Agile Satellite Planning for Multi-Payload Observations in Earth Science Models

Rich Levinson[1] **Sreeja Nag**[2] **Vinay Ravindra**[2]

NASA Ames Research Center, Moffett Field, CA 94035
[1]KBR Wyle Services, LLC, Moffett Field, CA 94035
[2]Bay Area Environmental Research Institute, Moffett Field, CA 94035

Abstract

We present the planner methods and preliminary results for a new paradigm of earth observing systems based on coordinating science observations for a constellation of agile satellites which can quickly maneuver to change viewing angles. Each satellite has multiple heterogeneous instruments.....

Introduction

State of the art earth observing satellites are slow to react to rapidly changing phenomena. Modifying an observation plan may take several days of command formulation and testing by engineers, then mission operations must review and test, before the changes are uploaded to the satellite for further testing. This long process, developed to work with large, expensive satellites with limited maneuverability, means they cannot react to rapidly changing natural phenomena such as rain. A new class of smaller satellites opens the door to a new type of earth science involving a constellation of more agile and less expensive satellites. This provides an opportunity for more rapid and direct coupling between model updates and planning new observations to improve the models based on a model quality metric. NASA calls this new paradigm which involves model-driven observation planning, the New Observing Strategy (NOS) (Moigne et al.)

We present the planning component for one such NOS, called D-SHIELD, being developed at NASA Ames Research Center, in collaboration with teams at University of Texas A&M and the University of Southern California (Nag et al. 2020). D-SHIELD's planner may be the first to create coordinated plans for multiple agile satellites, each with multiple instruments (payloads). The planner operates in a closed-loop context, updating the plan as it receives regular sensor updates. This application domain has not previously been defined in formal planning terms. We describe the planning challenges and solution, detailing the search space and search procedure, and present preliminary experiment results. Contributions include initial dentification of the planner's search space, constraints, heuristics and performance metrics.

Problem Description

D-SHIELD (Distributed Spacecraft with Heuristic Intelligence to Enable Logistical Decisions) involves using a (proposed) constellation of satellites looking at earth to improve soil moisture models which may be used to predict potential flooding locations. Each satellite includes 2 different x-ray instruments (L-band and P-band) to take images of ground positions (GP). The satellites are agile, meaning they can quickly maneuver to change viewing angles. Observations taken by the constellation of small, agile satellites are coordinated by a centralized planner. Observations may combine both instruments and/or multiple observations of a GP at multiple times to improve the quality of the soil moisture model. Model quality is measured in terms of the model error associated with each GP.

D-SHIELD uses high-fidelity models of spacecraft and soil moisture dynamics, combined with a constraint satisfaction planner to generate new observation plans for all satellites in the constellation quickly. The planner's objective is to ensure that model quality meets a minimum threshold. Model quality is inversely proportional to the model error associated with each GP. Model error is a combination of "knowledge error" (prior knowledge about the GP's soil moisture state) combined with "measurement error" associated with new observations. Knowledge error increases over time without new observations and increases when rain occurs. Measurement error is a function of which instrument (L-band or P-band) is used, the viewing angle, and the type of ground cover (e.g., baren, shrubs, forest, croplands).

Constraints

The planner must enforce several constraints:

- Image Lock - Each observation requires the instrument to hold its viewing angle for 3 seconds. This blocks out slewing to another viewing angle during that 3-second image lock period.
- Duplicate observations - We do not look at the same position twice in the same 24-hour period in order to cover more unobserved locations. There are exceptions for serendipitous cases when we aim at one GP but capture others in the same image, and for intentional cases when we plan a follow-up observation.
- Maneuver constraints (slew time and energy) - The satellite must slew to change viewing angles. There are constraints on how quickly it can change viewing angles, depending on the slew magnitude. This is called the slew time constraint. Energy consumption is also dependent on the slew magnitude. Changing viewing angle takes a different amount of time and en-

ergy depending on the combination of initial angle and target angle. The planner must ensure there is enough time to slew between each observation and track the energy consumed by each slew.

- Energy budget - The planner must ensure energy remains above a 70% minimum charge level. Energy is consumed at a steady rate whenever an instrument is on (taking images) plus a variable amount of energy is consumed with each slew. The charge level is increased at a steady rate by solar panels, except during eclipse when the charge is not increased. The planner ensures we do not generate plans which consume more energy than available.

Energy model and constraints: The planner tracks the amount of energy and battery charge for each satellite at each plan step (observation). The model includes:

- Initial energy available.
- Energy consumed per instrument operation
- Energy consumed while slewing between viewing angles
- Energy produced by the solar panels (except for eclipse periods which are also modeled).
- Minimum allowable energy level.

Measurement Error Table:
The science team provides an error table which defines the expected measurement error for all combinations of instruments at and viewing angles. The planner uses this information when choosing observation commands, preferring measurements with the least error.

Follow-up observations. The planner may choose to have a satellite take multiple images of same GP to minimize error. Technically it's observing the same region/viewing angle, which includes the GP. Examples include:

- Same instrument at different times (at the same or different angles)
- Both instruments at the same time (at the same or at different angles)
- Both instruments at different times (may involve different orderings of the instruments)

A choice involving a follow-up observation within 2 hours is considered a single observation by the science model. Scheduling a follow-pup means that sometimes a choice about what to do at one time requires reserving a timeslot in the future, placing constraints on a future TP. In these cases when we explicitly want to observe a GP twice with different viewing angles at different times, the planner must choose the ordering for the viewing angles. This is an exception to our default constraint of no duplicate observations, driven by the measurement error table indicating that the follow-up image will significantly reduce error.

The overall goal is to demonstrate a closed-loop system which continually updates a model of potential flooding regions, dynamically scheduling new observations to improve the model where the error is greatest, such as right after rain occurs, or places which have not been observed recently. Our dynamic soil moisture model is designed to detect regional model quality degradation in near real-time, providing input the planner about the highest priority GP to observe next. Ideally, the planner should be able to guarantee a minimum quality level for all GP in the model. This paper focuses on the D-Shield planner more than the science model.

Planner challenges:
The GP search space is very large, with roughly 1,600,000 GP total. For 3 satellites over a 6-hour period: The constellation as a whole sees a total of 1,518,344 GP distributed over 23,086 TP. Each satellite can see an average of 606,448 GP, distributed over an average of 7,795 TP. Each satellite has 2 instruments, each may be used at 62 possible viewing angles, and instruments may be used independently or in various combinations. This results in an average of 51 command choices/TP with high variance, including a maximum of 108 command choices/TP. These combinatorics are explosive due to

- The large number of GP choices and different permutations of instruments and viewing angles available per TP.
- The large number of GP choices: different combinations of times, instruments, and viewing angles available to view any given GP.
- Multiple satellites may view the same GP at different times

Another key challenge is that the physical search space (defined by orbits and instrument command choices) is different from the science search space (defined by science rewards associated with each plan). Contributions of this work include initial identification, quantification, and qualitative characterization of these two different search spaces along with providing an initial integration between them. NOS is a new and evolving paradigm, requiring flexible methods to represent and explore various plan constraints and quality metrics as we learn more about the domain properties.

The highly non-linear constraints on slewing and valid instrument combinations presents another challenge. We initially started with a MILP formulation but required more direct heuristic search control over the order in which variables and values are chosen than was possible with a standard MILP solver.

Figure 1 shows the D-Shield system. The process is initialized with inputs based on the satellite orbits and specifications for the spacecraft and instruments, which are used to determine available observation times, along with slew time and energy requirements. Those raw inputs go

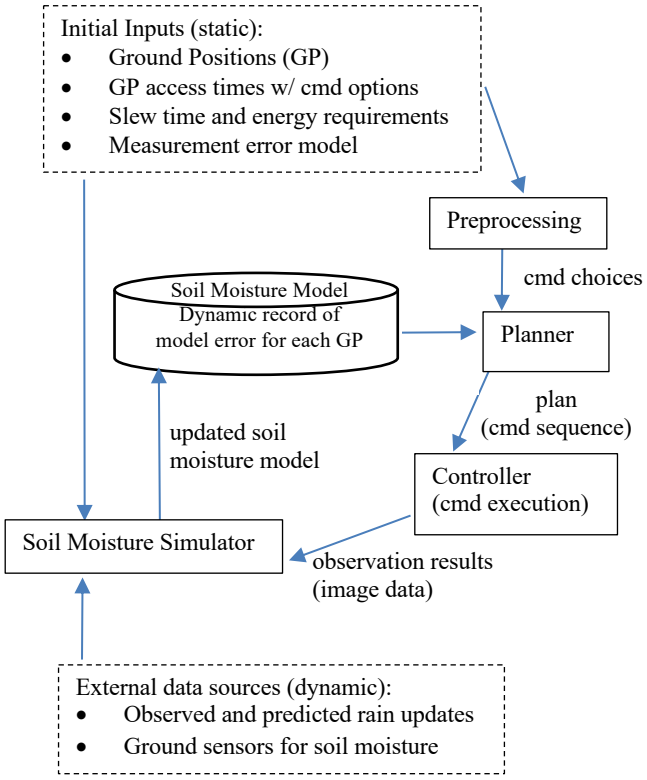

Figure 1: D-Shield architecture

through pre-processing to generate the planner input files. The planner searches through the space of all available observation times and decides what to look at, when to look at it, and how to look at it (which combination of instruments and viewing angles). The planner produces a sequence of commands for the Controller to execute (command the instruments to take the images).

The Controller collects the observation data and passes that to the Science Simulation Model, to update the model based on the new observations. The Dynamic Soil model is a database which maintains a recording of the current model error for each GP. The planner uses this information to sort the GP in decreasing order from highest error to lowest.

Continual re-planning: D-SHIELD is designed to run iteratively and continually. Each planning iteration generates a plan for 6 hours (configurable), after which the plan is executed producing new measurements. Those new observations are passed to the science model which updates its model and produces an updated set of GP scores. This runs continuously over a course of multiple days, tracking the error associated with each GP over time, and generating plans to reduce model error as it changes over time.

The planning objective is to create a coordinated multi-satellite observation plan which improves the model quality by observing the GP which the highest model error. Each plan step is an observation, an instrument command of the form <time, instrument (L and/or P), viewing angle>, which specifies the time when one or both instruments will take images at the given viewing angle. We partition the earth's surface 'tiles' which represent the region which can be imaged at a given viewing angle. The viewingAngle defines which tile we will observe. Each tile is 9-km x 9-km and contains multiple GP. Thus, each plan step covers multiple GP.

Solution

Preprocessing:
We start with a pre-processing phase which consolidates, reformats and compresses the following heterogeneous raw input data sources.

- GP file: The planner reads in a file defining 1,662,486 ground positions (observation targets) along with their biome types (e.g., forest, grasslands, shrubbery, ocean).
- Access time files specify the times when each satellite can view each GP, based on satellite orbits. This file contains triples <time, gp, viewingAngle>.
- Rain files specifying the GP where rain was recently observed, along with where and when rain is predicted.
- Saturation files specifying which GP are already saturated with water.
- Measurement error files specify measurement errors associated with each combination of instruments and viewing angles, for each biome type, as a tuple: <instrument1, viewingAngle1, instrument2, viewingAngle2, biomeType>.
- Slew file specifies the time and energy required to slew (maneuver) between all viewing angle combinations.

Preprocessing removes redundancies from the raw inputs which are produced by multiple independent and heterogeneous systems, and filters data to remove data from the raw inputs which are not required by the planner. A key part of pre-processing is a step called choice flattening which eliminates redundancies in the raw input because there are multiple options to see the same GP at the same time. This eliminates about 66% of the initial (redundant) choices found in the raw input data.

The purpose of preprocessing is to prepare the data for the planner by producing files which are structured specifically to define the planner's search space. Preprocessing produces multiple files as output, defining two search spaces which must be integrated during planning (Figure 2). There is a separate TP choice file for each satellite but a single GP choice file for the whole constellation.

TP choices: Command choices and times for viewing each GP (per satellite)
Command search space.
One file for each satellite

Time	cmd choices	GP covered by choice
1311:	L.32:	[3165]
	L.34:	[3445, 3446]
	P.32:	[3165]
	P.33:	[3165]
	P.34:	[3445, 3446]

GP choices: When and how is best for viewing GP
Science search space used for local heuristics.
One file for whole constellation

GP	satellite	Time	cmd choices	measurement error
3165:	s1:	1311	[L.32]	.038
	s1:	1311	[P.32]	.003
	s1:	1311	[L.32 & P.32]	.003
	s2:	1259	[L.33]	.003
	s2:	1259	[P.33]	.003
	s2:	1259	[L.33 & P.33]	.003

Figure 2: TP choices and GP choices are planner inputs

Figure 2 shows the two types of files produced by preprocessing which are the planner inputs. The TP choice file (left) defines all of the timepoints (TP) when some GP is visible. There is one TP choice file for each satellite because each satellite covers GP at different times and each executes an independent command sequence. The Time column identifies the unique TPs when the satellite can see at least one GP. Each TP corresponds to a decision variable in the planner's search space, and the command choices for that TP correspond to the domain for that decision variable. The planner decides which command to choose at each TP for each satellite.

The Time column identifies the unique TPs when the satellite can see at least one GP. Each TP corresponds to a decision variable in the planner's search space, and the command choices for that TP correspond to the domain for that decision variable. The planner decides which command to choose at each TP for each satellite.

In the TP choices file, each TP maps to a set of command choices, and each command is associated with a set of GP which will be covered by that observation. Command choices are denoted as <instrument(s).viewingAngle>. For example L.32 means L-band at viewing angle 32. Note that a single command choice may actually involve two observations. The last cmd choice 'L32.P.32' means both instruments (L and P) will take an image at angle 32 at the same time. Note also that the planner may choose either L.32 or P.32 or both. All three will cover GP 3165, but with different measurement errors. The planner must choose a command for every timepoint when there is something to look at. This is the primary search space for the planner. It starts at the first TP and marches chronologically forward until it fills up all of the available timepoints.

The other file is the GP choice file (right), which defines all of the choices for how to look at a specific GP. This is a GP-centric view which specifies which command choices are better from each GP's perspective. These GP choices are used by the planner for local heuristics to sort the command choices at each TP choice. The local heuristic ranks the command choices for each GP in order of increasing measurement error. Each GP maps to a set of command choices for each satellite which can view it. These GP choices define different times and commands for viewing the same GP for all satellites. Each GP choice is associated with a different instrument error (based on that GP's biome type). These choices include all combinations of access times, instruments, and viewing angles. Note that there are three choices to view GP 3165 at time 1311 at angle 32, and another 3 choices to view it at time 12597 at angle 33. Each of these choices has a different measurement error.

Planning Approach: Heuristically guided Constraint Programming

The planner creates an observation plan for team of satellites, each with two instruments. The planner decides what to look at, when to look at it, & how to look at it. The plan is a sequence of time-indexed commands for both instruments, for each satellite. The planner chooses a command for every timepoint (TP) when the satellite can view a ground position (GP). There are 3 command types:

- `TakeImage(<instrument1, viewingAngle1> <instrument2, viewingAngle2>.` If both instruments are used at the same TP, they must point at the same angle, but the planner may choose to use the instruments at different times (within 2 hours) in which case the angles may be different. This command takes 3 seconds to execute.
- `SlewToAngle(angle):` This command takes a variable amount of time to execute depending on the initial and target angles.
- `Idle():` Turn off the instrument to save energy.

Figure 3 shows an example of planner output for one of the satellites. A file like this is produced for each satellite in the constellation because they each execute a separate observation command sequence. This example says: From time 2 through 4, take an observation using the P-band at angle 48, which covers GP 1236024. Then remain idle for 10 seconds from time 5 through 14. Then use the L-band at the same angle 48 from time 15 through 17. Then there is a

Plan for sat 1:

Time	Command
[2-4]	P.48
[5-14]	Idle
[15-17]	L.48
[18-36]	Idle
[37-40]	Slew
[41-43]	L.44
[44-45]	Slew
[46-48]	P.45
[49-49]	Idle
[50-51]	Slew
[52-54]	P.46

Figure 3: Example planner output for one satellite

23 second gap until the next observation at time 41. The next observation is at a different viewing angle, so slewing is required, taking 4 seconds of that gap, ending at or before time 40, so the new observation may start at time 41.

Planner Design

We are solving this as a Constraint Optimization Problem (COP) (Dechter 2003), which is defined generically as:

- Set X of variables $\{x_i, .. x_n\}$
- Set D of variable domains $\{d_i, .. d_n\}$, one for each variable
- Set C of constraints specifying legal variable combinations
- Satisfiability requirement: Find consistent set of variable assignments for all variables for hard constraints
- Optimization objective: minimize the cost for soft constraints

Our specific DSHIELD COP is defined as:

Problem: Assign commands for every satellite for every TP when it can observe any GP. The TP when a satellite can observe a GP is called an *access time*. There are far too many GP to them all so this is inherently an optimization problem. The objective is to reduce the average model error by observing as many high-error GP as possible.

Decision Variables:

We define a set of decision variables x_t^s, each representing the command choice for sat s at time t. $\forall\, \boldsymbol{t} \in T^s$, T^s= {All GP access times (TP) for sat s}

Variable Domains:

We define a set of variable domains d_t^s representing command choices for each x_t^s, $\forall\, \boldsymbol{t} \in$ {access times}. The domain of choices for x_t^s is the set of all command options for sat s at time t.

$d_t^s \in$ {(<instrument1, viewingAngle1>, <instrument2, viewingAngle2>)}

The region on the ground covered by each instrument is approximately 9 km x 9 km, which is larger than a GP, which is a single <lat, lon> point. This means every observation covers multiple GP. Some biome types are excluded such as frozen or ocean and GP of those types are ignored.

Our hard constraints include the 3-second image lock, and slew time, and energy budget. Our soft constraints (preferences) include minimizing energy consumption and maximizing the # of high-priority GP observed. We use a depth-first branch-and-bound (BnB) algorithm to optimize the soft constraints (Dechter 2003).

Objective

We want to maximize the model quality (minimize error) for every GP. We define GP err = f(knowledgeErr, measurementErr). This score represents the quality of the model for a given GP. KnowledgeError is a function of how old (stale) the prior data is and how much rain has occured.

Planner objective: minimize $\Sigma_g\, err_g \quad \forall\, g \in G$

Search Space is a Node tree:

Each node contains:

- Set of decision variables x_t^s, each representing the command choice for sat s at time t. $\forall\, \boldsymbol{t} \in T^s$, T^s= {All GP access times (TP) for sat s}
- Set of variable domains d_t^s representing command choices for each x_t^s, $\forall\, \boldsymbol{t} \in$ {access times}. The domain of choices for x_t^s is the set of all command options for sat s at time t.
- A node state for tracking the effects of each choice is a Python dictionary tracking state fluents such the current battery charge level.

The root node is initialized with variables for every TP for every satellite.

Root Node variables:
$x_0^1, x_1^1, x_2^1, x_2^2, x_3^1, x_3^2, x_4^2 x_5^2, x_6^1, x_6^2$, ...

Figure 4: Decision variables for the root node

Figure 4 shows and example of the decision variables for the root node. x_t^s = the command(s) for sat s at time t. The root node variables are sorted chronologically, so all variables for time N precede all variables for times greater than N. There are variables only for the times when the satellite as access to a GP. This example shows variables for sat 1 only at times 0, 1, and 6, there are variables for both sats 1 and 2 at times 2 and 3, and variables for sat 2 only at times 4 and 5. This is because those are the only times when the given sat has TP choices. Technically we may choose to solve the variables in any order, but our default is to solve them in chronological order. For the example in this paper, there are an average of 7,795 variables per satellite.

Node state: Each node also contains a 'state' property, which tracks the energy consumption and battery charge at each plan step.

D-Shield CSP Search Procedure:

```
1.  CreatePlan()
2.     rootNode.vars = {x_t^s}
3.     openNodes = {rootNode}
4.     while openNodes:
5.        node = chooseNode(openNodes)  // choose plan
6.        var  = chooseVariable(node)   // choose x_t^s
7.        val  = chooseValue(var)       // choose cmd
8.        child = createChildNode(node)
9.        child.vars.remove(var)
10.       child.addPlanStep(var, val)
11.       child.propagateChoices(var, val)
12.       child.updateState()    // update energy cost
13.       if child.isFeasible()  // verify energy is OK
14.       then openNodes.add(child)
```

Figure 5: **Search Procedure**

Figure 5 shows the search procedure. The root node is initialized with variables for all TP for all satellites, shown in figure 4. The algorithm iteratively chooses a node to expand, then chooses a variable from that node, and finally chooses a value for that variable. This is a generic search procedure which may be used for different planning problems. Lines 5 through 7, `chooseNode`, `chooseVariable`, and `chooseValue` are implemented by domain specific callback methods.

In our case, choosing a node corresponds to choosing a plan (the command sequence defined by the path from that node to the root). Choosing a variable corresponds to choosing a <sat, TP> pair and choosing the value corresponds to choosing a command for that sat at time TP. We have used this same system to schedule data downlinks by providing a different set of callbacks for these methods. Future work will integrate the observation planner with that downlink planner.

`chooseNode` currently uses a simple depth first approach, choosing the most recently added plan node. This method also implements our depth-first BnB algorithm for constraint optimization by backtracking to a previous node when a child node is pruned. Nodes are pruned if they violate the hard energy budget constraint, or if they are suboptimal according to the BnB algorithm. This method can also implement Monte Carlo rollouts which we may explore in future work.

`chooseVariable` currently chooses variables in chronological TP order.

`chooseValue` implements the local heuristics and is where most of the work occurs. The values are instrument command choices which may be sorted in various ways. For example, we have explored the following 4 local heuristics:

- **GP count** - commands are sorted in decreasing order of the # of GP covered by the command
- **GP model error -** commands are sorted in decreasing order of sum of model error for each GP covered (choose commands which cover the most GP with highest model error)
- **GP measurement error -** commands are sorted in decreasing order of sum of measurement errors for each GP covered, based on the measurement error for that command. Measurement errors depend on biome type, so the same measurement may have different errors based on the GP biome type. For example, soil moisture is harder to measure accurately in shrubbery than in baren land.
- **GP choice rank -** commands are sorted in decreasing order of "GP preference". Each GP 'ranks' the command options, where rank 1 means the best possible option to view the GP, and rank 2 is second best, etc.

The experiments in this paper compare 2 of these chooseValue variants: GP count and GP measurement error.

Constraint handlers: Line 11 is a domain specific callback which implements our constraints:

- Image Lock: hold position (viewing angle) for 3 seconds for each observation.
- No duplicate observations (with exceptions).
- Agility Constraints: The satellite must slew to change viewing angles.
- Energy budget: Don't generate plans which consume more energy than available.

Constraints are applied only to variables for the same satellite, except for the duplicate observation constraint which is applied to all variables for all satellites. This means the image lock constraint for sat 1, requires only that sat 1 hold it's position. Sat 2 is unconstrained while Sat 1 is holding position. On the other hand, if sat 1 observes GP 1, then sat 2 is constrained to *not* observe GP 1.

Constraints are enforced through *choice propagation*, which uses *forward checking* (Russell and Norvig 2021) after each choice to remove any future choices which are inconsistent with it. For example, the 3-second image lock is implimented as follows: when a decision is made to start taking an image at tick 10, then all choices for timepoints 11 and 12 are removed so nothing else will be scheduled during that 3-second hold. Choice propagation is also used to enforce constraints for slew time and duplicate observations. We can annotate the removed choices with the name of the constraint which ruled them out, providing a simple form of explainable planning.

The following choice propagation example shows how it used to remove duplicate GP observations from future variables.

(a) $[x_{25}^{s}$: {~~L.32: [123]~~,
L.33: [436349, 436350, 436351],
P.32: [436350, 436351, 436352]]

(b) $[x_{36}^{s}$: ~~{P.42: [253]}]~~

Figure 6: Choice propagation examples

Figure 6 shows two examples of choice propagation. Case (a) shows that after GP 123 is observed, it is removed from the list of GP covered for every choice for all future variables. In this example, that was the only GP covered by command choice L.32, so the command L.32 is removed from the domain of choices for variable x_{25}^{s} (TP 25). Case (b) shows that when the last choice is removed from a variable's domain (producing an empty domain), then the variable is removed from the node. In this case, after GP 253 is observed, it's removed from the GP list for command P.42, leaving an empty GP list. So the command P.42 is removed from the domain for variable x_{36}^{s}, leaving an empty variable domain, so x_{25}^{s} is removed from the list of open variables. This is different from a pure CSP system where all variables must receive a valid assignment, and a variable with an empty domain indicates infeasibility. In our case, choice propagation removes variables which have no valid assignments based on the path dependencies of the current plan (node). Thus the # of assigned variables in the final plan is less than the initial number in the root node.

Explainable constraint processing: Whenever a choice or variable is removed due to constraint enforcement, it is annotated with the name of the constraint which removed it. This provides a simple form of explanation when trying to understand the planner's choices.

Multiple-pass planning: We group the full set of GP into priority groups which correspond to model error cohorts . The first priority are the rainy GP, where there has been recent or predicted rain, leading to increased model uncertainty and error. Second priority are GP where there has not been recent or predicted rain. Both of those groups exclude GP where the ground has already been saturated, which are a third priority.

On the first pass, the planner schedules as many first-priority (rainy) GP as can fit in the 6-hour plan horizon, then it 'backfills' the gaps in that plan with non-rainy GP. When filling a gap, the planner must enforce the slew constraints to splice the new observations into the existing plan. It must slew from the last high-priority viewing angle at the start of the gap and must slew to the next planned high-priority viewing angle at the end of the gap.

Plan "score" objectives:
To implement our BnB optimization algorithm, the node selection method chooses nodes with the best plan score. The plan score is a blend of 3 optimization objectives:

- ***Maximize*** observed Ground Positions (GP)
- ***Minimize*** observation error
- ***Minimize*** energy cost

Preliminary Experiment Results
Preliminary experiment results are presented below for our example with 3 satellites with a 6-hour planning horizon.

Heuristic	# images (makespan)	# of GP observed	Avg Err /GP
MaxGpCount	6,452	24,641	.025
MinGpChoiceErr	6,445	14,401	.015

Figure 7: Preliminary experiment results comparing two `chooseValue` heuristics.

Figure 7 compares two variants of the `chooseValue()` local heuristic (line 7 in Figure 5) vs. total # of observed GP and the average error per observed GP. All GP are initialized with a default error of 0.04 (the soil moisture model standard for maximum allowable error). The two heuristics (described above) are:

- MaxGpCoverage - command choices are sorted in decreasing order of # of GP covered by the cmd.
- MinGpChoiceErr - cmd choices are sorted in increasing order of the sum of errors for all GP covered by the cmd.

These initial results clearly show the trade between maximizing the # of GP observed vs. minimizing the measurement error.

Future work involves similar experiments with other heuristic variants. We are exploring a ranked-choice voting scheme: command choices for each TP are ranked by the collective set of GP covered by all commands choices for that TP. Each GP ranks the command choices for that TP from 1st choice to last choice. The command which receives the most #1 votes will be selected.

GP-choice search space: We plan to explore planning observations based on GP choices as the primary search space rather than using TP choices as our search space. Intuitively this means solving the most important GP first rather than solving each TP in chronological order. This can be implemented by changing the `chooseVariable()` method to select variables which cover GP with the highest error in non-chronological order.

Downlink planning: We previously used this same COP system to implement a downlink planner. That planner tracked two priority levels of collected data and scheduled timeslots on ground stations to download the data in priority order. It used the same algorithm as figure 5, but was provided with different callbacks for `chooseVariable()`

and `chooseValue()`. We plan to integrate that downlink planner with the observation planner presented here.

Conclusion

References

Dechter. R, 2003. Constraint Processing. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

J. L. Moigne, M. M. Little and M. C. Cole, "New Observing Strategy (NOS) for Future Earth Science Missions," IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium, 2019, pp. 5285-5288, doi: 10.1109/IGARSS.2019.8898096.

Nag, S., Moghaddam, M., Selva, D., Frank, J, Ravindra, V., Levinson, R., Azemati, A., Aguilar, A., Li, A., Akbar, R., 2020. D-Shield: Distributed Spacecraft with Heuristic Intelligence to Enable Logistical Decisions, Proc. of the 2020 IEEE International Geoscience and Remote Sensing Symposium.

Russell, S., Norvig, P. 2020. Artificial Intelligence: A Modern Approach. Fourth Edition. Pearson.