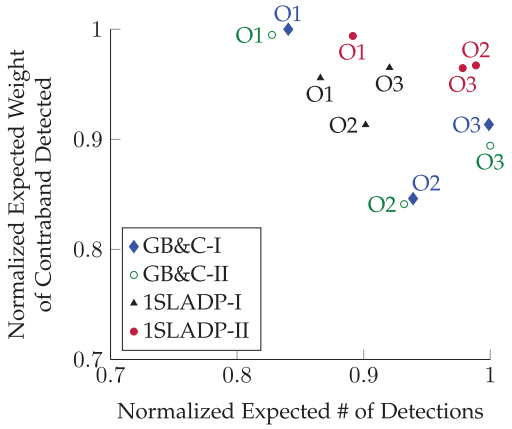


Fig. 9. A normalized view comparing the performance of all the algorithms, with respect to the expected weight of contraband disrupted (O1), the expected number of interdictions (O2), and the expected number of smugglers (O3).

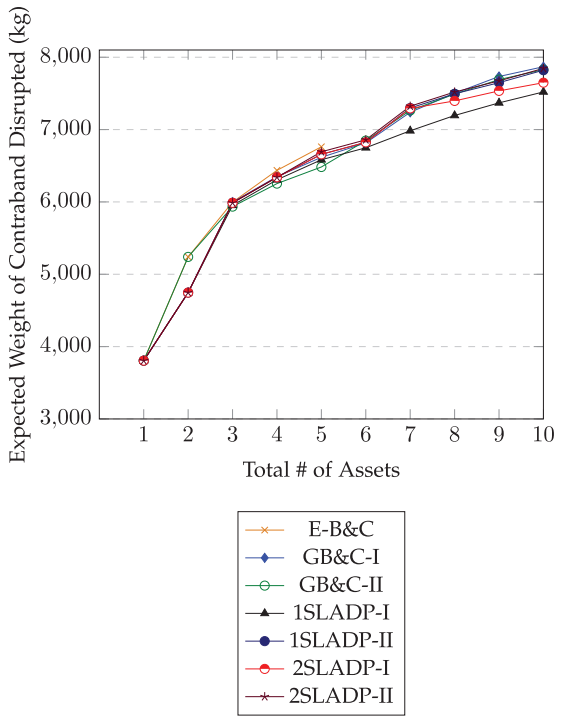


Fig. 10. The expected weight of contraband disrupted for each algorithm by varying the number of available assets.

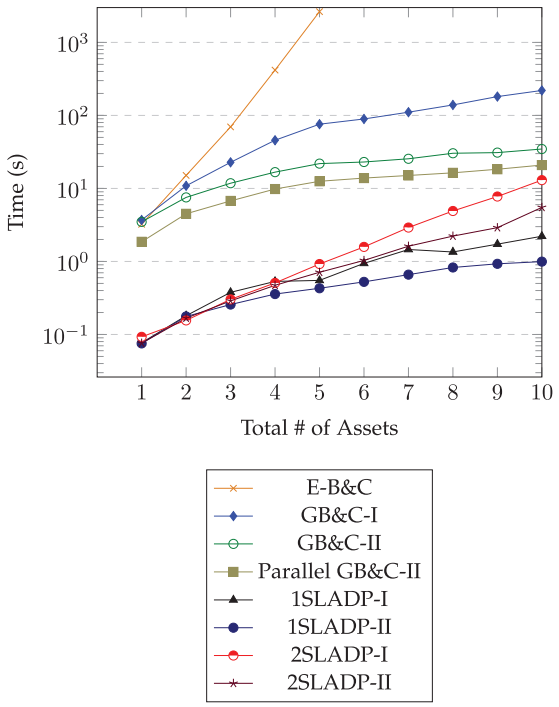


Fig. 11. The CPU runtimes for each algorithm by varying the number of available assets.

solving with respect to objective O1, while the ADP-based methods are able to get better solutions for the expected number of detections and expected number of smugglers when solving with respect to objectives O2 and O3 with the exception of 1SLADP-I for objective O3.

C. Scalability: Available Asset Sensitivity

In this section, we use objective O1 for the scalability studies with respect to the number of assets. To measure the scalability, we limited the number of assets available for allocation for the scenario from 1 to 10 aircraft. Figs. 10 and 11 show the expected weight of contraband disrupted and the runtimes, respectively. The detailed values are given in Tables IX and X. In Fig. 10 and Table IX, we see that ADP-based algorithms (1SLADP-I, 1SLADP-II, 2SLADP-I, and 2SLADP-II) are able to obtain similar amounts of contraband disrupted, differing by only up to 339.7 kg (4.6%) of contraband.

Similarly, the B&C-based algorithms (E-B&C, GB&C-I, and GB&C-II) are able to obtain similar amounts of contraband disrupted, differing by only up to 279.1 kg of contraband among the three. E-B&C, intuitively, outperformed the other B&C variations (and all other algorithms for that matter) among the scenarios simulated until runtime became an issue. GB&C-II is able to obtain a better result compared to GB&C-I when there are two, six, or seven assets available for allocation. This is due to the nature of the scenario or the characteristics of the smuggler cases. Since GB&C-I iterates through all available assets, there is a tendency that closer (with respect to assets' home base) cases

TABLE IX
Expected Weight of Contraband Disrupted (kg) for Varying Asset Availability

No. of assets	E-B&C	GB&C-I	GB&C-II	1SLADP-I	1SLADP-II	2SLADP-I	2SLADP-II
1	3806	3806	3806	3806	3806	3806	3806
2	5240	4747	5240	4747	4747	4747	4747
3	5997	5997	5935	5952	5988	5988	5988
4	6436	6349	6253	6306	6341	6341	6341
5	6762	6617	6483	6583	6658	6658	6693
6	–	6816	6847	6748	6823	6823	6858
7	–	7239	7265	6983	7298	7292	7323
8	–	7501	7490	7195	7494	7396	7519
9	–	7734	7691	7369	7648	7535	7673
10	–	7869	7828	7520	7821	7648	7846

TABLE X
Simulation Runtime (s) for Varying Asset Availability

No. of assets	E-B&C	GB&C-I	GB&C-II	Parallel GB&C-II	1SLADP-I	1SLADP-II	2SLADP-I	2SLADP-II
1	3.23	3.70	3.48	1.86	0.08	0.08	0.09	0.08
2	15.1	10.9	7.54	4.49	0.18	0.18	0.16	0.17
3	69.6	22.8	11.8	6.75	0.38	0.26	0.30	0.29
4	418	45.7	16.7	9.82	0.53	0.36	0.51	0.47
5	2639	75.9	21.9	12.5	0.55	0.43	0.92	0.71
6	–	89.4	23.0	13.9	0.94	0.53	1.58	1.03
7	–	111	25.5	15.1	1.46	0.66	2.91	1.62
8	–	139	30.3	16.4	1.35	0.83	4.91	2.22
9	–	181	31.0	18.32	1.73	0.93	7.75	2.90
10	–	220	34.7	20.9	2.22	0.99	13.0	5.50

are allocated first, since there is less travel time and, hence, are more rewarding. In turn, this may limit the options available to assets considered for allocation in later iterations since cases, previously in close proximity to their home base, may already be allocated and, due to longer travel time, will be much less rewarding or not at all. Similar problems arose with 1SLADP-I algorithm, which obtains less expected contraband disrupted compared to 1SLADP-II algorithm when there are more than six assets available for allocation, differing by up to 314.9 kg of contraband. We are able to minimize the effect of this problem by applying a two-step lookahead strategy. 2SLADP-I algorithm obtains less expected contraband disrupted compared to 2SLADP-II algorithm when there are more than five assets available for allocation, differing by up to 197.6 kg of contraband.

As Fig. 11 and Table X show, E-B&C has the slowest runtime. There is a maximum speedup of 34.8, 120.6, 210.6, 4794, 6146, 2861, and 3711 and an average speedup of 9.8, 30.9, 53.7, 1177, 1542, 809, and 994 when comparing the runtimes of GB&C-I to GB&C-II, parallel GB&C-II, 1SLADP-I, 1SLADP-II, 2SLADP-I, and 2SLADP-II, respectively. Over all the asset availability scenarios tested, the average speedups of GB&C-II, parallel GB&C-II, 1SLADP-I, 1SLADP-II, 2SLADP-I, and 2SLADP-II are 3.6, 6.1, 87, 143, 52, and 72 times, respectively, faster compared to GB&C-I.

Our key finding here is that, with a 1.6% sacrifice in optimality on average, GB&C-II provides a solution nearly identical to that of E-B&C, while offering a solution in a fraction of the time (up to nearly 210.6 times faster among the simulated results). Alternatively, at a cost of 2.5% suboptimality on average, but more than 6146 times faster speedup, we can run 1SLADP-II for a given scenario. Similarly, at a cost of 2.4% suboptimality on average, 2SLADP-II offers more than 3711 times faster speedup.

In general, GB&C-II should be used when the total number of assets is less than 3 due to its minimal sacrifice in optimality (on average 1.6%). When the number of assets is greater than 3, 2SLADP-II should be used.

D. Scalability: Varying the Number of Cases

Here, we vary the number of cases from 1 to 10, while fixing the number of available assets to 10. Figs. 12 and 13 show the expected weight of contraband disrupted and the runtimes, respectively. The detailed values for each figure are given in Tables XI and XII, respectively. From Fig. 12 and Table XI, we see that all the algorithms have very similar solution quality. We see a noticeable increase in contraband disruption for case 8 (5000 kg of contraband). All algorithms obtained a similar amount of expected contraband disrupted.

Fig. 13 and Table XII show the runtimes. As expected, GB&C-I has the slowest runtimes, while the 1SLADP-II algorithm has the fastest runtime (<1 s). There are maximum speedups of 7, 11, 99, 221, 17, and 40 when comparing the runtimes of GB&C-I to GB&C-II, parallel GB&C-II, 1SLADP-I, 1SLADP-II, 2SLADP-I, and

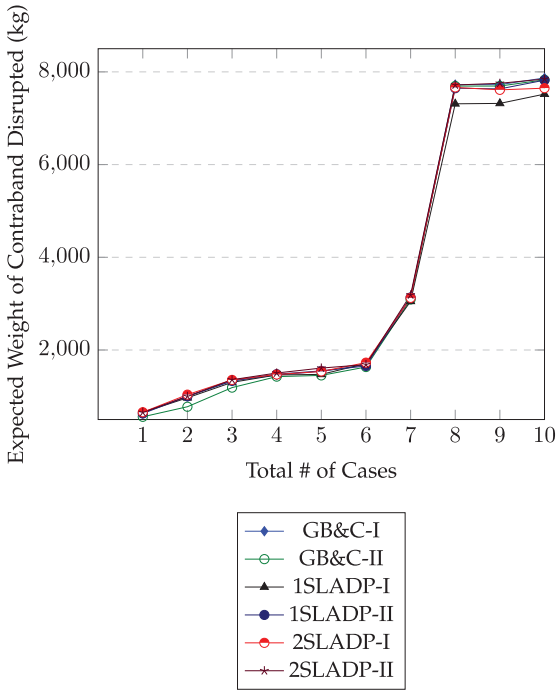


Fig. 12. The expected weight of contraband disrupted for each algorithm by varying the number of available cases.

TABLE XI
Expected Weight of Contraband Disrupted (kg) for Varying Case Availability

No. of cases	GB&C-I	GB&C-II	1SLADP-I	1SLADP-II	2SLADP-I	2SLADP-II
1	653.2	558.7	653.2	629.6	657.7	653.2
2	1000	775.8	968.7	999.4	1037	999.4
3	1332	1189	1302	1334	1355	1359
4	1459	1425	1462	1479	1471	1504
5	1540	1452	1474	1549	1537	1610
6	1680	1633	1707	1655	1727	1685
7	3145	3090	3049	3133	3125	3188
8	7725	7694	7311	7648	7665	7716
9	7725	7694	7320	7633	7612	7754
10	7869	7828	7520	7821	7648	7846

TABLE XII
Simulation Runtime (s) for Varying Case Availability

No. of cases	GB&C-I	GB&C-II	Parallel GB&C-II	1SLADP-I	1SLADP-II	2SLADP-I	2SLADP-II
1	0.85	0.34	1.08	0.68	0.23	1.99	0.86
2	4.09	1.36	4.47	0.66	0.32	3.74	1.27
3	5.88	2.67	1.89	0.85	0.42	4.13	1.58
4	11.8	3.93	3.06	1.00	0.54	5.24	1.91
5	18.0	4.77	4.01	1.48	0.57	5.51	2.16
6	24.7	7.48	5.31	1.34	0.64	6.95	2.38
7	56.0	11.7	7.27	1.45	0.72	7.79	2.55
8	108	13.2	8.56	1.67	0.80	9.45	3.13
9	183	15.9	9.85	1.92	0.89	11.9	4.00
10	220	34.7	20.9	2.22	0.99	13.0	5.50

2SLADP-II, respectively. On average, the speedups of the GB&C-II, parallel GB&C-II, 1SLADP-I, 1SLADP-II, 2SLADP-I, and 2SLADP-II algorithms were 4.3, 6, 33, 71.8, 6, and 16.5 times, respectively.

The key point here is that the algorithm 2SLADP-I is very efficient and is recommended for scenarios when the number of cases is less than or equal to the number of assets, which is often the case.

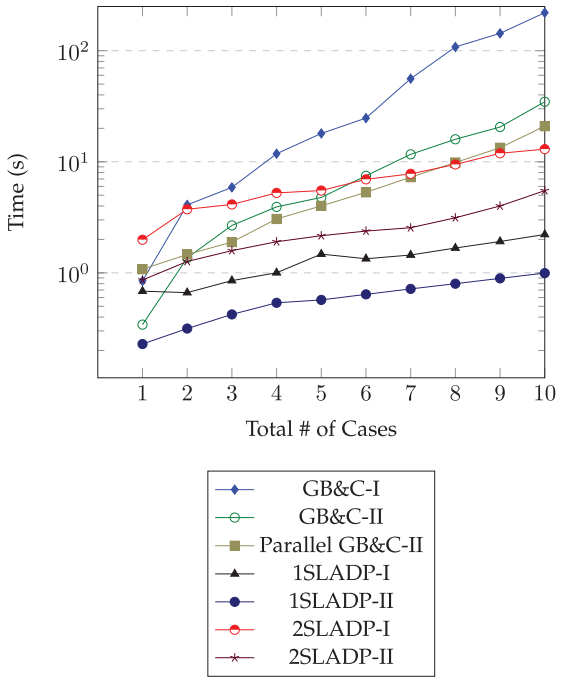


Fig. 13. The CPU runtimes for each algorithm by varying the number of available cases.

E. Robustness: Monte Carlo Evaluation of Asset Allocation Strategies

To test the robustness of each asset allocation algorithm, we simulated 100,000 trajectories of smugglers (10,000 from each case) behaving as in our benchmark scenario. Sampling from the PoA surfaces, we obtained waypoints for each smuggler at each time epoch and joined them together to obtain a full path. From these paths, we measured whether the smuggler traversed through any allocated patrol boxes during the allocated search time, and if so, what was the aircraft's probability of detecting the target during those time epoch(s). Table XIII shows the detailed performance statistics for each algorithm over the 100,000 Monte Carlo simulations. A useful metric to measure an algorithm's goodness is that of nominal-the-best SNR [37], that is,

$$\text{SNR} = 10\log_{10}\frac{\mu^2}{\sigma^2}. \tag{33}$$

TABLE XIII
Monte Carlo Analysis (from 100,000 Runs)

Objective	Mean of contraband detected (μ) in kg	Standard deviation of contraband detected (σ) in kg	SNR (dB)
GB&C-I	7616	246.3	29.8
GB&C-II	7632	252.5	29.6
1SLADP-I	7645	244.2	29.9
1SLADP-II	7610	218.4	30.8
2SLADP-I	7648	240.3	30.1
2SLADP-II	7612	203.3	31.5

Nominal-the-best SNR is a useful measure when the goal is to maximize a mean and minimize the variation. Note that maximization of this metric seeks to minimize the coefficient of variation (=standard deviation/mean) and is thus a measure of robustness of a solution. From the results of 100,000 Monte Carlo runs, we found that the algorithm 2SLADP-II performs the best with respect to objective O1, when measured using the SNR. The 2SLADP-II algorithm obtained, on average, 7612 kg of contraband (out of a total of 12,200 kg purportedly transported).

As Table XIII shows, all algorithms were able to obtain a similar expected amount of contraband, with 2SLADP-II proving to be the most robust, as measured in terms of nominal-the-best SNR.

V. CONCLUSION AND FUTURE WORKS

In this paper, we proposed five asset allocation algorithms to the maritime surveillance problem: 1) E-B&C: enumerate all possible asset–case combinations over all times and find the optimal allocation. 2) GB&C-I: enumerate and solve for the best allocation for each asset and compute the PoSD for each asset, and iteratively generate a schedule based on the highest PoSD. 3) GB&C-II: similar to E-B&C, except the algorithm directly enforces the asset schedule once the best allocation is found. 4) *m*SLADP-I: utilize multistep lookahead rollout in a heuristic to iteratively schedule asset–case assignments for individual time epochs. 5) *m*SLADP-II: similar to *m*SLADP-I, except that the algorithm directly enforces the asset schedule based on the highest incremental reward.

We validated each algorithm and solved the NP-hard counter-smuggling surveillance problem in a relatively short amount of time for any of the three objectives examined—maximizing the contraband disrupted, number of detections, or number of smugglers detected. We found that B&C-based methods are able to obtain more contraband when optimizing the amount of contraband disrupted, while the approximate dynamic approaches are better at optimizing over the number of smugglers and the number of detections.

We conducted scalability and robustness analyses to evaluate the solution quality, runtimes, and contraband detection performance behavior of each algorithm. We found that the algorithms scale reasonably well with the problem size. We also found that ADP-based approaches are able to obtain effective asset allocations within seconds of computation time with a minimal sacrifice in optimality, while proving to provide the most robust solution as measured by the SNR metric. Additionally, we found the 2SLADP-II algorithm to be the best when measuring with respect to nominal-the-best SNR. Our future work includes further sensitivity analyses with varying asset types, aloft times, number of unavailable assets, and rest times, and spatiotemporal variations in the PoA surface (e.g., scenario-based asset allocation to handle uncertainty in PoA surfaces). Additionally, higher fidelity simulations could easily be analyzed for more accurate detection models and other operational PoA surfaces (e.g., historical flow surface and active cases). Future work also includes the incorporation of unmanned aerial vehicles (UAVs), either as in [39], where solely UAVs collaborate, or in a mixed-initiative sense, an augmentation of our proposed approach.

REFERENCES

[1] United Nations Office on Drugs and Crime, "World Drug Report 2010," 2010.

[2] United Nations Office on Drugs and Crime, "World Drug Report 2011," 2011.

[3] Joint Chiefs of Staff, *Command and Control for Joint Maritime Operations.* Washington, DC, USA: Joint Chiefs of Staff, 2013.

[4] A. Smirnov
"Context-driven decision making in network-centric operations: Agent-based intelligent support,"
in *Proc. CKM Workshop*, vol. 7, no. 812, Jan. 2006.

[5] D. Sidoti, D. F. Ayala, S. Sankavaram, X. Han, M. Mishra, W. An, D. Kellmeyer, J. Hansen, and K. Pattipati
"Decision support information integration platform for context-driven interdiction operations in counter-smuggling missions,"
in *Proc. IEEE/SICE Int. Symp. Syst. Integr.*, 2014, pp. 659–664.

[6] D. Sidoti, X. Han, L. Zhang, G. V. Avvari, D. F. M. Ayala, M. Mishra, M. S. Sankavaram, D. L. Kellmeyer, J. A. Hansen, and K. R. Pattipati
"Context-aware dynamic asset allocation for maritime interdiction operations,"
IEEE Trans. Syst., Man, Cybern.: Syst., vol. 50, no. 3, pp. 1055–1073, 2020.

[7] D. Sidoti, D. F. M. Ayala, X. Han, M. Mishra, S. Sankavaram, W. An, K. R. Pattipati, and D. L. Kleinman
"Evaluating the value of information in the presence of high uncertainty,"

in *Proc. 18th ICCRTS: C2 in Underdeveloped, Degraded, and Denied Operational Environments*, June 2013.

[8] J. A. Hansen, D. Hodyss, C. H. Bishop, and W. Campbell "Coupled METOC/INTEL risk assessment," U.S. Patent Appl. 13/272,272, Oct. 13, 2011. [Online]. Available: https://www.google.com/patents/US20120095946

[9] B. O. Koopman "A theoretical basis for method of search and screening," Columbia Univ., New York, NY, USA, Tech. Rep., 1946.

[10] B. O. Koopman *Search and Screening: General Principles with Historical Applications*, vol. 7. New York, NY, USA: Pergamon Press, 1980.

[11] L. H. Nunn "An introduction to the literature of search theory," Operations Evaluation Group, Center for Naval Analyses, Alexandria, VA, USA, Tech. Rep., 1981.

[12] N. Blachman and F. Proschan "Optimum search for objects having unknown arrival times," *Oper. Res.*, vol. 7, no. 5, pp. 625–638, 1959.

[13] S. M. Pollock "Sequential search and detection," MIT Operations Research Center, Cambridge, MA, USA, Tech. Rep., 1964.

[14] A. Charnes and W. W. Cooper "The theory of search: Optimum distribution of search effort," *Manage. Sci.*, vol. 5, no. 1, pp. 44–50, 1958.

[15] L. D. Stone *Theory of Optimal Search*, vol. 118. Amsterdam, The Netherlands: Elsevier, 1976.

[16] M. Lukka *On the Optimal Searching Tracks for a Stationary Target*, no. 4. Turku, Finland: Institute for Applied Mathematics, University of Turku, 1974.

[17] M. Mangel "Search for a randomly moving object," *SIAM J. Appl. Math*, vol. 40, no. 2, pp. 327–338, 1981.

[18] A. Washburn and K. Wood "Two-person zero-sum games for network interdiction," *Oper. Res.*, vol. 43, no. 2, pp. 243–251, 1995.

[19] D. M. Pfeiff "Optimizing employment of search platforms to counter self-propelled semi-submersibles," Naval Postgraduate School, Monterey, CA, USA, Tech. Rep., 2009.

[20] J. O. Royset and R. K. Wood "Solving the bi-objective maximum-flow network-interdiction problem," *INFORMS J. Comput.*, vol. 19, no. 2, pp. 175–184, 2007.

[21] S. H. Jacobson, L. A. McLay, S. N. Hall, D. Henderson, and D. E. Vaughan "Optimal search strategies using simultaneous generalized hill climbing algorithms," *Math. Comput. Model.*, vol. 43, no. 9–10, pp. 1061–1073, 2006.

[22] K. Ng and N. Sancho "Regional surveillance of disjoint rectangles: A travelling salesman formulation," *J. Oper. Res. Soc.*, vol. 60, no. 2, pp. 215–220, 2009.

[23] A. O. Hero, D. Castañón, D. Cochran and K. Kastella *Foundations and Applications of Sensor Management*. Berlin, Germany: Springer, 2007.

[24] M. Kress, J. O. Royset, and N. Rozen "The eye and the fist: Optimizing search and interdiction," *Eur. J. Oper. Res.*, vol. 220, no. 2, pp. 550–558, 2012.

[25] J. Pietz and J. O. Royset "Generalized orienteering problem with resource dependent rewards," *Naval Res. Logistics*, vol. 60, no. 4, pp. 294–312, 2013.

[26] S. Campos III "An analysis of mathematical models to improve counter-drug smuggling operations," Naval Postgraduate School, Monterey, CA, USA, Tech. Rep., 2014.

[27] J. Pietz and J. O. Royset "Optimal search and interdiction planning," *Military Oper. Res.*, vol. 20, no. 4, pp. 59–73, 2015.

[28] E.-M. Wong, F. Bourgault, and T. Furukawa "Multi-vehicle Bayesian search for multiple lost targets," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2005, pp. 3169–3174.

[29] H. Sato and J. O. Royset "Path optimization for the resource-constrained searcher," *Naval Res. Logistics*, vol. 57, no. 5, pp. 422–440, 2010.

[30] J. O. Royset and H. Sato "Route optimization for multiple searchers," *Naval Res. Logistics*, vol. 57, no. 8, pp. 701–717, 2010.

[31] K. Ng and A. Ghanmi "An automated surface surveillance system," *J. Oper. Res. Soc.*, vol. 53, no. 7, pp. 697–708, 2002.

[32] W. An, D. F. M. Ayala, D. Sidoti, M. Mishra, X. Han, K. R. Pattipati, E. D. Regnier, D. L. Kleinman, and J. A. Hansen "Dynamic asset allocation approaches for counter-piracy operations," in *Proc. 15th IEEE Int. Conf. Inf. Fusion*, 2012, pp. 1284–1291.

[33] K. Byers "Situational awareness for surveillance and interdiction operations (SASIO): Tactical installation protection," Master's thesis, Naval Postgraduate School, Monterey, CA, USA, 2010.

[34] D. L. Bessman "Optimal interdiction of an adaptive smuggler," Naval Postgraduate School, Monterey, CA, USA, Tech. Rep., 2010.

[35] D. P. Bertsekas *Dynamic Programming and Optimal Control*, vol. 1, no. 2. Belmont, MA, USA: Athena Scientific, 1995.

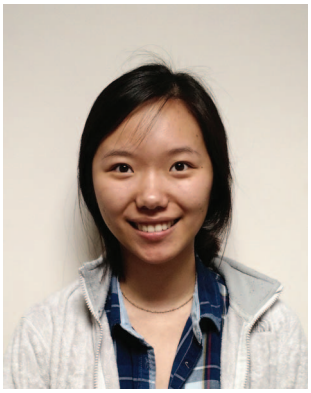
[36] M. R. Garey and D. S. Johnson "'Strong' NP-completeness results: Motivation, examples, and implications," *J. ACM*, vol. 25, no. 3, pp. 499–508, 1978.

[37] M. S. Phadke *Quality Engineering Using Robust Design*. Upper Saddle River, NJ, USA: Prentice Hall, 1995.

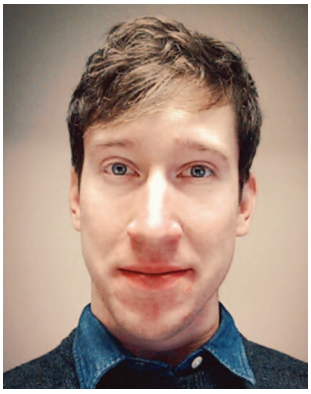
[38] L. J. Watkins "Self-propelled semi-submersibles: The next great threat to regional security and stability," Master's thesis, Naval Postgraduate School, Monterey, CA, USA, 2011.

[39] A. J. Healey, D. Horner, S. Kragelund, B. Wring, and A. Monarrez "Collaborative unmanned systems for maritime and port security operations," in *Proc. IFAC Conf. Control Adv. Marine Syst.*, Sep. 2007

[40] G. V. Avvari, D. Sidoti, M. Mishra, L. Zhang, B. K. Nadella, K. R. Pattipati, and J. A. Hansen "Dynamic asset allocation for counter-smuggling operations under disconnected, intermittent and low-bandwidth environment," in *Proc. IEEE Symp. Comput. Intell. Security Defense Appl.*, 2015, pp. 1–6.

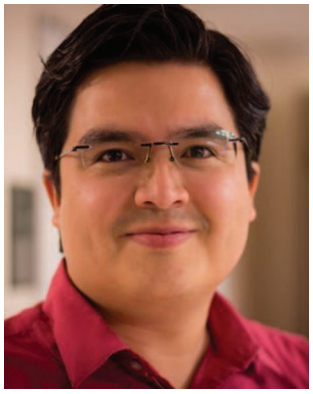


Lingyi Zhang received the B.S. and M.S. degrees in electrical and computer engineering from the University of Connecticut, Storrs, CT, USA, in 2014 and 2019, respectively. She is currently working toward the Ph.D. degree at the Department of Electrical and Computer Engineering at the same university. Her current research interests include modeling dynamic and uncertain environments for asset allocation and path planning, optimization-based techniques for mission planning and coordination, and adaptive Kalman filtering. She was the co-recipient of the Tammy Blair award for best student paper at FUSION 2016.



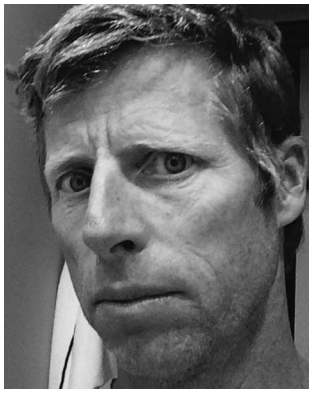
David Sidoti received the B.S., M.S., and Ph.D. degrees in electrical and computer engineering from the University of Connecticut, Storrs, CT, USA, in 2011, 2016, and 2018, respectively. He is currently a Computer Scientist with the Meteorological Applications Development Branch, U.S. Naval Research Laboratory, Marine Meteorology Division (MMD), Monterey, CA, USA. He is NRLMMD's primary machine learning subject matter expert and contributes to a broad portfolio of machine learning projects relating to atmospheric and oceanographic forecasting. His current interests include multiobjective algorithms for dynamic scheduling and resource management in weather-impacted environments, and deep learning applications to numerical weather prediction. He was the co-recipient of the Tammy Blair award for best student paper at FUSION 2016. In 2018, he was recognized and awarded a Distinguished Scholar Jerome and Isabella Karle's Fellowship.

Gopi Vinod Avvari received the B. Tech. degree in electronics and communication engineering from Acharya Nagarjuna University, Guntur, India, in 2011, and the M.S and Ph.D. degrees in electrical engineering from the University of Connecticut, Storrs, CT, USA, in 2016 and 2018, respectively. He is currently a Machine Learning Scientist with Aptiv, Agoura Hills, CA, USA. His research interests include developing algorithms for the advanced driver assisting systems and autonomous driving.



Diego F. M. Ayala (M'06) received the B.S. degree in electrical engineering from the Industrial University of Santander (Universidad Industrial de Santander), Bucaramanga, Colombia, in 2007, and the M.Sc. degree in electrical engineering from the University of Connecticut, Mansfield, CT, USA, in 2014, where he is currently working toward the Ph.D. degree in electrical and computer engineering. His current research interests include development of models and methods for synthesizing adaptive organizations and for planning in dynamic and uncertain environments, multiobjective optimization techniques for engineering applications, development of models of distributed information processing in human teams, Bayesian networks, Markov processes, and machine learning in the context of big data analysis, and renewable energy technologies with SMART grid applications.

Manisha Mishra received the B.S. degree in electrical engineering from Punjab Technical University, Jalandhar, India, in 2004, the M.S. degree in electrical and computer engineering from the University of Hawaii, Honolulu, HI, USA, in 2008, and the Ph.D. degree in electrical and computer engineering from the University of Connecticut, Storrs, CT, USA, in 2017. She was a Software Engineer with Infosys Technologies, India, in 2005. She was an ORISE Post-Graduate Researcher with U.S. Army Engineering Research and Development Center—Cold Region Research Labs in 2017. She is currently an Algorithm Engineer with Aptiv Corporation. Her research interests include modeling dynamic and uncertain environments for asset allocation and path planning, context-aware decision support systems, risk analysis, system diagnosis and prognosis, and optimization-based techniques for mission planning and coordination.



David L. Kellmeyer received the M.S. degree in human factors engineering from Ohio University, Athens, OH, USA, in 1992, where his research focused on highway safety and artificial intelligence. He is currently a Human Factors Engineer and Project Manager with the User-Centered Design (UCD) Team, SPAWAR Systems Center—Pacific, San Diego, CA, USA (SSC-Pac). He served 4 years with the Army Medical Service Corp providing industrial hygiene and ergonomics support for the Center for Health Promotion and Preventive Medicine and the National Security Agency (NSA). During the past 20 years with SSC-Pac, he has led projects focusing on the need to develop human–computer interfaces that reduce workload and support better supervisor control and situational awareness. He has supported multiple industry teams to include Arsenal Ship, DD 21, and the Unmanned Combat Aerial Vehicle (UCAV). He has served in three off-site tours in Washington, DC, USA, supporting the Department of Homeland Security, the Transportation Safety Administration (TSA), and the Office of Naval Research. He was the lead designer for the Joint Inter-Agency Task Force—South's (JIATF-South) new command center. During the last 5 years, he has led two Office of Naval Research projects that transitioned supervisory control technologies to the Submarine Combat Office and the Navy's Command and Control program office. He has worked for numerous Department of Defense and government agencies, including The White House, Pacific Command, Strategic Command, NSA, TSA, Department of Homeland Security, and JIATF-South.

James A. Hansen received the undergraduate and master's degrees in aerospace engineering from the University of Colorado Boulder, Boulder, CO, USA, in 1992 and 1993, respectively, and the doctoral degree from the University of Oxford, Oxford, U.K., in 1998. He was a Rhodes Scholar with Atmospheric, Oceanic, and Planetary Physics Department, University of Oxford. He was an Associate Professor with the Earth, Atmospheric, and Planetary Sciences Department, Massachusetts Institute of Technology, Cambridge, MA, USA. He is currently the Superintendent of the Marine Meteorology Division, Naval Research Laboratory (NRL), Washington, DC, USA. He has been with NRL, since 2006, and during that time he has been the Lead Scientist of the NRL Probabilistic Prediction Research Office and the Head of the Meteorological Applications Development Branch. He has authored or coauthored dozens of peer-reviewed publications. His current research interests include estimation of environmental forecast uncertainty and the use of that uncertainty in decision making. He was a recipient of numerous awards and honors, the most recent of which are the Navy Meritorious Civilian Service Award for meritorious performance of service and the A. S. Flemming Award for Exceptional Federal Service. He was a Research and Development Lead in the Piracy Attack Risk Surface Project—a now operational system for coupling environmental, intelligence, and behavioral factors to predict the risk of pirate attack. He serves on several scientific advisory boards and was an Editor for *Monthly Weather Review*.

Krishna R. Pattipati received the B.Tech. degree in electrical engineering with highest honors from the Indian Institute of Technology, Kharagpur, India, in 1975, and the M.S. and Ph.D. degrees in systems engineering from the University of Connecticut, Storrs, CT, USA, in 1977 and 1980, respectively. He was with AlphaTech, Inc., Burlington, MA, USA, from 1980 to 1986. He has been with the Department of Electrical and Computer Engineering, University of Connecticut, where he is currently the Board of Trustees Distinguished Professor and the UTC Chair Professor in Systems Engineering. His research activities are in the areas of proactive decision support, uncertainty quantification, smart manufacturing, autonomy, knowledge representation, and optimization-based learning and inference. A common theme among these applications is that they are characterized by a great deal of uncertainty, complexity, and computational intractability. He is a co-founder of Qualtech Systems, Inc., a firm specializing in advanced integrated diagnostics software tools (TEAMS, TEAMS-RT, TEAMS-RDS, TEAMATE), and serves on the board of Aptima, Inc. He was selected by the IEEE Systems, Man, and Cybernetics (SMC) Society as the Outstanding Young Engineer of 1984, and received the Centennial Key to the Future award. He was the Editor-in-Chief of the IEEE Transactions on Systems, Man, and Cybernetics—Part B from 1998 to 2001, Vice-President for Technical Activities of the IEEE SMC Society from 1998 to 1999, and Vice-President for Conferences and Meetings of the IEEE SMC Society from 2000 to 2001. He was co-recipient of the Andrew P. Sage Award for the Best SMC Transactions Paper for 1999, the Barry Carlton Award for the Best AES Transactions Paper for 2000, the 2002 and 2008 NASA Space Act Awards for "A Comprehensive Toolset for Model-Based Health Monitoring and Diagnosis," and "Real-Time Update of Fault-Test Dependencies of Dynamic Systems: A Comprehensive Toolset for Model-Based Health Monitoring and Diagnostics," and the 2003 AAUP Research Excellence Award at UCONN. He also won the best technical paper awards at the 1985, 1990, 1994, 2002, 2004, 2005, and 2011 IEEE AUTOTEST Conferences, and at the 1997 and 2004 Command and Control Conferences. He is an elected Fellow of IEEE and of the Connecticut Academy of Science and Engineering.