A Sampling Based Approach to Robust Planning for a Planetary Lander

Connor Basich[1,2], Joseph A. Russino[1], Steve Chien[1], and Shlomo Zilberstein[2]

***Abstract*— Planning for autonomous operation in unknown environments poses a number of technical challenges. The agent must ensure robustness to unknown phenomena, unpredictable variation in execution, and uncertain resources, all while maximizing its objective. These challenges are exacerbated in the context of space missions where uncertainty is often higher, long communication delays necessitate robust autonomous execution, and severely constrained computational resources limit the scope of planning techniques that can be used. We examine this problem in the context of a Europa Lander concept mission where an autonomous lander must collect valuable data and communicate that data back to Earth. We model the problem as a hierarchical task network, framing it as a utility maximization problem constrained by a strictly monotonically decreasing energy resource. We propose a novel deterministic planning framework that uses periodic replanning and sampling-based optimization to better handle model uncertainty and execution variation, while remaining computationally tractable. We demonstrate the efficacy of our framework through simulations of a Europa Lander concept mission in which our approach outperforms several baselines in utility maximization and robustness.**

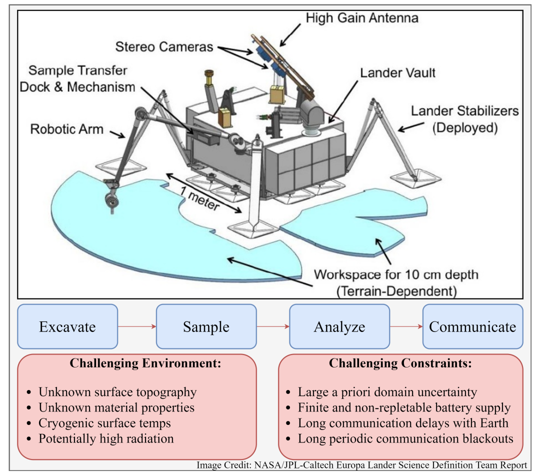

Fig. 1: An illustration of the Europa Lander concept mission in which an autonomous lander is tasked with performing *in-situ* analysis of sampled surface material and communicating the collected data to Earth.

I. Introduction

Planning in space-based domains is often challenged by large uncertainty, very low margins for error, and stringent technical constraints that render many planning techniques impractical or infeasible.

Traditional approaches to planning in space-based domains have consequently utilized deterministic or sampling-based planning methods [1], [2] which are fast and computationally inexpensive, and can be very effective when paired with *a priori* expert domain knowledge [3]. Recent work has investigated how periodic replanning, flexible execution, and online model updates can be used in conjunction with deterministic planning techniques to improve the efficacy and robustness of the plans executed by a space-based robotic system [4].

While these approaches can be effective in well-understood domains, they do not actively consider model uncertainty and off-nominal behavior during plan generation, and are hence *reactive* in how they handle the domain uncertainty. In this work, we propose a planning framework that extends the algorithm from [4] by *proactively* anticipating deviations from nominal execution by incorporating domain uncertainty into the plan generation. We examine the efficacy of our approach in the context of a proposed concept mission in which a lander is tasked with analyzing surface material to acquire valuable scientific data by performing *in-situ* analysis of samples excavated from the surface of the Jovian moon Europa, and communicating that data back to Earth [5].

This concept mission entails several challenges that differentiate it from prior missions. First, *a priori* knowledge of the environment is severely limited and uncertain. Second, the system's battery supply is finite and non-repletable (i.e. there is no possible power generation). Third, communication with Earth is constrained by two factors: (1) long communication delays when communicating with Earth due to distance mean ground-in-the-loop operations cannot be relied on (as the system will be losing battery while waiting for communications), and hence the lander must be capable of operating fully autonomously. And (2), due to Jovian occlusion the lander will be faced with long periodic communication blackouts (roughly 42 out of every 84 hours) which constrain when the lander is capable of downlinking the data it has collected to Earth. As utility is only assigned to data that is acquired and *successfully downlinked to Earth*, in order for the Lander to be successful it needs to carefully manage the trade-off between data acquisition and communication. Furthermore, it must do this while constrained by a finite and monotonically decreasing battery supply, limited knowledge of its environment, and limited communication with Earth.

We model the problem as a *hierarchical task network* (HTN) [6] due to the structured nature of the tasks that the lander can perform, and consequently use an anytime

Portions of this work were performed by the Jet Propulsion Laboratory, California Institute of Technology under a contract with the National Aeronautics and Space Administration (80NM0018D0004).

[1]Jet Propulsion Laboratory, California Institute of Technology, Pasadena, California,{joseph.a.russino, steve.a.chien}@jpl.nasa.gov

[2]University of Massachusetts Amherst, Amherst, Massachusetts, {cbasich, shlomo}@cs.umass.edu

heuristic-search algorithm designed for solving HTNs [4] as the primary subroutine of the proposed approach. Our approach is based on principles from Hindsight Optimization (HOP) [7] and works by hypothesizing a set of sampled scenarios that the system may face, planning for each scenario, and evaluating each of the sampled plans' performances across all scenarios. The plan with the highest weighted value is selected (this can be viewed as an approximation to maximizing expected utility). As the planner is deterministic, we also perform periodic replanning to ensure that the system's performance does not deviate too far from expectation. This approach has similarities with determinization-based methods that have been highly successful in solving large Markov decision processes (MDPs) [8], [9]. While MDPs and other stochastic sequential decision making models have been had success in many settings, they are computational expensive and ill-suited for domains with concurrent actions and continuous states such as what we consider here.

We present empirical results against two baseline approaches similar to those used in prior missions: a greedy planner with replanning and an anytime heuristic-search based planner with replanning. We show that the proposed planning framework, **HTNSearch-PHRA**, is more effective on average and more robust to uncertainty across five different mission scenarios. In addition, we analyze the effect of increasing the size of the hypothesized scenario set by comparing the performance of the algorithm with four different sets of hypothesized scenarios.

II. Problem Description

A. Domain Overview

The primary goal of the Europa Lander concept mission is to excavate and sample the moon's surface, analyze the sampled material for signs of biosignatures, and communicate that data back to Earth over at least thirty days [5]. Additionally, there are secondary objectives to take panoramic images of the Europan surface and collect seismographic data. Lander operations are therefore limited to primary objective tasks, secondary objective tasks, and data communication. This provides significant structure to the problem, since the concept mission clearly defines the sequence of actions required to achieve these goals.

The Europa Lander concept mission is also constrained by a finite battery that cannot be recharged. Battery life is a depletable resource, and the lander must use its energy as efficiently as possible. Each task – including sleeping – consumes energy from the battery, and the algorithm must plan accordingly to maximize utility in the face of this constraint. In addition to this challenge, the surface characteristics of Europa are uncertain, and any prior mission model that is generated before landing are assumed to be inaccurate. In particular, the energy consumption of the excavation and sample collection tasks is largely unknown. There is also significant variation in the utility of any given sample, since the value of sampling a given target on Europa depends on whether the material is scientifically interesting, e.g. if a biosignature is present.

B. Problem Formulation

Due to the presence of continuous state variables and the necessity of modeling concurrent actions, we find that stochastic planning models such as MDPs do not support our problem domain well while still being efficient to solve. Instead, as our problem has additional structure in how tasks are conditioned, we represent our model as a hierarchical task network (HTN). Hierarchical task networks have been extensively studied over the last several years as efficient models for planning in highly structured domains where expert knowledge can be embedded directly into the planner [10], [11]. In an HTN, hierarchical tasks are decomposed into a set of subtasks. We refer to the higher-level tasks as *parent tasks*, and refer to their children as *subtasks*. Parent tasks may decompose into a number of different partially ordered sets of subtasks; we refer to each of these sets as a potential *decomposition* of that parent task. Finally, we refer to tasks with no decompositions as *primitive tasks*. These primitive tasks represent tasks that the lander can be directly commanded to perform. Decompositions enable us to significantly reduce the planning search space as we can treat all subtasks of a parent task as a singular block during the planning process; for example, the model treats "excavate, sample, transfer, analyze" as a single unit and schedules the subtasks accordingly.

Formally, an input to the problem is a set $\mathcal{T} = \{T_1, ..., T_n\}$ and a system state S. Each T_i is a task that is represented by the tuple $\langle p_i, d_i, e_i, u_i, s_i, C_i, W_i, D_i \rangle$ where p_i is the priority of the task, d_i is the expected duration of the task, e_i is the expected rate of energy usage by the task, u_i is the expected utility of the task, s_i is the preferred start time of the task, C_i is the set of constraints that must be satisfied for the task to be scheduled, $W_i = \{[t_{i1}, t_{i2}], ..., [t_{in-1}, t_{in}]\}$ is the set of time windows that the task can be scheduled in, and $D_i = \{T_{i1}, ..., T_{im}\}$ is the partially ordered set of decompositions of the task, which can be empty if T_i is a primitive task.

The state, S, is represented by a set of *timeline values* that model various parameters of the system such as remaining battery supply, current data load, current time, whether the arms have been heated, etc.

There are four main parent task types in the mission model. The first is a `Preamble` which consists of post-landing initialization and other one-time initialization tasks, and must be executed immediately upon landing. Second are data acquisition tasks which consist of `excavation`, `sample collection`, `sample transfer`, and `sample analysis` tasks. Excavation can take place in one of three excavation sites, and may be skipped if a site has previously been excavated. For collection tasks, the lander may choose between one of several collection targets at any given excavation site (repeated sampling of the same target is allowed with no penalty). The analysis task returns the dataload acquired from a given sample upon completion; dataload represents the maximum potential utility that the acquired data provides upon successful downlink to Earth. Third, there are Seismograph/Panorama tasks which consist of `seismographic data collection` and `panoramic image data collection`; these tasks provide less data but are more reliable in their execution. Fourth is the `communication` task which decomposes into either a single, or a sequence of two, communication(s), either of which can be of raw data or compressed data.

In this domain, we assign utility solely to the successful completion of a communication task, where communicating raw data provides greater utility but consumes more energy than communicating compressed data; note that both communication tasks consume the same amount of dataload. Expected utilities are assigned to tasks *a priori* but we note that in practice may likely be updated online as new information is obtained by the system.

III. Approach

The underlying planning method employed in this approach relies on heuristic search. Search-based planning algorithms have been popular for a number of years as they (1) do not require that the full state space is evaluated to produce a solution [12], (2) are often anytime algorithms that can return a solution at any point during runtime [13], and (3) can easily leverage heuristics to reduce the computational burden while still achieving high performance [14].

A. HTNSearch

Our algorithmic contribution, Algorithm 1, relies on the heuristic search approach for solving HTNs described in [4] – which we henceforth refer to as `HTNSearch` – as its primary subroutine. We offer a brief description of their algorithm.

`HTNSearch` initializes a search graph on a flattened version of the input task network, where nodes hold partial plans and edges hold (flattened) task decompositions or primitive tasks. As the algorithm is deterministic, the cost and utility associated with any node is their respective sums over the tasks in the partial plan. The algorithm performs a heuristic branch and bound search procedure over the search graph where the search is bounded by both the feasibility of partial plans (their total energy cost cannot exceed the current battery supply, and the plan adheres to inter-task constraints), and optional computational constraints on the number of explorable nodes. For any plan and decomposition pair, (P, d), a density based heuristic value of $utility(P) + \frac{utility(d)}{cost(d)}$ is used and ties are broken in favor of lower cost.

The main limitation of this approach is that it is a deterministic algorithm for a non-deterministic domain. In other words, the plan that is produced assumes that the future will operate exactly according to expectation. The original approach addresses this issue primarily through the use of online model updates and frequent replanning to "course-correct" the system online. This idea is similar to that of determinization-based approaches for solving very large MDPs, such as FF-Replan [8], which have been shown to perform well in the MDP setting, particularly in the infinite horizon case, but are not robust to dead-ends. Hence, we propose a planning algorithm that *proactively* considers off-nominal behavior and execution during plan time, rather than just *reactively* responding to off-nominal behavior and execution. We demonstrate through empirical evaluations that our approach is indeed more robust to off-nominal scenarios than the standard heuristic search, performing as well or better in both positive and negative scenarios, without sacrificing performance in the nominal case.

B. HTNSearch with Post Hoc Robustness Analysis

The pseudocode for the proposed approach can been seen in Algorithm 1, and we describe it here. Algorithm 1 takes in as input a hierarchical task network, $\mathcal{T}$, and begins by producing a set of hypothesized scenarios, $\mathcal{H}$, via the function *hypothesizeScenarios* (line 3). A scenario $h \in \mathcal{H}$ is comprised of an initial state and an instantiation of the parameterized domain (e.g. the timeline impacts of each task). This function is left general as its implementation will be both domain and purpose specific. In our experiments, we hypothesize multiple scenarios where the current battery life of the lander is varied to represent the uncertainty over the "true" battery life of the lander. However, other parameters may be varied such as the time or energy to perform various tasks, the likelihood of positive data from different samples, or the possibility of excavated sites collapsing. These scenarios may be developed using expert knowledge *a priori*, or may be generated online by drawing from distributions that parameterize the domain model.

For each hypothesized scenario $h \in \mathcal{H}$, the algorithm instantiates h by updating the relevant parameters of $\mathcal{T}$ and then calls `HTNSearch` on the newly instantiated task network to produce the best-found plan $\mathcal{P}$ (lines 5-7) within the computational constraints. The plan $\mathcal{P}$ is evaluated on each scenario $h' \in \mathcal{H}$ by computing the expected utility following $\mathcal{P}$, $V^{\mathcal{P}}(\mathcal{T}, h')$, using a stochastic execution graph subroutine (line 10). The observed expected utility is weighted by a function *getScenarioWeight* and the weighted value is added to the total score of the plan, $\mu^{\mathcal{P}}$ (lines 9-10). Finally, the plan that had the highest total score is returned.

The function *getScenarioWeight* returns a real valued num-

Algorithm 1: HTNSearch-PHRA

Input: A hierarchical task network $\mathcal{T}$ and state s
Result: A plan $\mathcal{P}^*$

1 $\mathcal{P}^* \leftarrow$ None
2 $\mu^* \leftarrow -\infty$
3 $\mathcal{H} \leftarrow$ hypothesizeScenarios$(\mathcal{T}, s)$
4 **for** $h \in \mathcal{H}$ **do**
5     $\hat{\mathcal{T}} \leftarrow instantiateScenario(\mathcal{T}, h)$
6     $\mathcal{P} \leftarrow$ HTNSearch$(\hat{\mathcal{T}})$
7     $\mu^{\mathcal{P}} \leftarrow 0$
8   **for** $h' \in \mathcal{H}$ **do**
9         $\gamma \leftarrow getScenarioWeight(\mathcal{T}, h')$
10         $\mu^{\mathcal{P}} \leftarrow \mu^{\mathcal{P}} + \gamma V^{\mathcal{P}}(\mathcal{T}, h')$
11   **if** $\mu^{\mathcal{P}} > \mu^*$ **then**
12         $\mathcal{P}^* \leftarrow \mathcal{P}$
13         $\mu^* \leftarrow \mu^{\mathcal{P}}$;
14 **return** $\mathcal{P}^*$

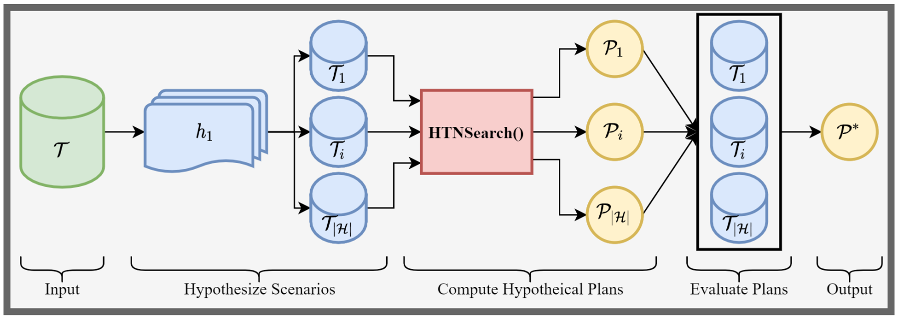

Fig. 2: An illustration of the `HTNSearch-PHRA` algorithm.

ber in $[0, 1]$, given an HTN and a scenario. In our case, we compute the Legendre-Gauss quadrature weights assuming that the parameters relevant to the hypothesized scenarios are normally distributed. In general we believe that any relative likelihood-based weighting scheme will work, however we note that offline optimization of these weights may be worthwhile particularly when the scenarios considered are over multiple different task network parameters.

We compute the expected utility of the plan $\mathcal{P}$ using a stochastic subroutine that builds the non-deterministic execution graph of $\mathcal{P}$ given the distributions which parameterize the task network (energy cost, duration, data, and utility). Computing the expected utility of a deterministic plan in a stochastic domain is significantly cheaper than computing a fully stochastic policy in the first place, and still allows us to observe a more accurate evaluation of each plan.

Finally, as the HTN search algorithm dominates the other subroutines in terms of runtime complexity, the runtime of Algorithm 1 is $\sim |\mathcal{H}|$ times the runtime of `HTNSearch` when $|\mathcal{H}|$ is small. However, as $|\mathcal{H}|$ grows, the computation spent evaluating plans grows at a rate of $O(|\mathcal{H}|^2)$. Furthermore, there are diminishing returns to increasing the number of hypothesized scenarios considered, particularly when adding very low likelihood scenarios to $\mathcal{H}$. An analysis of this can be found further in the paper. Ultimately, the problem of determining which, and how many, scenarios to include in $\mathcal{H}$ is an important element in balancing the trade off between efficiency and effectiveness.

IV. Experiments

A. Experimental Setting

To evaluate the performance of Algorithm 1, we compared against two baselines: `HTNSearch` and `GREEDY`. In `GREEDY`, priorities were assigned *a priori* to each task decomposition, and the planner greedily attempts to schedule tasks in order of priority at each planning cycle, skipping over tasks if they cannot be scheduled due to conflicts or violated constraints. Priorities are assigned offline using a combination of hard-coded domain knowledge (e.g. the Preamble must have the highest priority) and Monte carlo trials on sampled priority orderings across the input task.

We simulated five different stochastic variants of the domain: (1) Nominal, (2) Low Energy, (3) High Energy, (4) High Consumption, (5) Low Consumption. In (1) there are no off-nominal, or unexpected, events or behaviors that occur during simulation. In (2) and (3) there is a sudden change ($\pm 20\%$) in remaining battery life that occurs 500 time units into the simulation as the battery recalibrates. In (4) and (5), energy is stochastically drained or added to the observed remaining battery supply at every state update. In **all** cases, task impacts such as energy rates, duration, and data load are drawn from low variance Gaussians centered around the nominal mean *at runtime*.

In our experiments, we specifically focused on off-nominal variations on remaining battery for three reasons. The first is that, historically, battery measurements have come with large uncertainty. The second is that battery life is the most valuable resource in this domain, as time only matters in that there is a constant minimum Hotel load, and zero remaining battery supply is a terminal absorbing state. The third reason is that deviations in battery supply or energy consumption act as direct proxies for most other off-nominal behavior (extra time to complete a task, getting stuck, failing to perform a task requiring it to be repeated, etc.). However, the algorithm presented is not relegated to such a constraint, and in general can capture arbitrary scenarios.

B. Experimental Results

The main results of our experiments can be seen in Figures 3 and 4. The first experiment compares the performance of three algorithms: `GREEDY`, `HTNSearch`, and `HTNSearch-PHRA(3)`. The second experiment compares the performance of four variations of `HTNSearch-PHRA`, where the size of the hypothesized scenario set is increased.

1) Cross-Method Comparison: Figure 3 shows the mean and standard error of the utility achieved by each planning algorithm across the five domain variants. Unsurprisingly, both `HTNSearch` and `HTNSearch-PHRA` outperform `GREEDY`, particularly in the LE and HC variants which demonstrate the brittleness of a greedy approach which has no proper recourse for handling unexpected negative situations. `HTNSearch-PHRA`, which proactively selects plans that are robust to low-energy hypothesized scenarios, is more robust to the "pessimistic" variants, producing more utility on average in both the Low Energy and High Consumption scenarios. The difference is more significant in the Low Energy variant as `HTNSearch-PHRA` can proactively plan for having less energy, and does not just react to the latest

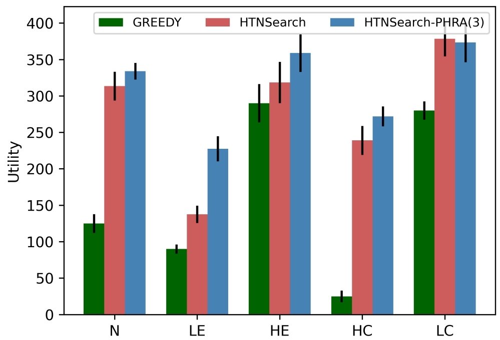

Fig. 3: Utility achieved by each planning algorithm on all five variants. Values are mean and standard error over 10 trials.

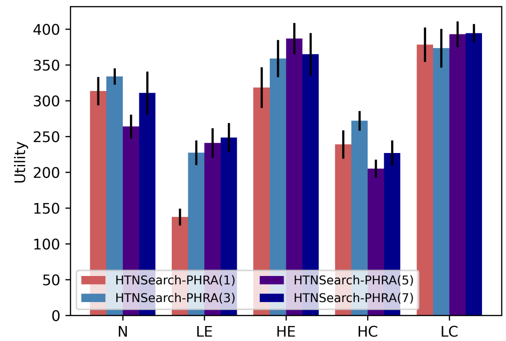

Fig. 4: Utility achieved by Alg. 1 with $|\mathcal{H}| = 1, 3, 5$, and 7 on all five variants. Values are mean and standard error over 10 trials.

observations and state updates (as `HTNSeach` does). However, the performance is still greater in the High Consumption variant because it can constantly respond to the small off-nominal deviations in behavior, and though it selects plans that are robust to high energy scenarios, it does not know that such a situation will occur until it has. In the optimistic variants, where energy is more abundant than expected, both `HTNSearch` and `HTNSearch-PHRA` perform comparably as they are both able to make use of the excess battery supply, having achieved higher utility than in the Nominal case. As above, however, our algorithm performs better in the High Energy variant as it proactively creates plans robust to similar situations. Finally, it is worth emphasizing that `HTNSearch-PHRA` also performed comparably – up to noise – to `HTNSearch` in the Nominal domain where the base task network that is used by `HTNSearch` *is correct* (up to stochasticity). This means that our approach, although sensitive to off-nominal situations, does not sacrifice performance quality in the nominal case as well.

Overall, these results demonstrate that the proposed algorithm performs comparably or better to each baseline approach in all domain variants considered, but the benefits are most notable in cases where the deviations are large and sudden, rather that small and frequent, where reactive "course-correcting" is less effective. The difference is more visible in the "pessimistic" variants because the *reactive* approach can always benefit from extra battery supply as the excess is observed, but can not always bounce back from energy deficits. However, proactively producing plans that are sensitive to both these scenarios ensures that the system follows a plan that never performs too poorly in any hypothesized scenario, while also retaining the benefits of reactively course-correcting.

2) Analysis of $\mathcal{H}$ on the Quality of `HTNSearch-PHRA`: Figure 4 shows the mean and standard error of the utility achieve by Algorithm 1 with four different hypothesis set sizes on each domain variant. We observe that increasing the size of the hypothesized scenario set generally improves performance but faces diminishing returns with the number of additional scenarios. We suggest that the reason for this is that hypothetical scenarios with low likelihood impact the score of each generated plan to a sufficiently low degree that plans generated for low likelihood scenarios are never actually selected to be scheduled. This issue may be compounded by the fact that each hypothesized scenario alters the same parameter, namely battery supply, in our experiment.

The reason that `HTNSearch-PHRA(5)` and `HTNSearch-PHRA(7)` perform better than `HTNSearch-PHRA(3)` in the energy based variants compared to the consumption based variants is likely that the ability to constantly course correct in response to small observed deviations dominates the effect of considering low-likelihood scenarios during the planning phase. However, if we consider variants (N) and (HC), `HTNSearch-PHRA(5)` and `HTNSearch-PHRA(7)` actually perform worse than the other two. The reason for this is that the plans selected are too conservative – on account of being scored on low likelihood negative outcomes – and end up costing utility over the course of the full problem horizon.

In the future, we plan to further analyze the effect of a more diverse portfolio of hypothesized scenarios across more domain parameters, and whether there is a minimum scenario likelihood needed to lead to a meaningful impact. As the runtime of the algorithm scales with the size of $\mathcal{H}$, ensuring that $|\mathcal{H}|$ is small while still covering a sufficient set of off-nominal scenarios is important for an effective mission.

V. Related Work

Onboard planning and execution are of great interest to the space domain. The *Remote Agent* was an architecture for onboard planning and execution addressing remote autonomous operation with deadlines, resource constraints, and concurrent activities [15]. They used a batch planner with a refinement search paradigm [16] to construct a temporally flexible plan but did not consider utility in plan generation and did not perform continuous replanning due to the computational expense and long planning time (indeed the replans were scheduled in the prior plan). The Earth Observing One (EO-1) spacecraft [17] was designed specifically to react to dynamic scientific events. Planning was performed by the CASPER planning software [18], which did not produce temporally flexible plans, along with the onboard executive SCL to flexibly interpret the *execution* of a plan to handle minor execution runtime variations. The flight and ground planners [19] both used a domain specific search algorithm that enforced a strict priority model over observations for limited model of utility. This scenario is similar to that proposed in this paper, in which the lander must react to dynamic events and observations in order to maximize its utility, while still adhering to both mission and spacecraft constraints. Recently, the Intelligent Payload Experiment (IPEX) also successfully used the CASPER planning software to achieve its mission objective, further validating the efficacy of using onboard replanning to handle dynamic events and observations during operation even when the plans are not temporally flexible [20].

The M2020 Perseverance rover also plans to fly an onboard planner [21] to recover productivity lost from following fixed time plans [22]. The M2020 planning architecture relies on rescheduling and flexible execution [23], ground-based compilation [1], heuristics [2], and very limited handling of planning contingencies [24]. However, many characteristics of the M2020 mission are fundamentally different from the concept mission we consider here, such as the lack of reliable *a priori* model parameters, the non-repletable battery, and the long communications blackout time windows incentivizing greater mission autonomy.

While the motivations of our approach are similar to the area of robust optimization, our work differs from prior work [25], [26] in two key aspects. First, robust optimization is a method for avoiding solutions to convex optimization problems that end up being infeasible in practice due to the realizations of uncertain parameters. However, as our problem is an indefinite planning problem, formulating it as a convex optimization problem is not straightforward. Second, uncertainty sets are often built from sampled data in the absence of well-defined priors; however, in our case, we assume the existence of well-defined priors over the uncertain parameters (e.g. battery life). We do not use these distributions during planning as full stochastic planning is intractable given the computational resources of the lander.

VI. Conclusion

Planning and scheduling tasks in the presence of large *a priori* uncertainty is a challenging problem for space-based missions. The plans need to be robust and effective without risking compromising system safety or mission success even in the face of domain uncertainty and severe computational constraints. These issues are exacerbated in the context of the Europa Lander concept mission where there is a monotonically decreasing battery supply and large windows of communication blackouts. In this work, we present a planning algorithm – `HTNSearch-PHRA` – that functions by running an efficient HTN planner on a set of hypothesized off-nominal scenarios and selecting the plan with the largest weighted expected utility across all scenarios to proactively account off-nominal execution. We validate the approach empirically on a simulated Europa Lander domain where we compared it against existing baselines across five different stochastic mission scenarios. We demonstrate that the approach is more robust to off-nominal deviations and unexpected scenarios than the existing baselines, having consistently better performance while still being computationally efficient (running on the order of a few seconds).

References

[1] W. Chi, J. Agrawal, S. Chien, E. Fosse, and U. Guduri, "Optimizing Parameters for Uncertain Execution and Rescheduling Robustness," in *ICAPS*, July 2019.

[2] W. Chi, S. Chien, and J. Agrawal, "Scheduling with Complex Consumptive Resources for a Planetary Rover," in *ICAPS*, Oct. 2020.

[3] M. Ai-Chang, J. Bresina, L. Charest, A. Chase, J.-J. Hsu, A. Jonsson, B. Kanefsky, P. Morris, K. Rajan, J. Yglesias, *et al.*, "Mapgen: mixed-initiative planning and scheduling for the mars exploration rover mission," *IEEE Intelligent Systems*, 2004.

[4] D. Wang, J. A. Russino, C. Basich, and S. Chien, "Using flexible execution, replanning, and model parameter updates to address environmental uncertainty for a planetary lander," in *I-SAIRAS*, 2020.

[5] K. P. Hand, *Report of the Europa Lander science definition team*. National Aeronautics and Space Administration, 2017.

[6] D. S. Nau, T. C. Au, O. Ilghami, U. Kuter, J. W. Murdock, D. Wu, and F. Yaman, "SHOP2: An HTN Planning System," *JAIR*, Dec. 2003.

[7] E. K. Chong, R. L. Givan, and H. S. Chang, "A framework for simulation-based network control via hindsight optimization," in *CDC*. IEEE, 2000.

[8] S. W. Yoon, A. Fern, and R. Givan, "FF-Replan: A baseline for probabilistic planning." in *ICAPS*, 2007.

[9] L. E. Pineda and S. Zilberstein, "Planning under uncertainty using reduced models: Revisiting determinization," in *ICAPS*, 2014.

[10] L. Macedo and A. Cardoso, "Case-based, decision-theoretic, HTN planning," in *ECCBR*. Springer, 2004.

[11] U. Kuter, D. Nau, M. Pistore, and P. Traverso, "Task decomposition on abstract states, for planning under nondeterminism," *Artificial Intelligence*, 2009.

[12] E. A. Hansen and S. Zilberstein, "LAO: A heuristic search algorithm that finds solutions with loops," *Artificial Intelligence*, 2001.

[13] S. Zilberstein, "Using anytime algorithms in intelligent systems," *AI magazine*, 1996.

[14] B. Bonet and H. Geffner, "Planning as heuristic search," *Artificial Intelligence*, 2001.

[15] N. Muscettola, P. P. Nayak, B. Pell, and B. C. Williams, "Remote agent: To boldly go where no AI system has gone before," *Artificial intelligence*, 1998.

[16] A. K. Jónsson, P. H. Morris, N. Muscettola, K. Rajan, and B. D. Smith, "Planning in Interplanetary Space," in *ICAIPS*, 2000.

[17] S. Chien, R. Sherwood, D. Tran, B. Cichy, G. Rabideau, R. Castano, A. Davis, D. Mandl, S. Frye, B. Trout, *et al.*, "Using autonomy flight software to improve science return on Earth Observing One," *Journal of Aerospace Computing, Information, and Communication*, 2005.

[18] S. A. Chien, R. Knight, A. Stechert, R. Sherwood, and G. Rabideau, "Using Iterative Repair to Improve the Responsiveness of Planning and Scheduling." in *ICAIPS*, 2000.

[19] S. Chien, D. Tran, G. Rabideau, S. Schaffer, D. Mandl, and S. Frye, "Timeline-based space operations scheduling with external constraints," in *ICAPS*, 2010.

[20] S. Chien, J. Doubleday, D. R. Thompson, K. L. Wagstaff, J. Bellardo, C. Francis, E. Baumgarten, A. Williams, E. Yee, E. Stanton, *et al.*, "Onboard autonomy on the intelligent payload experiment cubesat mission," *Journal of Aerospace Information Systems*, 2017.

[21] G. Rabideau and E. Benowitz, "Prototyping an Onboard Scheduler for the Mars 2020 Rover," in *IWPSS*, June 2017.

[22] D. Gaines, R. Anderson, G. Doran, W. Huffman, H. Justice, R. Mackey, G. Rabideau, A. Vasavada, V. Verma, T. Estlin, *et al.*, "Productivity challenges for Mars rover operations," in *ICAPS Workshop on Planning and Robotics*. London, UK, 2016.

[23] W. Chi, S. Chien, J. Agrawal, G. Rabideau, E. Benowitz, D. Gaines, E. Fosse, S. Kuhn, and J. Biehl, "Embedding a Scheduler in Execution for a Planetary Rover," in *ICAPS*, June 2018.

[24] J. Agrawal, W. Chi, S. Chien, G. Rabideau, S. Kuhn, and D. Gaines, "Enabling Limited Resource-Bounded Disjunction in Scheduling," in *IWPSS*, July 2019.

[25] A. Ben-Tal, L. El Ghaoui, and A. Nemirovski, *Robust Optimization*. Princeton University Press, 2009.

[26] D. Bertsimas and D. B. Brown, "Constructing uncertainty sets for robust linear optimization," *Operations Research*, 2009.