JOURNAL OF AEROSPACE INFORMATION SYSTEMS
Vol. 14, No. 1, January 2017

Optimized Stochastic Coordinated Planning of Asynchronous Air and Space Assets

Eric Robinson*
Joint Base Pearl Harbor-Hickam, Honolulu, Hawaii 96853
Hamsa Balakrishnan†
Massachusetts Institute of Technology, Cambridge, Massachusetts 02139
and
Mark Abramson‡ and Stephan Kolitz§
Charles Stark Draper Laboratory, Inc., Cambridge, Massachusetts 02139

DOI: 10.2514/1.I010415

There are many organizations that use satellites and drones to collect information, such as ground photographs or atmospheric pressure measurements. Often, these separate organizations have overlapping collection interests, yet they are controlled by separate planning systems with asynchronous scheduling cycles. This paper develops a method for coordinating various collection tasks between the planning systems to increase the overall utility of the collected data. The method focuses on allocation of collection requests to scheduling systems rather than complete centralized planning over the entire system so that the current planning infrastructure can be maintained. Previous work in this area is expanded upon by inclusion of an online learning method to capture information about the uncertainty pertaining to the scheduling and completion of collection tasks, which is subsequently used in a mathematical programming method for resource allocation. An analysis of results and improvements as compared to current operations is presented at the end through a few different theoretical scenarios. These results provide evidence that the newly developed methods can increase the total value of serviced requests compared to current operations, with some theoretical scenarios producing more than double the value using the new methods over the current styles of planning.

Nomenclature

l	=	arbitrary planner
r	=	arbitrary request
v_{rl}	=	value obtained if request r is completed by planner l
$\boldsymbol{x}$	=	arbitrary vector of real-valued statistical attributes (used in Sec. IV)
$\boldsymbol{x}_i$	=	specific instance of the attribute vector $\boldsymbol{x}$ (used in Sec. IV)
x_{rl}	=	decision variable associated with sending request r to planner l during a certain execution period
$\boldsymbol{x}^k$	=	user-defined instance of the attribute vector $\boldsymbol{x}$ for creating Bayesian prior beliefs (used in Sec. IV)
y	=	arbitrary binary statistical observation (used in Sec. IV)
y_i	=	specific instance of the binary observation y (used in Sec. IV)
y_{rG}	=	composite decision variables for sending r to each planner $l \in G$
$\boldsymbol{\lambda}$	=	vector of Bayesian parameters (used in Sec. IV)

I. Introduction

CURRENTLY, there are many organizations in the United States that use unmanned assets, such as satellites or drones, to obtain information. This may include taking photographs of the ground, gathering infrared photographs, taking atmospheric pressure measurements, or any conceivable form of data collection. Often, these separate organizations have overlapping collection interests or flight plans that are sending sensors into similar regions. Exploiting such common interests between the organizations could potentially allow asset sensing time to be used elsewhere. Unfortunately, different organizations often have command and control stations for their assets that are spread across the nation. Even within an organization, separate missions might control their assets from different locations. This separation can make exploitation of common collection interests a nontrivial task because it might be difficult to gather all of the representatives for a meeting or even a telephone conference. Each of these organizations also has different objectives: a problem that further isolates and complicates mutual correspondence. An automated, coordinated approach to assigning collection tasks among these segregated missions/organizations, or "stovepipes," has the potential to significantly reduce the problem of isolation in order to increase the realized utility of the collections made by the assets.

Organizations collect data for various purposes. The National Aeronautics and Space Administration (NASA) collects data for scientific research; the U.S. Department of Defense (DOD) collects data for intelligence, surveillance, and reconnaissance. Often, these organizations use their own air and/or space assets in order to best manage the collection parameters: time, location, desired level of resolution, and type of data being obtained (infrared, visible light, atmospheric pressure, etc.) for their targets of interest. Even within organizations (e.g., branches of the DOD or NASA), there may be separate, generally unshared, collection assets. The stovepiped nature of these planners prevents cross communication

Presented as Paper 2013-4815 at the AIAA Infotech@Aerospace (I@A) Conference, Boston, MA, 19–22 August 2013; received 25 July 2015; revision received 13 July 2016; accepted for publication 23 September 2016; published online 11 January 2017.

*Executive Officer and C-17A Pilot, 15th Operations Support Squadron, 800 Hangar Ave Bldg. 2050. Member AIAA.
†Associate Professor, Aeronautics and Astronautics, 77 Massachusetts Ave. 33-328. Associate Fellow AIAA.
‡Principal Member of the Technical Staff, Information and Decision Systems, 555 Technology Square. Member AIAA.
§Distinguished Member of the Technical Staff, 555 Technology Square. Associate Fellow AIAA.

among separate sensing assets, creating inefficiencies that could be avoided through coordinated planning. Such isolation between operations of separate planners means that requests may not be allocated to the most appropriate sensors based on their specific requirements.

Studies requiring sensing assets have become increasingly more prevalent in the past few years, suggesting a more urgent need for a coordinated planning framework. One of the major recent uses of sensor systems involves employing satellites or unmanned aerial vehicles (UAVs) to study Earth's climate. The Earth Observing System, designed by NASA, is one such example of a set of sensors being used for climatology. Within this system, NASA launched many satellites. The Suomi National Polar-Orbiting Partnership (NPP) satellite is one of these, launched in October 2011. This satellite orbits Earth about 14 times per day, observing nearly the entire surface in this time period. The sensors onboard NPP perform many different climate-related operations, such as creating global models of temperature and moisture profiles for use by meteorologists, monitoring the ozone levels near the poles, measuring atmospheric and oceanic properties, and examining both emitted and reflected radiation from Earth's surface [1]. The Aqua satellite, also in the NASA Earth Observing System, collects information about the following:

> Earth's water cycle, including evaporation from the oceans, water vapor in the atmosphere, clouds, precipitation, soil moisture, sea ice, land ice, and snow cover on the land and ice. Additional variables also being measured by Aqua include radiative energy fluxes, aerosols, vegetation cover on the land, phytoplankton and dissolved organic matter in the oceans, and air, land, and water temperatures [2].

These are just two examples of the many satellites currently in orbit collecting climate-related information, all of which have some sort of overlapping interests and may even contain some of the same sensor models. As such, implementing a coordinated planning scheme within satellite planners of the Earth Observing System, or any climate-related satellites, could prevent redundant gathering of the same data while spreading collection demands more evenly across the satellites for more effective sensor utilization.

Recent interest in examining natural disasters has also increased, furthering the need to efficiently coordinate between sensor planners. The Hurricane and Severe Storm Sentinel, which is a NASA investigation designed to enhance understanding of the "processes that underlie hurricane formation and intensity change in the Atlantic Ocean basin," is one such example of a mission trying to learn more about natural disasters [3]. The U.S. Forest Service has also recently been employing UAVs and satellites to help image active wildfires, reducing the risks of "smoke, excessive thermal wind drafts, and unfamiliar terrain" on the pilots that usually do the imaging in airplanes or helicopters [4].

Science and forestry are not the only areas that could benefit from coordinated planning. The intelligence and reconnaissance communities use a tremendous amount of autonomous vehicles and sensing assets to complete missions. Indeed, the concept of coordination is already recognized as being important; according to the Joint Doctrine for Targeting, which defines how targets for remote sensing should be created and collected, a "primary consideration" for developing targeting plans "is the joint force's ability to coordinate, deconflict, prioritize, synchronize, integrate, and assess joint targeting operations" [5]. Clearly, a coordinated planning framework is in line with this objective, and it certainly could improve the overall utility of sensing data collected for the intelligence and reconnaissance communities.

In essence, coordinated planning can be implemented within any situation that requires the use of advanced sensor systems that exist in satellites, unmanned aerial vehicles, underwater vehicles, or ground vehicles. A coordination planner (CP) could be created inside of a web service for users to easily upload requests online, using the standards set forth in [6], overcoming the geographical separation problem of the current stovepiped operations. A CP web service could also help fulfill the concept of a sensor web, which "consists of intra-communicating, spatially-distributed sensor pods that are deployed to monitor and explore environments" so that "information gathered by one pod is shared and used by other pods" [7], or even to create a partial sensor web in which only a subset of the data is shared across the sensing assets if that is more preferable.

The initial groundwork for developing a CP was performed by Thomas Herold in [8]. Herold went into depth describing the operational concept of the CP in a real-world context, which he used to motivate development of the coordinated planning problem. Using this description, Herold provided a linear programming formulation to address a deterministic scenario of request allocation across the mission planners.

Related analysis pertaining to centralized planning of multiple viewing assets (satellites, UAVs, etc.) has been widely studied. In [9], Sakamoto considered the problem of efficiently planning missions for a group of UAVs in a centralized manner. He proposed a robust mixed-integer programming formulation in order to create UAV mission plans that had a high likelihood of being feasible in a stochastic environment. In [10], Blair Negron solved a very similar problem, planning missions for multiple UAVs given a set of tasks in three dimensions. Negron solved a very general problem, including time windows, observation duration, and location information for each task, as well as maximum altitude, minimum altitude, endurance, and travel time between locations as inputs for the UAV. By including such a large amount of generality in her model, the resulting mathematical programming formulation that Negron developed became inefficient for large applications. To fix this issue, Negron developed a metaheuristic that created mission plans very efficiently without sacrificing much value from optimality, thereby allowing quick solutions, even for very large problems.

In [11], the authors approached control of unmanned assets for a wide array of tasks (search, target classification, attack, damage assessment, etc.). The solution approach used a hierarchical division of the problem into multiple layers of control. The authors constructed and simulated an auction-based formulation to determine how to best assign tasks to various groups of vehicles. A main insight was that allowing multiple assets to cooperate on a single task provided better global results. However, the hierarchical method employed for the control of UAV task assignment and completion in [11] still addressed a version of collection planning in which all of the agents worked together toward the same overall objective, and not for the objectives of stovepiped planners.

The authors of [12] solved a problem of completing a large set of tasks with a small number of UAVs by using a mathematical program for centralized assignment of tasks to assets, and then creating a separate scheduling algorithm to decide the paths taken by the individual assets. The main two differences between this type of problem and the coordinated planning problem were as follows:

1) The coordinated planning problem assigned tasks to planners, and not to individual assets.
2) The coordinated planning problem allowed planners to have their own scheduling/control algorithms for their assets.

Although these papers provided insights about other potential solution approaches for the coordinated planning problem, they all focused on highly centralized planning that did not take into account the issue of planning for asynchronous, distributed systems (i.e., stovepipes).

Work has also been performed relating to efficient tasking of satellite and UAV assets. The authors of [13] developed a tool for the centralized planning of many target points for which a large number of satellites were available. The authors split the decisions hierarchically: first, assigning tasks to satellites, and then separately planning the task start and end times for each satellite to maximize the total value of the targets obtained. The authors of [14] addressed the uncertainty inherent in the planning of photographs taken by a single satellite, using a mathematical programming formulation that was motivated by a Markov decision process. Their model considered the probability that a photograph would actually be completed if it was incorporated into the schedule for the current day, as well as the probability that the photograph would be selected and subsequently completed for a future day under a given policy, to design a schedule that maximized the total expected value of realized photographs subject to any feasibility constraints. The authors suggested that these probabilities could be determined by simulation or could be adaptively learned in an online manner through a machine-learning-based approach. They recommended the learning approach because it allowed the formulation to adapt to changes in the system over time. A more general problem of dynamically assigning abstract resources to tasks over time was considered in [15]. The problem, coined by the authors as "the dynamic assignment problem," was solved using a combination of network optimization and approximate dynamic programming techniques.

This paper presents a method for developing a CP that focuses on allocation of collection requests to scheduling systems rather than complete centralized planning over the entire system so that the current planning infrastructure can be maintained. This method expands on the previous work done in [8] by inclusion of a learning method to capture information about the uncertainty involved in data collection from such assets, which are subsequently used in a mathematical programming method for resource allocation. With the information gained using online learning, this mathematical programming formulation has the ability to model many tradeoffs pertaining to the assignment of collection tasks to planning systems. This solution technique is tested on a few potential scenarios, the results of which indicate that the models can produce vastly improved utility in the sensing collection results obtained over the current stovepiped approach, whether "utility" is number of requests completed, the average priority of the requests (in a system where certain requests are more important than others), or another objective.

II. Problem Definition and Motivation

A. Terminology

Conceptually, the coordinated approach is instantiated in the aforementioned CP, which uses an algorithmic method for analyzing the parameters of various collection tasks within a set of planners (i.e., missions or organizations) in order to determine the best allocation of collection requests to planners. By performing this analysis and allocation, the CP effectively increases the ability of the assets on each of the stovepiped planners to complete all collection requests. At the highest level is the coordination system interface. This is the actual computer program, person, or other device that collects requests from users who wish to use the CP technology. In this context, a request is defined as the collection specification, input into the CP by some user, for obtaining a piece of data (ground image, infrared image, climate data, etc.) at a given location, during a specific time window, using sensing assets (motivated by the definition of a request found in [10]). A planner is defined as a function that takes requests as inputs to produce an operation schedule for an asset or set of assets, such as UAVs, airplanes, satellites, ground vehicles, or underwater vehicles. Each planner has its own planning cycle, not necessarily synchronized with the other planners, which consists of planning, upload, and execution phases (see Fig. 1).

The coordinated planning problem, which we analyze in this paper, involves coordinating collection requests between stovepiped missions to increase the overall utility of the system, without forcing the missions to significantly alter their planning systems. The CP first takes user-defined requests and prior information about collection interests from the participating planners as inputs in order to determine efficient pairings of requests with planners. The CP then must send its own versions of these requests to the planners that it considers in its system, separately asking each one to complete some subset of the user-defined requests as determined by the pairings. It is imperative that these requests be sent during the appropriate planning phases for each planner, so we assume that the CP is aware of the planning cycles for each planner.

The CP builds up a queue of requests over time, which is the set of all user-defined requests that have not yet been completed. The CP reviews this queue periodically to create pairings and coordination requests, which we refer to as the CP iteration. The time length of this review is referred to as the CP iteration length. To ensure that the CP always has at least one opportunity to send requests during each planning period of a given planner, the CP iteration length is assumed to be shorter than the lengths of the planning periods for all of the individual planners. This assumption could be relaxed in reality if needed, but the cost would be that the CP may not have the opportunity to send requests for some execution phases on individual planners.

The first step in reviewing the CP queue involves employing an opportunity finder, which is a filter to determine feasible pairings of requests to planners. For example, consider a system with three planners: one associated with a UAV that flies over the Atlantic Ocean, and two associated with separate satellites. The planning cycles of these three planners would resemble that shown in Fig. 1 (note that "SAT1" and "SAT2" are short for "Satellite 1" and "Satellite 2"). A CP sending requests to these three planners would have knowledge of these planning cycles and would systematically determine pairings of requests to the planners based on its knowledge of entire system; however, the planners themselves would maintain complete control of their assets. If two requests were to be submitted to this notional system, one for a visual photograph in Nevada and the other for a point elevation reading in Germany, the opportunity finder would filter out the possibility of pairing the visual photograph in Nevada with the planner for the Atlantic Ocean UAV, or pairing the point elevation request with a satellite that does not have an appropriate sensor (e.g., light detection and ranging) to complete the task. This opportunity finder finds all pairings that have a positive probability of being completed, and it removes all other pairings. A version of the generic opportunity finder mentioned in [8] is used for the evaluation in this paper; however, in-depth analysis into specific capabilities, such as incorporating any knowledge about current plans to exploit piggybacking opportunities, is left to future research.

The next step involves choosing the best pairings, and subsequently sending the coordination requests to the planners. Each of these coordination requests can then be accepted, rejected, completed, or failed by the individual planners; we call this the status of the coordination requests. Each of these terms is defined as follows:

1) Accepted indicates that a planner has selected the given coordination request to be included in the planned schedule for its asset(s).
2) Rejected indicates that a planner has decided not to include the given coordination request in the planned schedule for its asset(s).
3) Completed indicates that the given coordination request has been fulfilled by a planner and the data are ready for delivery to the user (this implies that the request must first have been accepted by this planner).
4) Failed indicates that the given coordination request could not be completed due to unforeseen circumstances, even though it was accepted by the planner.

To choose the most appropriate pairings, a few modeling assumptions about the level of interaction between the planners and the CP must be made.

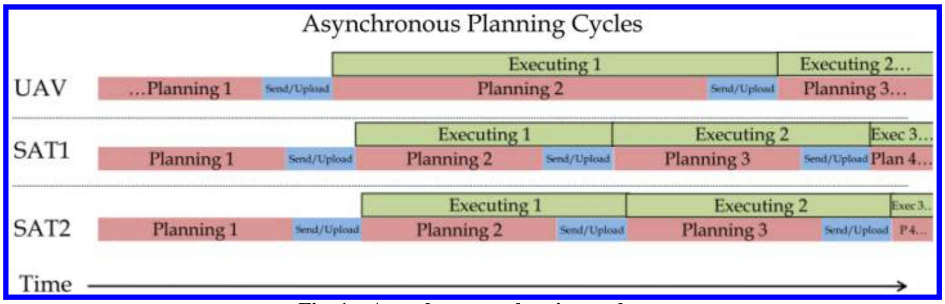

Fig. 1 Asynchronous planning cycles.

1. Assets

Each planner has assets for which the capabilities are known to the CP in advance. In addition, the planners either provide their own opportunity finder via a web service to inform the CP of request feasibility during an execution phase or the CP has enough knowledge about the planner's assets to be able to create its own opportunity finder.

2. Capacity

There exists a maximum number of requests that can be received by each planner in a given execution phase. In most cases, this should be the result of a predefined contract between the CP and the planners (to satisfy data storage or other limitations inherent to each planner), as is the case for sensor planning services following the specification in [6]. However, even planners that do not explicitly limit the number of CP requests still have a practical saturation limit that can be modeled by a capacity, ensuring that the planners are only tasked with appropriate requests. In this latter case, the saturation limit could be determined by historical data analysis to create an initial estimate of the practical saturation limit, or it could be imposed spatially (e.g., by limiting the mean distance from the center of a request cluster). For this paper, it is assumed that all capacities are already known, either through a contract or prior historical analysis. By imposing these capacity constraints, we can reasonably assume that, given the general location in which an asset will be collecting sensing data, the probability that a planner will complete a given request is negligibly influenced by the other coordination requests being sent to that planner. This allows for the reasonable independence assumption that, if requests $r_1, \ldots, r_k$ are all sent to planner l, then the set of probabilistic events of the form "request r_i is completed by planner l" for all $i \in \{1, \ldots, k\}$ form an independent set of events, as do the events "request r_i is accepted by planner l" for all $i \in \{1, \ldots, k\}$, conditioned on the known capacity being obeyed.

3. Reservation Fee

A given planner l might impose a "reservation fee" in dollars of $c_l > 0$ per coordination request that it considers. These funds are assumed to be collected upon submission of the request, although they might be refundable at a later time if the planner does not complete the request. We assume that there is a set budget of funds available to the CP at each iteration for the intent of satisfying necessary reservation fees.

B. Information Flow

General-purpose collection management requires a large flow of information between users and planners. The coordinated system that we use allows users to input requests to the coordination system interface rather than directly to the planners. It can also be used via a web service to give users an approximate probability that their request(s) will be completed by various planners. The CP does not actually perform any of the scheduling for the individual planners, nor does it alter any of the current infrastructure. Rather, it adds to what already exists, allowing planners to have asynchronous planning cycles.

A single iteration of coordinated planning consists of two phases: the information-gathering phase and the coordinated planning phase. The information-gathering phase begins at periodically spaced epochs in time, where the period between epochs is the aforementioned CP iteration length. This phase continues until the next epoch, when a new coordinated planning phase and information-gathering phase are initiated. This coordinated planning phase ends when the coordination requests for the current iteration have been sent to planners, and it is, in general, much shorter than the information-gathering phase. This process is illustrated in Fig. 2. Note that the information-gathering and coordinated planning phases overlap such that the ith coordinated planning phase starts simultaneously with information-gathering phase $i + 1$, although the coordinated planning phase is shorter.

During the information-gathering phase, users input their requests to the CP via the coordination system interface, which adds the requests to the queue. Also, during this phase, the CP receives feedback from the individual planners pertaining to the status of previously assigned coordination requests. Any request that is completed during this phase is removed from the queue, and the collected data are made available to the appropriate users. All other notifications (i.e., accepted, rejected, and failed) are parsed into data that are stored for later use. It is important to note that requests that have not yet been completed are not removed from the queue until they are either completed by a planner or the user-defined observation time window expires (at which point, the request is removed and the user is informed that the data were not collected). This allows for the CP to resend the same request to any feasible planner at multiple points in time if practical. In our example, this could be represented by assuming that the point elevation request is not urgent, and therefore the user has given a two-day time-window for completion. Thus, if the CP tasked the planner corresponding to a low-Earth-orbit satellite with collecting the point elevation early on the first day but the satellite was unable to complete the collection at that time, the request could be resent to that planner to reattempt during a later pass.

We assume that planners return information about accepted/rejected coordination requests at some time after the coordination request has been sent but before the execution phase begins for the appropriate request/planner pairing. We do not make any explicit assumptions within our mathematical model as to when planners inform the CP about completed/failed requests, other than it has to be after the execution phase has ended for the appropriate pairing. However, for analysis purposes, we will only consider scenarios in which the CP is informed of completions and failures immediately following the end of the appropriate execution phases. This models the idea that planners should desire to make this information available as quickly as possible.

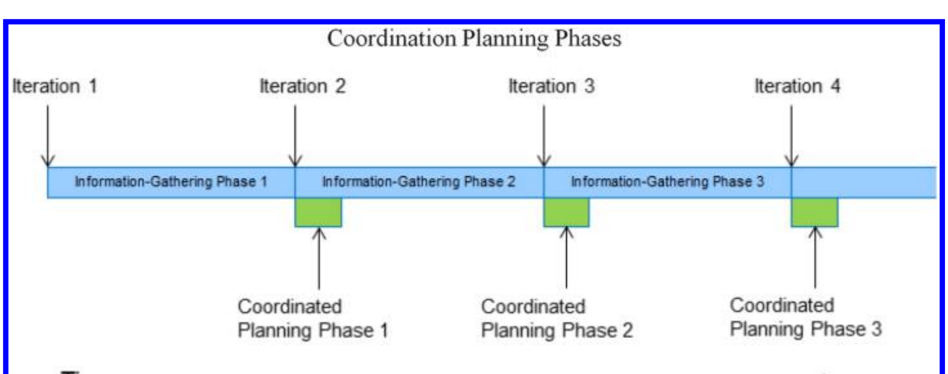

Fig. 2 Phases within a coordinated planning iteration.

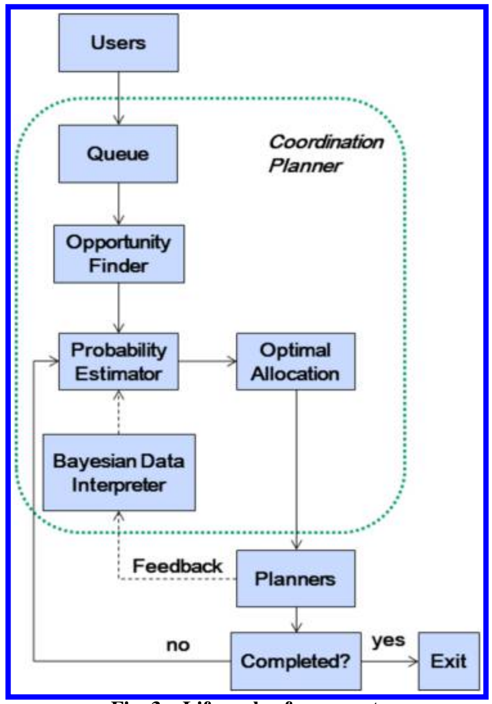

Fig. 3 Life cycle of a request.

During the coordinated planning phase, all of the data are reviewed in order to send requests to planners. This process begins by passing all requests in the queue through the opportunity finder to determine feasible pairings of requests to planners. Then, a probability estimator reviews all of the stored data pertaining to the results of prior coordination requests (i.e., the defining attributes of the coordination requests, as well as whether they were accepted/rejected, completed/failed, and by which planner) to create probability estimates for acceptance and completion of the feasible pairings of requests to planners. This information is sent to an optimization algorithm that determines efficient pairings relative to some predefined utility (e.g., number of requests completed), and then these pairings are forwarded to the appropriate planners, allowing for the possibility of a request to be sent to multiple planners if desired. This process of request movement is depicted pictorially in Fig. 3. In this flowchart, the steps associated with the coordination planning phase are inside of the dotted line, and the information-gathering steps are outside of this line.

Continuing with the example from earlier, the opportunity finder would have filtered out the visual photograph from going to the UAV planner during the coordinated planning phase. Supposing that both of the satellites had orbits that passed over Nevada, as well as appropriate sensors, the opportunity finder would have identified the photograph request to satellite pairings as being feasible. The probability estimator would then provide the CP with an approximate probability that the planner for each satellite would accept the request and task their respective asset to take a visual photograph over Nevada. The estimator would also give an approximation for the chance that the visual photograph would be successfully collected by the satellite, given that it had already been accepted, and an estimate of the future probability of the request being sent again if needed. Using this information, an optimization would be run by the CP to determine which of the two satellites should be sent the visual photograph request based on capacity and budget constraints, described in Sec. II. If it was determined that only the first of the two satellites should receive the request; then, the CP would forward the visual photograph request to the planner for the first satellite, ready to be considered during that planner's next execution phase.

At this point, the CP would begin its information-gathering phase. The satellite planner would run its own internal process to determine its willingness/ability to incorporate that photograph request into its plans. It would then inform the CP of its decision to accept/reject the photograph at some point before the start of the respective execution phase, which the CP would use to determine if it should resend the request to increase its chances of completion. After the execution phase, the satellite planner would promptly return the desired visual photograph or inform the CP that the request was unable to be collected (e.g., if there was excessive cloud cover, or if the sensor failed). If the photograph was completed, it would be removed from the CP queue and returned to the user, who could then accept the product as final or resubmit for another collection if desired. If the satellite was unable to complete the photograph or chose not to accept it, then the CP would keep that request in its queue until the next coordinated planning phase.

In the current stovepiped system, users input requests directly to the planners, and planners return completed data requests directly to the users (see Fig. 4). As already mentioned, stovepiped systems present a vast array of problems and inefficiencies in collection management. The lack of communication in these systems forces users to bear the burden of finding the best possible sensor(s) to use for their specific requests, which can lead to users trying to locally optimize their schedules without regard for others. Users may also be unaware of the benefits of certain assets, or simply not have the personnel contacts to use other assets that would be well suited for their tasks. Some other major disadvantages of stovepipes include their inability to efficiently find piggybacking opportunities (i.e., chances to add their requests onto the previously scheduled plans of a different asset that may be operating in a desirable location) or from pooling requests between various users to find a more efficient allocation of requests to sensors.

All of these difficulties suggest that a highly centralized planning system could be quite useful. Although there are some domains in which it makes sense to implement this type of system, often fully centralized planning is highly impractical. We will rarely be able to implement a centralized planning system, because this would require a complete overhaul of the existing planning infrastructure. Computer interfaces between assets/planners would need to be redesigned, and headquarters of such mission planners would likely need to be relocated so that the operators of planners/assets could work together. The individual planners would have to give up their respective planning cycles. For these reasons, we do not focus on centralized systems, even if theoretically more efficient.

A coordinated planning scheme has the potential to maintain many of the benefits of both stovepiped and centralized planning systems while eliminating their problems. With coordination, we eliminate the lack of communication inherent to the current system of stovepipes by providing a single, automated platform that can interface with each planner to send requests and receive information. Planners are given the liberty to choose

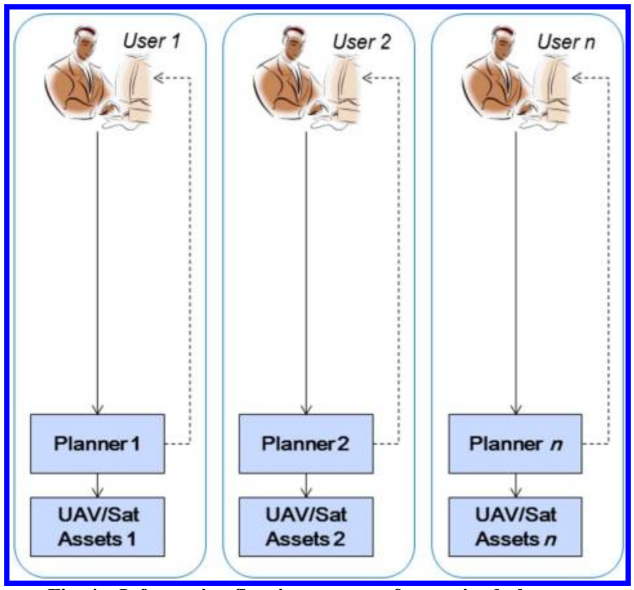

Fig. 4 Information flow in a system of stovepiped planners.

their own planning algorithms and control their assets, and they are even allowed to set their schedules before considering any coordination requests. By coordinating, we can eliminate the communication gap between stovepipes without implementing the impractical restrictions on the planners that come attached to a centralized system.

III. Mathematical Model

Using the problem definition and background from the previous section, a formal method for request assignment in a single coordinated planning phase can be developed. For the remainder of this section, all notation refers to a specific coordinated planning phase unless otherwise noted. Various sets associated with the state of the coordinated planner are defined in Table 1. Each of these sets is defined in terms of its status at the beginning time t_{plan} of the specific coordinated planning phase being considered. In addition, a few other assumptions are as follows:

1) Each pairing (r, l) of a request r to a planner l has an inherent predicted value of $v_{rl} > 0$ if r were to be completed by l, but not by any other planners.

2) Planner l will only consider a request r for the execution phase associated with the planning phase of l that is currently active at time t_{plan}. If no planning phase is currently active on l (i.e., during the send/upload phase), then the request can still be paired to that planner, but l will consider it for the execution phase associated with the first planning phase following t_{plan} (see Fig. 1 for a visual depiction of various planner phases). In either case, the execution phase for planner l being considered by the CP at time t_{plan} is denoted d_l^{next}. Note that decisions of this type will not be finalized until the last CP iteration before d_l^{next} because, by construction, requests are not sent until the send/upload phase.

3) There are no limits on the number of planners to which the request can be sent in any single coordinated planning phase.

4) Only the "best" completed pairing yields value to the user, so the utility of a past request r is defined to be the maximum of the set of all values v_{rl} such that a coordination request associated with r is completed by planner l.

5) The total utility is the sum of the individual request utilities for each request r in the queue.

When deciding where to send coordination requests, we do not know which ones will be completed. The objective of the coordinated planning phase is therefore to produce pairings (r, l) that yield the maximum expected total utility, requiring the existence of a well-defined and meaningful probability distribution for "total utility." To show that such a distribution exists, and subsequently to derive its expectation, we introduce random variables Z_{rl} such that $Z_{rl} = v_{rl}$ if, and only if, a coordination request associated with r is completed by planner l during execution phase d_l^{next} or any thereafter; otherwise, $Z_{rl} = 0$. Following the integer programming methodology described in [16], we introduce integer decision variables x_{rl} where $x_{rl} = 1$ if, and only if, the CP chooses to send request r to planner l for consideration during execution phase d_l^{next}; otherwise, $x_{rl} = 0$. Define $q(r, l, x_{rl})$ to be the probability that request r will not be completed by planner l during execution phase d_l^{next} or any thereafter, parametrized as a function of the decision x_{rl}. Then, the probability mass function for an arbitrary Z_{rl} can be written as

$$P(Z_{rl} = z) = \begin{cases} 1 - q(r, l, x_{rl}) & \text{if } z = v_{rl} \\ q(r, l, x_{rl}) & \text{if } z = 0 \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

Table 1 Set definitions

Set	Definition	Arbitrary element
R	Set of all requests r in the CP queue at time t_{plan}	Some request r
L	Set of all individual planners l in the coordinated system at time t_{plan}	Some planner l
R_l	Set of all requests r such that (r, l) is a feasible pairing at time t_{plan}, determined by the opportunity finder	Some request r
L_r	Set of all planners l such that (r, l) is a feasible pairing at time t_{plan}, determined by the opportunity finder	Some planner l
RFE_{rl}	Set of all remaining known feasible execution phases d for request r on planner l at time t_{plan}	Some execution phase d
S_{rl}	Set of all execution phases d in which a coordination request associated with r has already been sent to planner l at time t_{plan}	Some execution phase d

Using the assumptions mentioned in Sec. II that the events "request r_i is completed by planner l" for all $i \in \{1, \ldots, k\}$ form an independent set of events, as do the events "request r_i is accepted by planner l" for all $i \in \{1, \ldots, k\}$, conditioned on the known capacity being obeyed, and assuming (for simplicity) that the decisions of future coordinated planning phases can be approximated as being mutually independent, the quantity $q(r, l, x_{rl})$ can be expanded as

$$q(r, l, x_{rl}) = \left(1 - x_{rl} \times P_a(r, l, d_l^{\text{next}}) \times P_c(r, l, d_l^{\text{next}})\right) \times \left[\prod_{d_l \in \text{RFE}_{rl}} \left(1 - P_s(r, l, d_l) \times P_a(r, l, d_l) \times P_c(r, l, d_l)\right)\right] \times \left[\prod_{d_l \in S_{rl}} \left(1 - P_a(r, l, d_l) \times P_c(r, l, d_l)\right)\right] \tag{2}$$

where

1) $P_s(r, l, d_l)$ is the probability that request r will be sent to planner l for consideration during execution phase d_l, conditioned on x_{rl} and the current status of the queue at time t_{plan}.

2) $P_a(r, l, d_l)$ is the probability that request r will be accepted by planner l for incorporation into the plans of execution phase d_l, given that r is already sent to l.

3) $P_c(r, l, d_l)$ is the probability that request r will be completed by planner l during execution phase d_l, given that r is already sent to, and accepted by, planner l.

4) RFE_{rl} and S_{rl} are as defined in Table 1.

The functions $P_s(r, l, d_l)$, $P_a(r, l, d_l)$, and $P_c(r, l, d_l)$, referred to as probability estimators, are assumed to be known at time t_{plan} (a method for online learning of these functions is presented in the next section). In addition, once a coordination request for the pairing (r, l) during execution phase d_l is sent or accepted, we update $P_s(r, l, d_l) = 1$ or $P_a(r, l, d_l) = 1$, respectively, for that pairing (completed requests are removed from the queue). Using this model and the notation from Table 1, the utility of request r is a random variable V_r where

$$V_r = \max_{l \in L_r} \{Z_{rl}\}$$

so the expected utility of request r is

$$E\{V_r\} = E\left\{\max_{l \in L_r} \{Z_{rl}\}\right\}$$

To calculate this expectation, the set L_r is sorted into an indexed set $\tilde{L}_r$ ordered according to increasing values v_{rl} for all $l \in L_r$ so that, if $\tilde{L}_r(i)$ denotes the ith planner of this set for all $i \in \{1, 2, \ldots, |\tilde{L}_r|\}$ and $v(i)$ the value of the associated request/planner pairing $(r, \tilde{L}_r(i))$, then $v(i) \le v(j)$ for all $i \le j$. From Eq. (1), the value $Z_{r,\tilde{L}_r(i)}$ associated with the ith pairing $(r, \tilde{L}_r(i))$ can only take the value $v(i)$ or zero, implying that V_r must take a value from the set $\{0, v(1), \ldots, v(|\tilde{L}_r|)\}$ because

$$V_r = \max_{l \in L_r} \{Z_{rl}\}$$

Define C_i to be the event where request r is completed by planner $\tilde{L}_r(i)$ but not by any $\tilde{L}_r(j)$ satisfying $i < j$. By construction of C_i, $Z_{r,\tilde{L}_r(j)} = 0$ for all $j > i$ and $Z_{r,\tilde{L}_r(i)} = v(i)$. This fact implies that

$$\left(Z_{r,\tilde{L}_r(i)} = v(i)\right) \ge \left(0 = Z_{r,\tilde{L}_r(j)}\right) \tag{3}$$

Now, for all $j < i$, because $v(j) \le v(i)$ and each $Z_{r,\tilde{L}_r(j)}$ can only take on the values $v(j) \ge 0$ or zero by definition, we have

$$\left(Z_{r,\tilde{L}_r(i)} = v(i)\right) \ge v(j) \ge Z_{r,\tilde{L}_r(j)} \tag{4}$$

Combining Eqs. (3) and (4) means that $v(i) = Z_{r,\tilde{L}_r(i)} \ge Z_{r,\tilde{L}_r(j)}$ for all $i \ne j$; so, conditioned on the event of C_i having probability $P(C_i)$, we have

$$V_r = \max_{l \in L_r} \{Z_{rl}\} = Z_{r,\tilde{L}_r(i)} = v(i) \tag{5}$$

Suppose that r is completed by at least one planner $\tilde{L}_r(k)$ for some $k \in \{1, \ldots, |\tilde{L}_r|\}$ so that the set of all indexed planners in $\tilde{L}_r$ that completed the request r is finite and nonempty. This condition implies the existence of a maximum index $k \in \{1, \ldots, |\tilde{L}_r|\}$, with $i \ge k$, such that r is completed by planner $\tilde{L}_r(i)$ but not by any $\tilde{L}_r(j)$ satisfying $i < j$, which corresponds exactly to the event C_i. Define C_0 to be the event that the request r is not completed by any planner in $\tilde{L}_r$. Then,

$$\bigcup_{i=0}^{|\tilde{L}_r|} C_i$$

covers the entire sample space of possible completions for a request r on planners in L_r. Furthermore, each pair $C_i, C_j \in \{C_0, C_1, \ldots, C_{|\tilde{L}_r|}\}$, with $i \ne j$, must be pairwise disjoint. To see this, assume (without loss of generality) that $j > i$. Then, $j \ge 1$, so $C_j \ne C_0$, meaning that the event C_j requires r to be completed by $\tilde{L}_r(j)$. However, C_i necessarily does not have r completed on $\tilde{L}_r(j)$ because $i < j$. Therefore, C_i and C_j can never happen simultaneously, so they must be pairwise disjoint. This result, in combination with the fact that

$$\bigcup_{i=0}^{|\tilde{L}_r|} C_i$$

covers the entire sample space of possible completions, implies that $\{C_0, C_1, \ldots, C_{|\tilde{L}_r|}\}$ forms a partition of the sample space. Therefore, using Eq. (5),

$$E\{V_r\} = \sum_{i=0}^{|\tilde{L}_r|} E\{V_r | C_i\} P(C_i) = \sum_{i=0}^{|\tilde{L}_r|} v(i) P(C_i) \tag{6}$$

Recall that $q(r, l, x_{rl})$ is defined as the probability that zero coordination requests associated with request r will be completed by planner l during any execution phase, as a function of the decision x_{rl}. Assume (for simplicity) that each planner completes a request r independently of every other planner. Although this may not be completely accurate (for example, in cases where cloud cover prevents two separate planners from completing the same request), it does provide a simple method for obtaining a realistic, quantifiable expression for each $P(C_i)$. Then, we have the following:

$$\begin{aligned} P(C_i) &= P\Big(r \text{ is completed by } \tilde{L}_r(i) \text{ during some execution period}\Big) \prod_{j=i+1}^{|\tilde{L}_r|} P\Big(r \text{ is never completed by } \tilde{L}_r(j)\Big) \\ &= \Big[1 - q(r, \tilde{L}_r\Big(i), x_{r,\tilde{L}_r(i)}\Big)\Big] \left[\prod_{j=i+1}^{|\tilde{L}_r|} q\Big(r, \tilde{L}_r(j), x_{r,\tilde{L}_r(j)}\Big) \right] \end{aligned} \tag{7}$$

Combining Eqs. (2), (6), and (7), $P(C_i)$ and, therefore, $E\{V_r\}$ both become nonlinear in the decision variables x_{rl}; see Algorithm 1 for a combination of Eqs. (2), (6), and (7) into a practical method for computing $E\{V_r\}$. Therefore, the original objective for this problem, to maximize the expected total utility

$$E\left\{\sum_{r\in R} V_r\right\}$$

at each planning iteration, is also nonlinear because

$$E\left\{\sum_{r\in R} V_r\right\} = \sum_{r\in R} E\{V_r\} \tag{8}$$

We can eliminate the nonlinearity by introducing binary decision variables $y_{rG} \in \{0, 1\}$ into the formulation, which represent composite decisions. Specifically, each decision variable y_{rG} takes a value of one if we send the request r to each planner $l \in G$, for some set $G \subseteq L_r$, but not to any other planners; otherwise, we set $y_{rG} = 0$. We introduce one such variable for each potential (r, G) pair such that $r \in R$ and $G \in T_r$, where T_r is the set of all subsets of L_r with, at most, $N^{\max}$ elements, including the empty set, for some given limit $N^{\max} > 0$ of coordination requests that can be sent per user request. For each of these composite variables, we introduce a composite utility k_{rG}, which is defined to be the expected utility of request r if a coordination request associated with r is sent to each of the planners in G, thereby choosing each x_{rl} according to Eq. (10). Then,

$$k_{rG} = E\{V_r; x_{rl} \quad \text{for all } l \in L_r\} \tag{9}$$

where $E\{V_r\}$ is calculated using Eq. (6) or Algorithm 1 for request r being sent to all planners $l \in G$, and the decision variables x_{rl} for all $l \in L_r$ take the values

$$x_{rl} = \begin{cases} 1 & \text{if } l \in G \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

We must have $y_{rG} = 1$ for exactly one set $G \in T_r$ so that we do not count the value of more than one set G for any given request r, which is equivalent to including the mathematical constraints

Algorithm 1 Expected utility of a request

Inputs: request r, decision variables x_{rl} for all $l \in L_r$

1: Sort the set L_r into a new indexed set $\tilde{L}_r$ ordered according to increasing values v_{rl} for all $l \in L_r$. In other words, if $\tilde{L}_r(i)$ denotes the ith planner of this set and $v(i)$ the value of the associated request/planner pairing $(r, \tilde{L}_r(i))$, then $v(i) \leq v(j)$ for all $i \leq j$.

2: Initialize $v_{\text{total}} = 0$ and $i = 1$.

3: While $i \leq |\tilde{L}_r|$, loop through the following:

- a: Define $l = \tilde{L}_r(i)$, which is the ith element of $\tilde{L}_r$.
- b: Define $p = q(r, l, x_{rl})$ from Eq. (2), the probability that no coordination requests associated with request r will be completed by planner l during any execution phase, as a function of the decision x_{rl}.
- c: Update $(p)(v_{\text{total}}) + (1 - p)v(i) \rightarrow v_{\text{total}}$.
- d: Update $i + 1 \rightarrow i$.

4: Output $E\{V_r\} = v_{\text{total}}$.

$$\sum_{G \in T_r} y_{rG} = 1 \quad \forall\, r \in R \tag{11}$$

in our integer programming formulation. We also know that, if some $y_{rG} = 1$, then it must be true that $x_{rl} = 1$ for all $l \in G$, so we must include the constraints

$$y_{rG} \le x_{rl} \qquad \forall\, r \in R,\;\; G \in T_r,\;\; l \in G \tag{12}$$

Because we are performing a maximization of the objective in Eq. (8), the constraints [Eqs. (11) and (12)] are also sufficient to ensure that, if all estimated probabilities are in the open interval (0,1) with $x_{rl} = 1$ for all $l \in G$ and $x_{rl} = 0$ for all $l \notin G$, then $y_{rG} = 1$. This is because $k_{rG} = E\{V_r\}$ strictly increases if an extra planner l is added to G, so we have the relationship that, if $G_1 \subset G_2$, then $k_{rG_1} < k_{rG_2}$. Thus, in order to maximize Eqs. (8) and (9) while satisfying constraint (11), we must set $y_{rG'} = 1$ where $G' = \{l | x_{rl} = 1\}$, which is exactly the correct composite variable. (Even if some of the estimated probabilities are not in the open interval (0,1), we still obtain an optimal solution because the only possible variables y_{rG} that would be set to one in a maximization are those where (r, G) satisfies $k_{rG} = k_{rG'}$.)

Coordinated planning may also involve budget and capacity requirements. Letting c_l be the fee charged by planner l per submitted request and b the money available to the CP per iteration, the budget constraint from Sec. II is

$$\sum_{r \in R} \sum_{l \in L_r} c_l x_{rl} \le b \tag{13}$$

Similarly, letting n_l be the maximum number of coordination requests that can be sent to an arbitrary planner l in an iteration, the capacity constraints from Sec. II are

$$\sum_{r \in R_l} x_{rl} \le n_l \quad \forall\, l \in L \tag{14}$$

Combining these constraints, the goal is to choose values for the variables x_{rl} and y_{rG} that solve

$$\max \sum_{r \in R} \sum_{G \in T_r} k_{rG} y_{rG} \tag{15}$$

$$\begin{aligned}
\text{subject to} \sum_{r \in R} \sum_{l \in L_r} c_l x_{rl} &\le b \\
\sum_{r \in R_l} x_{rl} &\le n_l \quad \forall\, l \in L \\
\sum_{G \in T_r} y_{rG} &= 1 \quad \forall\, r \in R \\
y_{rG} &\le x_{rl} \quad \forall\, r \in R, \quad G \in T_r, \quad l \in G \\
x_{rl} &\in \{0, 1\} \quad \forall\, r \in R, \quad l \in L_r \\
y_{rG} &\in \{0, 1\} \quad \forall\, r \in R, \quad G \in T_r
\end{aligned}$$

Once this integer program is solved, we simply send a coordination request for the user request r to each planner l such that $x_{rl} = 1$.

The number of decision variables y_{rG} in this composite variable formulation [Eq. (15)] has the potential to grow very quickly at a rate of

$$O\left(|R||L|^{N^{\max}}\right)$$

for a fixed value of $N^{\max}$, assuming every request is feasible on every planner. This not only has the effect of vastly increasing the computational complexity of the integer program but also increases the number of operations required to instantiate the integer program at an exponential rate. For example, Algorithm 1 must be run once per k_{rG} value; thus, the number of evaluations increases at the same rate as y_{rG}. Although the tractability can be controlled to a certain extent through the parameter $N^{\max}$, development of an effective alternate approximation or heuristic approach for sufficiently large problems would be beneficial. One such heuristic related to this work can be found in [17], although evaluation of the tradeoff between runtime and quality is left to future research.

IV. Estimating Uncertainty

The integer programs of the previous section rely on the probability estimators P_s, P_a, and P_c. Determining these functions can be difficult due to nonstationary levels of request saturation and the absence of prior data to analyze. For this paper, a Bayesian logistic regression (BLR) model was used (see [18] for a background into Bayesian analysis and logistic regression). Under this model, a set of data $D = \{(\boldsymbol{x}_1, y_1), \ldots, \boldsymbol{x}_m, y_m)\}$ is observed, from which either a classifier or probability estimate must be extracted. Within D, each $y_i \in \{0, 1\}$ is the response variable for the ith observation, and each $\boldsymbol{x}_i$ is a vector of observed attributes related to the ith observation. The assumption is then made that the log-odds ratio for an arbitrary unknown observation y can be expressed as a linear combination of the attributes in $\boldsymbol{x}$ so that

$$\log\left(\frac{P(y = 1 | \boldsymbol{x}, \boldsymbol{\lambda})}{P(y = 0 | \boldsymbol{x}, \boldsymbol{\lambda})}\right) = \boldsymbol{\lambda}^T \boldsymbol{x}$$

for some vector of Bayesian parameters $\boldsymbol{\lambda}$. Rearranging this expression results in a generalized linear model with the logistic link function given in [19], implying

$$P(y=1|\boldsymbol{x},\boldsymbol{\lambda}) = \mathrm{logit}^{-1}(\boldsymbol{\lambda}^T\boldsymbol{x}) = \frac{e^{\lambda^T x}}{1+e^{\lambda^T x}}$$

We assume that all observations y_i are independently conditioned on $\boldsymbol{\lambda}, \boldsymbol{x}_i$. To make this model Bayesian, the parameters $\boldsymbol{\lambda}$ are assumed to be random and, as such, are given a prior probability density $p(\boldsymbol{\lambda})$. In addition, it is assumed that $\boldsymbol{\lambda}$ is independent of all observations $\boldsymbol{x}_i$ so that, for a single observation, Bayes's rule implies

$$p(\boldsymbol{\lambda}|\{(\boldsymbol{x},y)\}) = \frac{p(\boldsymbol{\lambda},\boldsymbol{x},y)}{p(\boldsymbol{\lambda},y)} = \frac{p(y|\boldsymbol{\lambda},\boldsymbol{x})p(\boldsymbol{\lambda},\boldsymbol{x})}{p(y|\boldsymbol{x})p(\boldsymbol{x})} = \frac{p(y|\boldsymbol{\lambda},\boldsymbol{x})p(\boldsymbol{\lambda})p(\boldsymbol{x})}{p(y|\boldsymbol{x})p(\boldsymbol{x})} \propto p(y|\boldsymbol{\lambda},\boldsymbol{x})p(\boldsymbol{\lambda}) \tag{16}$$

This can be extended to all m observations in D, yielding a posterior distribution on $\boldsymbol{\lambda}$ from which $p(y|\boldsymbol{x})$ can be determined via simulation [18].

Due to the dynamic nature of coordinated planning, obsolete data need to be discarded over time, allowing the probability estimators to adapt to changes in the system. In addition, differences between the probability estimators require observations to be separated by planner according to whether they refer to a request being sent, accepted, or completed. Thus, if L is the set of planners in the system, a total of $3|L|$ groups of observations will be collected: SentData(l), AcceptData(l), and CompleteData(l), for all planners l. Observations for this are constructed as follows:

1) For a given planner l, an observation $y^{\text{sent}}_{(r,l,d)} \in \text{SentData}(l)$ where $y^{\text{sent}}_{(r,l,d)} = 1$ indicates that a coordination request was sent associated with the request/planner-execution phase triple (r, l, d); otherwise, $y^{\text{sent}}_{(r,l,d)} = 0$.

2) An observation $y^{\text{accept}}_{(r,l,d)} \in \text{AcceptData}(l)$ takes the value $y^{\text{accept}}_{(r,l,d)} = 1$ if the coordination request was accepted for the triple (r, l, d); otherwise, $y^{\text{accept}}_{(r,l,d)} = 0$ if the coordination request was sent but not accepted.

3) An observation $y^{\text{complete}}_{(r,l,d)} \in \text{CompleteData}(l)$ takes the value $y^{\text{complete}}_{(r,l,d)} = 1$ if a coordination request was completed for the triple (r, l, d); otherwise, $y^{\text{complete}}_{(r,l,d)} = 0$ if the coordination request was sent and accepted but not completed.

It is important to note that, for the data in each of the AcceptData(l) sets, only coordination requests that have already been sent are considered; for CompleteData(l), only coordination requests that have already been sent and accepted are considered. In addition to these, the attribute sets SentAttributes(l), AcceptAttributes(l), and CompleteAttributes(l) are also recorded, where any attribute vector $\boldsymbol{x}_{(r,l,d)}$ in one of these sets is associated with the response $y_{(r,l,d)}$ in the corresponding dataset, i.e., SentAttributes(l) corresponds with SentData(l), AcceptAttributes(l) with AcceptData(l), and CompleteAttributes(l) with CompleteData(l). The observations are recorded in the appropriate sets as they are received over time.

To allow the user to incorporate complicated prior beliefs, the following method is used to construct the Bayesian prior distribution as a form of regularization/protection against the high variability inherent in estimates based on minimal data. These beliefs are constructed using quantitative belief statements of the following form:

We estimate that the probability $P(y=1|\boldsymbol{x}^k)$ given some attributes $\boldsymbol{x}^k$ is f_k, but we have c_k confidence that it is within the interval $[a_k, b_k]$.

In these statements, $0 < a_k < f_k < b_k < 1$ represent estimates about the conditional probability $P(y=1|\boldsymbol{x}^k)$, and $c_k \in (0,1)$ is some fractional level of confidence in the statement. Suppose that we have n such confidence statements, so $k = 1, \ldots, n$. We will use the information from these statements to produce a prior distribution on $\boldsymbol{\lambda}$. We assume a multivariate normal distribution with independent component random variables for this prior on $\boldsymbol{\lambda}$ due to its unimodal structure, convenient parametrization in terms of a mean vector and covariance matrix, and property that linear combinations of its component random variables are still normal. Although this may not induce sparsity in the results as explained in [20], and the component random variables may not quite form an independent set, the nice structure of the Gaussian allows an analytical method to be used for injection of outside beliefs that may not be afforded by using a sparsity prior, or by modeling dependencies between the component random variables. To build this prior distribution, we will first assume that the jth component of $\boldsymbol{\lambda}$ has a univariate normal distribution with mean μ_j and variance v_j for all $j = 1, \ldots, d$, where d is the length of $\boldsymbol{\lambda}$. Thus, $\boldsymbol{\lambda}$ must have a multivariate normal distribution with mean vector $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_d)$ and covariance matrix $\mathrm{diag}(\boldsymbol{v})$, where $\boldsymbol{v} = (v_1, \ldots, v_d)$. Once we have constructed $\boldsymbol{\mu}$ and $\boldsymbol{v}$, we will have a completely well-defined model for Bayesian inference.

To figure out good values for $\boldsymbol{\mu}$ and $\boldsymbol{v}$, let us examine the form of our beliefs. We can interpret the values f_k to be estimates for the percent of observations with attribute vector $\boldsymbol{x}^k$ that would be expected to have the response $y = 1$ rather than $y = 0$. Thus, it makes sense to construct the mode $\boldsymbol{\mu}$ of our prior distribution on $\boldsymbol{\lambda}$ to mimic traditional maximum-likelihood estimation, discussed in [21], by maximizing the "pseudolikelihood" function $L(\boldsymbol{\lambda})$ where

$$L(\boldsymbol{\lambda}) = \prod_{k=1}^{n}\left(\frac{e^{\lambda^T x^k}}{1+e^{\lambda^T x^k}}\right)^{f_k}\left(\frac{1}{1+e^{\lambda^T x^k}}\right)^{(1-f_k)} \tag{17}$$

We therefore set the values of $\boldsymbol{\mu}$ using the convex optimization

$$\begin{aligned}\boldsymbol{\mu} = \boldsymbol{\lambda}^* &= \operatorname*{argmax}_{\lambda} \quad L(\boldsymbol{\lambda}) = \operatorname*{argmin}_{\lambda} -\log(L(\boldsymbol{\lambda})) = \operatorname*{argmin}_{\lambda} -\sum_{k=1}^{n}\left[f_k \log\left(\frac{e^{\lambda^T x^k}}{1+e^{\lambda^T x^k}}\right) + (1-f_k)\log\left(\frac{1}{1+e^{\lambda^T x^k}}\right)\right] \\ &= \operatorname*{argmin}_{\lambda} \sum_{k=1}^{n}\left[f_k \log\left(1+e^{-\lambda^T x^k}\right) + (1-f_k)\log\left(1+e^{\lambda^T x^k}\right)\right]\end{aligned} \tag{18}$$

We now turn our attention to finding the values of the variance vector $\boldsymbol{v}$ using the confidence interval $[a_k, b_k]$ and the scalar c_k. Define θ^k in terms of the random vector $\boldsymbol{\lambda}$ such that

$$\theta^k = \mathrm{logit}^{-1}(\boldsymbol{\lambda}^T\boldsymbol{x}^k) = \frac{e^{\lambda^T x^k}}{1+e^{\lambda^T x^k}}$$

where $\mathrm{logit}(p) = \log(p/1-p)$ is the logit link function (see [18] for details). Thus, θ^k represents the estimate of $P(y=1|\boldsymbol{x}^k)$ given by the kth belief statement. We assume that each confidence interval is "centered" in the interval [0,1] in the sense that, for any given attribute vector $\boldsymbol{x}^k$, we have

$$P\Big(\theta^k \le a_k|\boldsymbol{x}^k\Big) = P\Big(\theta^k \ge b_k|\boldsymbol{x}^k\Big) \tag{19}$$

where

$$P\Big(\theta^k \in [a_k, b_k] \,\Big|\, \boldsymbol{x}^k\Big) = c_k$$

Combining this assumption with the confidence statement, we see that

$$P\Big(\theta^k \le a_k|\boldsymbol{x}^k\Big) = P\Big(\theta^k \ge b_k|\boldsymbol{x}^k\Big) = \frac{1-c_k}{2}$$

Manipulating this expression, we have

$$\frac{1-c_k}{2} = P\Big(\theta^k \le a_k|\boldsymbol{x}^k\Big) = P\Big(\mathrm{logit}(\theta^k) \le \mathrm{logit}(a_k)|\boldsymbol{x}^k\Big) = P\Big(\boldsymbol{\lambda}^T\boldsymbol{x}^k \le \mathrm{logit}(a_k)|\boldsymbol{x}^k\Big) = \phi\left(\frac{\mathrm{logit}(a_k) - \boldsymbol{\mu}^T\boldsymbol{x}^k}{\sqrt{(\boldsymbol{x}^k)^T\mathrm{diag}(\boldsymbol{v})\boldsymbol{x}^k}}\right) \tag{20}$$

where $\phi(\cdot)$ denotes the standard normal cumulative distribution function. In this expression, the second equality holds by monotonicity of the logit $(\cdot)$ function, and the third equality holds by definition of θ^k. The final equality holds because each of the λ_j are normal random variables with mean μ_j and variance v_j, independent of other λ_i or $\boldsymbol{x}^k$, implying that $\boldsymbol{\lambda}^T\boldsymbol{x}^k$ is normal with mean $\boldsymbol{\mu}^T\boldsymbol{x}^k$ and standard deviation $\sqrt{(\boldsymbol{x}^k)^T\mathrm{diag}(\boldsymbol{v})\boldsymbol{x}^k}$. By similar reasoning, we have

$$1 - \frac{1-c_k}{2} = \frac{1+c_k}{2} = P\Big(\theta^k \le b_k|\boldsymbol{x}^k\Big) = \phi\left(\frac{\mathrm{logit}(b_k) - \boldsymbol{\mu}^T\boldsymbol{x}^k}{\sqrt{(\boldsymbol{x}^k)^T\mathrm{diag}(\boldsymbol{v})\boldsymbol{x}^k}}\right) \tag{21}$$

Unfortunately, it is very possible that some of our belief statements are unintentionally in direct conflict with each other or with some of the modeling assumptions used to derive Eqs. (20) and (21), implying that existence of a solution $\boldsymbol{v}$ to the system of Eqs. (20) and (21) for all $k = 1, \ldots, n$ is not guaranteed. Thus, our objective in selecting $\boldsymbol{v}$ will be to choose parameters that either solve the system of Eqs. (20) and (21) for all $k = 1, \ldots, n$ if our belief statements are appropriate or are "close" in some sense to solving the system if no solution exists. To accomplish this goal, we note that having already selected the mode $\boldsymbol{\mu}$ of our prior distribution, conflicting belief statements make us less confident that the mode should actually be located at $\boldsymbol{\mu}$. This in turn implies that the confidence levels c_k given in the belief statements should actually be interpreted as upper bounds on our confidence about the location of the mode $\boldsymbol{\mu}$. In terms of probability statements, we interpret this bound to mean

$$P\Big(\theta^k \in [a_k, b_k] \,|\, \boldsymbol{x}^k\Big) \le c_k \tag{22}$$

We choose to enforce this bound by constraining feasible values of $\boldsymbol{v}$ to satisfy

$$\begin{aligned} P\Big(\theta^k \le a_k|\boldsymbol{x}^k\Big) &= \phi\left(\frac{\mathrm{logit}(a_k) - \boldsymbol{\mu}^T\boldsymbol{x}^k}{\sqrt{(\boldsymbol{x}^k)^T\mathrm{diag}(\boldsymbol{v})\boldsymbol{x}^k}}\right) \ge \frac{1-c_k}{2} \\ P\Big(\theta^k \ge b_k|\boldsymbol{x}^k\Big) &= 1 - \phi\left(\frac{\mathrm{logit}(b_k) - \boldsymbol{\mu}^T\boldsymbol{x}^k}{\sqrt{(\boldsymbol{x}^k)^T\mathrm{diag}(\boldsymbol{v})\boldsymbol{x}^k}}\right) \ge \frac{1-c_k}{2} \end{aligned} \tag{23}$$

We see that the first line of these constraints involving a_k will be automatically satisfied for any $a_k, \boldsymbol{\mu}$ such that

$$\mathrm{logit}(a_k) - \boldsymbol{\mu}^T\boldsymbol{x}^k \ge 0$$

This property is a direct result of the symmetry of the normal distribution about its mean $\boldsymbol{\mu}^T\boldsymbol{x}^k$ combined with the fact that $c_k \ge 0$. Similarly, the second line of constraints will be automatically satisfied for any $b_k, \boldsymbol{\mu}$ such that

$$\mathrm{logit}(b_k) - \boldsymbol{\mu}^T\boldsymbol{x}^k \le 0$$

Ignoring constraints that are automatically satisfied for all $\boldsymbol{v} > 0$ and using the properties

$$\phi^{-1}\left(\frac{1-c_k}{2}\right) < 0$$

and

$$\phi^{-1}\left(\frac{1+c_k}{2}\right) > 0$$

we can rewrite Eq. (23) in the equivalent linear form

$$(\boldsymbol{x}^k)^T \operatorname{diag}(\boldsymbol{v})\boldsymbol{x}^k \geq \left(\frac{\operatorname{logit}(a_k) - \boldsymbol{\mu}^T \boldsymbol{x}^k}{\phi^{-1}(1 - c_k/2)}\right)^2 \quad \forall\ k{:}\operatorname{logit}(a_k) - \boldsymbol{\mu}^T \boldsymbol{x}^k < 0$$

$$(\boldsymbol{x}^k)^T \operatorname{diag}(\boldsymbol{v})\boldsymbol{x}^k \geq \left(\frac{\operatorname{logit}(b_k) - \boldsymbol{\mu}^T \boldsymbol{x}^k}{\phi^{-1}(1 + c_k/2)}\right)^2 \quad \forall\ k{:}\operatorname{logit}(b_k) - \boldsymbol{\mu}^T \boldsymbol{x}^k > 0 \tag{24}$$

Thus, a vector $\boldsymbol{v}$ satisfies Eq. (24) if, and only if, it also satisfies Eq. (23). Also, any vector $\boldsymbol{v}$ satisfying Eq. (23) has the property that

$$P\Big(\theta^k \in [a_k, b_k] \,|\, \boldsymbol{x}^k\Big) = 1 - P\Big(\theta^k \notin [a_k, b_k] \,|\boldsymbol{x}^k\Big) = 1 - P\Big(\theta^k \leq a_k|\boldsymbol{x}^k\Big) - P\Big(\theta^k \geq b_k|\boldsymbol{x}^k\Big) \leq 1 - \frac{1 - c_k}{2} - \frac{1 - c_k}{2} = c_k$$

meaning it also satisfies Eq. (22). By limiting our solution space with the constraints [Eq. (23)], we retain a notion of confidence interval centrality similar to that given by assumption (19).

We also want to reduce our set of potential selections for $\boldsymbol{v}$ to be those in which individual components v_i each exhibit a similar amount of uncertainty relative to the respective Bayesian parameters λ_i that they describe. We do this to ensure that we are never overly confident in one parameter, which helps to prevent overfitting of our prior distribution to possibly incorrect beliefs. We model this restriction on $\boldsymbol{v}$ by imposing the constraints

$$\frac{v_i}{\gamma_i^2} \leq r^2 \frac{v_j}{\gamma_j^2}, \qquad \forall\ i, j \in \{1, \ldots, d\}{:}i \neq j \tag{25}$$

where γ_i represents the mean value of the ith component of the vectors $\boldsymbol{x}^k$, and $r \geq 1$ is a constant allowing control over how much spread we are allowing between the elements of $\boldsymbol{v}$ (a value for r that is closer to one represents less spread). We divide each v_i by γ_i^2 in order to correct between different scales on the units being used for each attribute of $\boldsymbol{x}^k$. As a final method for protecting against poorly written confidence statements, we impose the vector constraint

$$0 \leq v_{\min} \leq v_i \leq v_{\max} \quad \forall\ i \in \{1, \ldots, d\} \tag{26}$$

on the set of feasible values for $\boldsymbol{v}$, where $v_{\min}$ and $v_{\max}$ are constants giving a region of satisfactory values for $\boldsymbol{v}$. Once we have identified all feasible $\boldsymbol{v}$ satisfying our bounds in Eqs. (23) through (26), our goal is to search this feasible set to find the one giving the "most" information pertaining to our beliefs. To complete this objective, we note that

$$\operatorname{var}\Big(\operatorname{logit}(\theta^k) \,|\, \boldsymbol{x}^k\Big) = (\boldsymbol{x}^k)^T \operatorname{diag}(\boldsymbol{v})\boldsymbol{x}^k \tag{27}$$

Qualitatively, smaller magnitudes of $\operatorname{var}(\operatorname{logit}(\theta^k) \,|\, \boldsymbol{x}^k)$ for a particular k indicate that we have more information pertaining to requests with parameters $\boldsymbol{x}^k$. Different magnitudes of the vector $\boldsymbol{x}^k$ can inflate or deflate the preceding variance, so we divide Eq. (27) by $\|\boldsymbol{x}^k\|_2^2$ as a form of normalization, allowing us to compare on an absolute scale a measure of the amount of variability that $\boldsymbol{v}$ creates in the kth belief statement. Because lower belief variability indicates more information, we aim to choose the vector $\boldsymbol{v}$ to minimize the sum over all k of the normalized variances given by the expression

$$\frac{1}{\|\boldsymbol{x}^k\|_2^2} (\boldsymbol{x}^k)^T \operatorname{diag}(\boldsymbol{v})\boldsymbol{x}^k \tag{28}$$

Combining Eqs. (23) through (28), we select $\boldsymbol{v}$ via the linear program (LP)

$$\min_{\boldsymbol{v}} \sum_{k=1}^{n} \frac{1}{\|\boldsymbol{x}^k\|_2^2} (\boldsymbol{x}^k)^T \operatorname{diag}(\boldsymbol{v})\boldsymbol{x}^k \tag{29}$$

$$\text{subject to } (\boldsymbol{x}^k)^T \operatorname{diag}(\boldsymbol{v})\boldsymbol{x}^k \geq \alpha_k^2 \quad \forall\ k{:}\operatorname{logit}(a_k) - \boldsymbol{\mu}^T \boldsymbol{x}^k < 0 \tag{30}$$

$$(\boldsymbol{x}^k)^T \operatorname{diag}(\boldsymbol{v})\boldsymbol{x}^k \geq \beta_k^2 \quad \forall\ k{:}\operatorname{logit}(b_k) - \boldsymbol{\mu}^T \boldsymbol{x}^k > 0 \tag{31}$$

$$\frac{v_i}{\gamma_i^2} \leq r^2 \frac{v_j}{\gamma_j^2} \quad \forall i, j \in \{1, \ldots, d\}{:}i \neq j \tag{32}$$

$$0 \leq v_{\min} \leq v_i \leq v_{\max} \quad \forall\ i \in \{1, \ldots, d\} \tag{33}$$

where the constants α_k and β_k are given by

$$\alpha_k = \frac{\operatorname{logit}(a_k) - \boldsymbol{\mu}^T \boldsymbol{x}^k}{\phi^{-1}(1 - c_k/2)} \quad \beta_k = \frac{\operatorname{logit}(b_k) - \boldsymbol{\mu}^T \boldsymbol{x}^k}{\phi^{-1}(1 + c_k/2)}$$

We note that the box constraints [Eq. (33)] form a closed, bounded set. This implies that, if the feasible set of this LP is nonempty, then it must form a bounded polytope, guaranteeing the existence of an optimal solution. By construction of this LP, as long $v_{\max}$ is sufficiently large, then this LP has a nonempty feasible set, and thus an optimal solution v^*. By combining the maximum likelihood approach for constructing $\boldsymbol{\mu}$ with this linear program for constructing $\boldsymbol{v}$, we can translate any positive number of qualitative belief statements into an informative prior distribution on the Bayesian parameters.

Implementation of the aforementioned model requires careful consideration of the attributes used to define each observation $\boldsymbol{x}_i$. For the analysis in Sec. V, the BLR models used to construct $P_s(r, l, t)$ applied attributes such as the expected value of a single pairing (r, l) ignoring all other decisions, the expected value of all accepted assignments of request r to any planner in the system at a given iteration, and the expected value of all accepted assignments of request r to a single specific planner l. For P_a and P_c, attributes included the normalized great-circle distance from the UAV to the request computed by the algorithm described in [22–24], the mean viewing angle (in radians) required for the satellite on planner l to service request r during execution phase d calculated using J2 secular theory [24], the amount of time it would take to service a request divided by the total length of a given execution phase, and the total percent of time a request was feasible during a given execution phase of a specific planner. Simple closed-form expressions for each of these three probability estimators were created for use in the simulations by constructing point estimates of $\boldsymbol{\lambda}$ from the sample mean obtained via posterior simulation (using the package MCMCpack [25] in R).

V. Analysis and Results

To test the limits of the methods developed in the previous sections, performance of the CP was evaluated against the current stovepiped system via a Java testbed simulation. The simulation was designed to determine the advantages of the CP in various potential scenarios that could be encountered, investigate the limits of the approach under various environments, and analyze the tractability of the approach. In particular, the following specific questions/hypotheses were addressed to provide a significant cross section of the most valuable discussions pertaining to the implementation of the CP methods:

1) How does the CP respond to differences in the request generation process? Specifically, are there any major differences in performance if requests are received simultaneously at one time versus continually via a stochastic process?

Hypothesis 1: The CP should provide an approximately equivalent performance regardless of the process by which requests are received.

2) The CP is designed to provide major performance enhancements in a situation with scarce resources (i.e., collection assets) as compared to requirements (i.e., requests). How scarce do the resources have to be in order to realize these potential improvements?

Hypothesis 2: The CP should exhibit the same or better performance than a stovepiped system, regardless of the availability of collection assets as compared to resources. However, the amount of performance gain should increase with increasing levels of scarcity.

3) How does the CP react to changes in request or planner attributes? Specifically, does the use of the probability estimator presented in Sec. IV effectively capture knowledge about the differences between requests (e.g., longer feasible time windows, differences in location, type of sensor required) in a manner that increases realized value?

Hypothesis 3: The performance gap between the proposed CP and current stovepiped methods should widen with increasing variability of request attributes.

A. CP Performance Analysis

Our experimental setup included a one-week planning scenario with two UAV planners and six low-orbit satellite planners, each containing exactly one asset (approximately the size of the problem considered in the intended notional scenario of [8]). Requests were generated with one of 10 target types and 10 sensor types, where each target-to-sensor pair had a unique observation quality chosen uniformly on the interval [0, 1]. The number of sensors on assets for any given planner was chosen uniformly on the set {1,2,3}, and each sensor was chosen uniformly across the set of all 10 sensors. It is important to note that this structure presupposed the existence of commonality between planners that allowed for flexibility in the tasking of requests. Although the CP should perform equal to or better than the stovepiped system, even without this commonality due to its ability to consider all requests and planners simultaneously, evaluation of this point is left for future research. The target region from which the requests were drawn included latitudes on the interval [35°, 40°] and longitudes on the interval [−110°, −100°] (representing a region in the Western United States). These parameters for the planners, as well as the various parameters for their assets (Keplerian elements for the satellites, location, endurance, speed, and sensor type for the UAVs) were selected to have random levels of coverage on this target region, with the caveat that all satellites were restricted to low Earth orbit (in order to provide more viewing opportunities per unit time), and all parameters remained fixed for the entire experiment once selected. This setup simulated the concept that a real-world system could have many target types and sensor types, with planners that had multiple sensors. The CP information-gathering phase from Fig. 2 was set to be 0.5 h in length, with a coordinated planning phase length of 120 s. All computation was completed by CPLEX for Java on an Intel Core i5 processor. Planning used $N^{\max} = 3$ so that, for the problem sizes considered, all integer programs were solved to optimality. Each individual planner had a planning period that was longer than 0.5 h, simulating a fast-paced planning operation in which it was necessary to guarantee that each planner iteration was considered in at least one coordinated planning phase.

Each request was given a single target type selected uniformly from all possibilities, as well as a priority drawn uniformly on the interval [0, 1]. The latitude and longitude of each request were drawn uniformly from the aforementioned target region. The altitude was chosen to have a 50% chance of being at ground level and a 50% chance of being above ground. Given that the altitude was above ground, it was selected uniformly on the interval [0, 4] km above ground level. For simplicity, when performing satellite feasibility calculations, we assumed that "ground level" meant 0 ft above the reference ellipsoid for Earth, which made little difference because we would only have an error on the order of a few kilometers. The minimum observation duration length for each request was selected on the interval of [0.017, 0.17] h (approximately 1 to 10 min) because this represented a realistic set of possible duration lengths. Each request/planner pairing was assigned a dimensionless observation $\text{Quality}_{rl} \in [0, 1]$ based on the relationship between the type of request being simulated and the sensors attached to the assets for each planner. Each request was randomly assigned another dimensionless quantity $\text{priority}_r \in [0, 1]$, so that

$$v_{rl} = \frac{1}{2}(\text{priority}_r + \text{observation Quality}_{rl})$$

This structure simulated one possible value function construction with equally weighted attributes associated with the request alone (priority), as well as the request/planner pairings (observation quality). See [8] for a discussion on construction of the value function for more specific purposes. Feasibility of a request was determined by first ensuring the sensors on board the asset controlled by a planner could collect the desired request type. If at least one sensor could collect the data, then physical constraints were checked. For satellites, this meant verifying a nonempty time window during a given execution period where there existed a line of sight between the request location and the satellite orbit. For UAVs, this

Table 2 Parameter values by case

Parameter type vs value	Case 1	Case 2	Case 3	Case 4	Case 5	Case 6	Case 7	Case 8	Case 9	Case 10	Case 11	Case 12
Request generation process	A priori	Poisson	A priori	Poisson	A priori	Poisson	A priori	Poisson	A priori	Poisson	A priori	Poisson
Number of requests[a]	Small	— —	Medium	— —	Large	— —	Small	— —	Medium	— —	Large	— —
Poisson parameter	— —	Small	— —	Medium	— —	Large	— —	Small	— —	Medium	— —	Large
Time windows considered	Short	Short	Short	Short	Short	Short	Various	Various	Various	Various	Various	Various

[a]Only applicable for the A priori/non-Poisson request generation process.

meant verifying that the request location could be reached at maximum speed/maximum endurance. It was assumed that each planner sent information to the CP at the end of their respective execution periods detailing which coordination requests were just completed or failed. Any requests not completed by the end of the scenario or by the end of their respective time windows were considered failed, and therefore did not contribute to the total value obtained.

The execution period length of each UAV planner was 2 h, and each satellite planner was the period of one orbit. All UAV planners had capacity constraints for each send/upload phase of 15 requests, and all satellites had four request capacities to simulate more stringent data constraints on satellites. Because the intent of the hypotheses being investigated did not include investigation of the effects of reservation fees on output, it was assumed that there were no budget constraints for this problem (or equivalently, zero reservation fees for any planner).

With this baseline scenario fixed, the hypotheses presented at the beginning of the section were tested via 12 different cases by varying parameters as shown in Table 2. This table shows the conceptual intent of how each parameter was varied. Each term is defined formally as follows:

1) "A priori" implies that all requests were generated and submitted to the CP before the start of the planning scenarios.

2) "Poisson" implies that requests were generated throughout the planning scenario via a Poisson process, and it assumes that they were submitted last minute to the CP, i.e., there was no delay between the coordinated planning iteration and the beginning of the acceptable time window defined by the user.

3) "Small," "medium," and "large" are 1000, 2000, or 4000 requests for a priori, and 6, 12, and 24 requests/hour for the Poisson process (for an expected total number of requests similar in size to the a priori generation).

4) "Short" implies that time windows have lengths drawn uniformly from [0, 4] h, or until the end of the planning scenario if earlier. For a priori generation, the time window is selected to begin uniformly between 0 and 164 h after the scenario begins, ensuring all windows close before the end of the scenario.

5) "Various" implies time windows have lengths drawn uniformly from [0, 48] h, or until the end of the planning scenario if earlier. For a priori generation, the time window is selected to begin uniformly between 0 and 120 h after the scenario begins, ensuring all windows close before the end of the scenario.

Within each case, the CP was compared against a notional stovepiped planning system, as well as against a baseline "myopic" coordinated planned algorithm. Problem sizes for these systems are mostly notional at this time because users currently tend to use data archives due to an inability to directly manipulate sensing resources, as mentioned in [26], although the evaluation in this paper uses a similar level of tasking complexity to the intended notional scenario of [8]. The stovepiped explanation was motivated by the description of the "current practice" in [26], where each remote sensing asset was described as being controlled and managed by individual science missions. We model this idea by assuming that the owners of each asset generate requests that are specifically designed for that particular asset but are not submitted to any other assets (future research could include comparison against a more relaxed stovepipe system where users have the knowledge/capability to submit to every possible resource, thereby overloading each asset with extraneous inputs). Thus, each request/planner pairing inherently has high value. This was simulated by assuming that each request r was designed and submitted intelligibly by the users to the planner l^* satisfying

$$v_{rl^*} = \max_{l \in L_r}\{v_{rl}\}$$

for every feasible execution period on l^*. Capacity constraints were enforced in this environment by only considering the n_{l^*} most valuable requests on each planner l^*. The baseline myopic algorithm used an identical construct to the CP methods developed in this paper without incorporating the probability estimation techniques of Sec. IV. This was done by removing all Z_{rl} random indicator variables and replacing

$$k_{rG} = E\{V_r\} = E\left\{\max_{l \in L_r}\{Z_{rl}\}\right\}$$

with

$$k_{rG} = \max_{l \in L_r}\{v_{rl} x_{rl}\}$$

where each x_{rl} was predetermined during construction of the associated composite variable y_{rG}, as discussed in Sec. III, thereby removing the probabilistic model entirely.

B. Results

All requests considered by the CP were the same ones considered by the stovepiped system and the baseline comparison algorithm. The planning scenario was designed to include enough coordinated planning phases that the results should sufficiently converge in each test case (336 coordinated planning phases). Under this design, if each hypothesis is true, we expect to see supporting evidence as follows:

1) For Hypothesis 1, regardless of the request generation process used, or the number/rate of arrival requests, the total realized value of the proposed CP should meet or exceed that of the stovepiped methods.

2) For Hypothesis 2, regardless of the number/rate of arrival requests, the total realized value of the proposed CP should meet or exceed that of the stovepiped methods. However, holding the request and time window generation processes constant, the magnitude of the value gap should increase with larger numbers of requests or larger request arrival rates.

Table 3 Mean values per request and percent of total requests completed

Parameter type vs value	Case 1	Case 2	Case 3	Case 4	Case 5	Case 6	Case 7	Case 8	Case 9	Case 10	Case 11	Case 12
Full CP (mean)	0.39	0.29	0.30	0.25	0.17	0.17	0.43	0.42	0.29	0.30	0.17	0.18
Myopic CP (mean)	0.37	0.28	0.30	0.24	0.18	0.17	0.41	0.42	0.28	0.32	0.16	0.19
Stovepiped (mean)	0.04	0.03	0.03	0.02	0.02	0.01	0.06	0.06	0.04	0.043	0.02	0.02
Full CP (% completed)	87.1	64.4	57.7	54.7	28.7	29.7	92.8	94.4	52.3	56.6	27.7	29.0
Myopic CP (% completed)	82.8	62.9	57.2	51.4	29.0	29.2	86.8	93.2	49.1	58.1	26.3	29.1
Stovepiped (% completed)	6.1	4.6	4.6	3.2	2.8	2.8	8.9	9.8	5.0	5.7	2.6	2.69

3) For Hypothesis 3, regardless of the process by which time windows are generated, the total realized value of the proposed CP should meet or exceed that of the stovepiped methods. However, holding the request number/arrival rate and generation process constant, the magnitude of the value gap should increase with larger variability between time window lengths. This is expected because the CP probability estimator is better equipped to prioritize based on urgency than a stovepiped system due to its feature extraction and forward-looking capabilities.

Table 3 summarizes the percent of requests that were successfully completed, as well as the mean value obtained per number of requests in each scenario. The results are separated out as stovepiped for the stovepiped planning method, myopic CP for the baseline myopic algorithm, and full CP for the full coordinated planning method developed in this paper.

Within each environment, the first hypothesis was validated because the stovepiped operations produced values that were much lower than those produced by either form of coordinated planning, regardless of the conditions surrounding the test case, as shown in Table 3. The results for the second and third hypotheses were more interesting. The first parts of each hypothesis were supported by the experimental results because the CP methods outperformed the stovepiped operations in every test case. However, the predictions made by both hypotheses pertaining to the performance "gap" were incorrect. Rather than exhibiting a monotonic increase in magnitude with higher numbers of requests or greater arrival rates, the gap generally decreased without a significant pattern as the number of requests increased. This observation suggested that the CP could continue to exploit synergy between planners up to a certain level of scarcity, at which point the performance might decrease. The large test cases corroborated this concept, in which the full CP was actually slightly outperformed by the myopic CP in three of the four cases (although not by the stovepipes).

Regarding the third hypothesis, pairwise comparison of cases 1 with 7, 2 with 8, and up to 6 with 12 showed that, holding the request generation process and arrival rate constant, increased variability/length in the time windows increased the performance gap for the a priori generation but actually decreased the performance gap for the Poisson generation, especially between the full and myopic runs (with the myopic runs winning in a few situations). It was originally conjectured that the performance gap should increase because there would be higher variability between the requests, thereby allowing the CP to capitalize on distinction of predictive attributes. This result can likely be attributed to the observation that, in a situation where requests are not known until the last minute (e.g., the Poisson generation method), shorter time windows imply fewer opportunities for feasibility, which inherently favors a planning technique that can identify the most advantageous collection opportunities, such as the full CP method. However, an improvement in the request features being used for learning could potentially reverse this trend, although this study is left for future research.

In general, the results of the experiments imply that both the estimation procedure and the design of the formulations provided essential contributions to the increased performance of the stochastic methods over the myopic and stovepiped methods. However, an important piece to future research will be a statistical and expert analysis of planners to identify key predictive features. Because the test results here used only generic features that were not guaranteed to correlate directly with a probability (see Sec. IV), such an analysis has the potential for huge improvements. These improvements should be most pronounced in situations where planners are known to be very particular in the requests they prefer. As with any machine-learning-based method, though, improper application may yield poor results.

VI. Conclusions

The scenarios examined provide strong evidence that the coordinated planning methods described in this paper, including the full math programming formulation and probability estimation, provided a significant increase in the total value of serviced requests compared to stovepiped operations. In all of the results, the current stovepiped operations performed considerably worse than coordinated planning. This benefit to the planning of air and space data collection missions only had a small marginal cost to the owners of the assets, without even requiring much change in the current infrastructure. Due to the integer programming nature of the model, if the size of the problem was to be increased beyond the experiments performed here, further research should be pursued into tractability of the formulation, attributes used for probability estimation, and sensitivity to the various input parameters (e.g., request arrival distributions, scenario size, robustness of the model to probability estimation errors, and effects of independence assumptions). A tractability analysis for varying $N^{\max}$ would be particularly useful for understanding the tradeoff between runtime and opportunities to exploit synergy between planners, especially if compared to a heuristic or approximate solution for the formulation [Eq. (15)]. It also would be worthwhile to investigate potential tractability improvements that could be obtained by translating the $\{0,1\}$ binary constraints of formulation (15) into quadratic constraints, thereby creating a quadratically constrained quadratic program. Additionally, the cases considered assumed at least some willingness for planners to share their assets, as well as some overlap in the feasibility of the requests across multiple planners. The results might not be as applicable if these two assumptions do not hold; further study into the varying levels of cooperation could reveal insight into the true effect of these assumptions.

A great deal of work has been done in this paper to estimate various probabilities via online BLR. Future research into different models, such as neural networks or beta regression, may be of value. However, one of the major benefits of the BLR in this paper was the ability to construct a Gaussian prior distribution that reflected prior beliefs and confidence levels in a simple manner. The ability to do this depended directly on the special structure and assumptions involved in logistic regression models. Implementing an analogous Bayesian approach in a neural network may be much more difficult due to increased model complexity. Additionally, the learning method employed in this paper was designed to exploit common features between request/planner pairings. This idea assumed that common request attributes yielding some level of improvement in predictability must exist and be known to the CP learning method. An in-depth analysis of such features or attributes, as described in Sec. IV, could provide a great deal of improvement in the probability estimation process.

Because uncertainty is inherent in this problem, it may be useful to examine robust optimization approaches to finding a solution. A large benefit could also be obtained by development of an intelligent method for reducing the space of composite variables, which could allow the formulation to be tractable in much larger problems that may be experienced in the future. In any case, coordinated planning provides a unique opportunity to expand air and space capabilities, which is an opportunity that should be further pursued.

Acknowledgments

The authors thank the National Aeronautics and Space Administration for the support provided by the Earth Science Technology Office's Advanced Information System Technology program under grant number NNX12H81G. The views expressed in this paper are those of the authors and do not reflect the official policy or position of the U.S. Air Force, the U.S. Department of Defense, or the U.S. Government.

References

[1] Brill, J., "Suomi NPP," NASA, https://jointmission.gsfc.nasa.gov/suomi_mission_details.html, https://jointmission.gsfc.nasa.gov/atms.html [accessed 07 Dec. 2016].

[2] Graham, S., "Aqua Project Science [online database]," NASA, http://aqua.nasa.gov/ [retrieved 31 April 2013].

[3] Jenner, L., "NASA–HS3," NASA, https://espo.nasa.gov/hs3/ [retrieved 07 Dec. 2016].

[4] Marlaire, R., "NASA and US Forest Service to Tame Wildfires with UAV Capabilities," NASA Ames Research Center Release M06_35AR, Moffett Field, CA, May 2006, http://www.nasa.gov/centers/ames/news/releases/2006/06_35AR.html [retrieved 13 Jan. 2013].

[5] "Joint Doctrine for Targeting," Joint Chiefs of Staff Joint Publication 3-60, Joint Staff Pentagon, Washington, D.C., Jan. 2002.

[6] Simonis, I., and Echterhoff, J., *OGC Sensor Planning Service Implementation Standard*, 2nd ed., Open Geospatial Consortium, Wayland, MA, March 2011, Paper OGC 09-000, http://portal.opengeospatial.org/files/?artifact_id=38478 [retrieved 07 Dec. 2016].

[7] Delin, K. A., and Jackson, S. P., "Sensor Web: A New Instrument Concept," *Proceedings of SPIE Volume 4284, Functional Integration of Opto-Electro-Mechanical Devices and Systems*, The International Society for Optics and Photonics, Bellingham, WA, May 2001, pp. 1–9.
doi:10.1117/12.426856

[8] Herold, T. M., "Asynchronous, Distributed Optimization for the Coordinated Planning of Air and Space Assets," M.S. Thesis, Massachusetts Inst. of Technology, Cambridge, MA, 2008.

[9] Sakamoto, P., "UAV Mission Planning Under Uncertainty," M.S. Thesis, Massachusetts Inst. of Technology, Cambridge, MA, 2006.

[10] Negron, B. E. L., "Operational Planning for Multiple Heterogeneous Unmanned Aerial Vehicles in Three Dimensions," M.S. Thesis, Massachusetts Inst. of Technology, Cambridge, MA, 2009.

[11] Chandler, P., and Pachter, M., "Hierarchical Control for Autonomous Teams," *AIAA Guidance, Navigation, and Control Conference*, AIAA Paper 2001-4149, 2001, pp. 632–642.

[12] Li, D., and Cruz, J., "A Robust Hierarchical Approach to Multi-Stage Task Allocation Under Uncertainty," *44th IEEE Conference on Decision and Control, and 2005 European Control Conference*, IEEE, Piscataway, NJ, 2005, pp. 3375–3380.

[13] Abramson, M., Carter, D., Kolitz, S., Ricard, M., and Scheidler, P., "Real-Time Optimized Earth Observation Autonomous Planning," *NASA Earth Science Technology Conference*, The Earth Science Technology Office and the Office of Earth Science, NASA, Pasadena, CA, June 2003, Paper A5P1, https://esto.nasa.gov/2012test/conferences/estc-2002/Papers/A5P1(Abramson).pdf.

[14] Bensana, E., Verfaillie, G., Michelon-Edery, C., and Bataille, N., "Dealing with Uncertainty When Managing an Earth Observation Satellite," *Proceedings of the Fifth International Symposium on Artificial Intelligence Robotics and Automation in Space*, Vol. 440, European Space Agency Publications Division, Noordwijk, The Netherlands, 1999, pp. 205–210, http://robotics.estec.esa.int/i-SAIRAS/isairas1999/mainmenu.pdf.

[15] Spivey, M. Z., and Powell, W. B., "The Dynamic Assignment Problem," *Transportation Science*, Vol. 38, No. 4, 2004, pp. 399–419.
doi:10.1287/trsc.1030.0073

[16] Bertsimas, D., and Tsitsiklis, J. N., *Introduction to Linear Optimization*, Athena Scientific, Belmont, MA, 1997, pp. 451–472.

[17] Robinson, E. J., "Coordinated Planning of Air and Space Assets: An Optimization and Learning Based Approach," M.S. Thesis, Massachusetts Inst. of Technology, Cambridge, MA, 2013.

[18] Gelman, A., Carlin, J. B., Stern, H. S., and Rubin, D. B., *Bayesian Data Analysis*, 2nd ed., Chapman and Hall/CRC, Boca Raton, FL, 2004, pp. 88–95, 424–450.

[19] Genkin, A., Lewis, D. D., and Madigan, D., "Large-Scale Bayesian Logistic Regression for Text Categorization," *Technometrics*, Vol. 49, No. 3, 2007, pp. 291–304.
doi:10.1198/004017007000000245

[20] Seeger, M., Gerwinn, S., and Bethge, M., "Bayesian Inference for Sparse Generalized Linear Models," *Machine Learning: ECML 2007*, Springer, New York, 2007, pp. 298–309.

[21] Bertsekas, D. P., and Tsitsiklis, J. N., *Introduction to Probability*, 2nd ed., Athena Scientific, Belmont, MA, 2008, pp. 462–463.

[22] Sinnett, R., "Virtues of the Haversine," *Sky and Telescope*, Vol. 68, No. 2, Aug. 1984, p. 159.

[23] Veness, C., "GIS FAQ Q5.1: Great Circle Distance Between 2 Points," *Movable Type Scripts* [Online], http://www.movable-type.co.uk/scripts/gis-faq-5.1.html [accessed 07 Dec. 2016].

[24] Vallado, D. A., *Fundamentals of Astrodynamics and Applications*, 2nd ed., Microcosm Press, El Segundo, CA, 2001, pp. 135–264.

[25] Martin, A. D., Quinn, K. M., and Park, J. H., *Package 'MCMCpack'*, 1st ed., Comprehensive R Archive Network, Univ. of California, Berkeley, CA, April 2013, pp. 64–67, http://mcmcpack.berkeley.edu/files/MCMCpack-manual.pdf.

[26] Morris, R. A., Dungan, J. L., and Bresina, J. L., "An Information Infrastructure for Coordinating Earth Science Observations," *2nd IEEE International Conference on Space Mission Challenges for Information Technology (SMC-IT'06)*, IEEE, Piscataway, NJ, 2006, pp. 397–404.

G. P. Brat
Associate Editor