Optimal Planning to Coordinate Science Data Collection and Downlink for a Constellation of Agile Satellites with Limited Storage

Richard Levinson[1,2] , **Vinay Ravindra**[1,3] , **Sreeja Roy-Singh**[1,3]
[1]NASA Ames Research Center, Moffett Field, CA
[2]KBR Wyle Services, Moffett Field, CA
[3]Bay Area Environmental Research Institute, Moffett Field, CA
{rich.levinson, vinay.ravindra, sreeja.nag}@nasa.gov

Abstract

We present a novel Mixed Integer Linear Program formulation that produces optimal plans for a constellation of remote sensing satellites. The generalized formulation is applied to an operational NASA constellation to improve wildfire danger prediction. The planner generates integrated data collection and downlink plans for multiple agile satellites with limited storage capacity, minimum energy requirements, and temporal constraints. Observation targets and modes are associated with science rewards. The planner maximizes the aggregate rewards collected for all observations on all satellites.

Our generalized model for integrated data collection and downlink uses a novel interval-based abstraction called *Data Cycles*, without time-indexed variables. Data cycles organize the multitude of observation and downlink opportunities from 1 second granularity into sequences of data collection and downlink intervals. Experiments using large-scale real-world data yield optimal 24-hr plans for an eight satellite constellation, which capture 99% of the ∼23,000 available targets and 99.9% of available science rewards.

1 Problem Summary and Application

We present a generalized Mixed Integer Linear Program (MILP) formulation which includes many practical model elements required for producing coordinated plans for a constellation of agile satellites. Each satellite has a set of data collection and downlink opportunities. The planner chooses when, where (targets), and how (mode) each satellite will collect and downlink data to maximize aggregate science rewards. The satellites have limited data storage, requiring downlinks to free up space for more observations. Energy usage is tracked to ensure a minimum state of charge. The model enforces sequence and setup time constraints when two satellites see the same ground station (GS), or when two GS see the same satellite. It includes multiple observation modes with setup times to switch between them. Energy and storage constraints depend on the observation mode or GS selected. Multi-satellite coordination is achieved by ensuring each target is counted only once in the objective, even if seen multiple times.

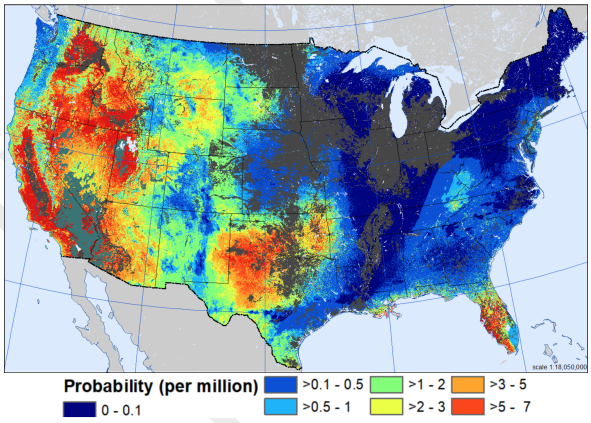

Figure 1: USGS Large Fire Probability (WLFP) Forecast for 8/1/20

The generalized MILP planner is then applied to a NASA mission to generate an optimal plan for agile, spaceborne observation and downlink that will improve the forecast of wildfire danger in pre-fire conditions. NASA's currently active CYGNSS mission [Ruf *et al.*, 2018] consists of eight spacecraft, each carrying a Global Navigation Satellite System - Reflectometry (GNSS-R) instrument. The instrument collects GNSS signals reflected off the Earth's surface, from which geophysical properties may be inferred. CYGNSS sensors usually remain ON in a low volume data collection mode, where information is compressed on-board (with loss). CYGNSS was originally designed to track cyclones. Current NASA scheduling for CYGNSS does not use automated planning. Nominally, the mission continually gathers and downlinks compressed data, with only occasional schedule changes planned by humans.

NASA has recently been using CYGNSS "off-label" to detect burned areas after a wildfire. For that case, a high volume data collection mode called *RawIF* is used where observations in raw format are downlinked without compression. Uncompressed data potentially holds information lost during compression, providing finer resolution images. Each observation involves collecting multiple signals reflected off the ground. Compressing the data reduces the number of reflected signals collected, and reduces the spatial resolution

and extent of the image. An open-source Earth Observation Simulator calculates time windows for observation and downlink opportunities, and eclipse intervals [Ravindra *et al.*, 2021]. CYGNSS observations are collected over the contiguous US and GS are in Hawaii, Chile and Australia.

The U.S. Geological Survey (USGS) produces a *wildfire danger probability* to estimate wildfire risk based on factors including weather, vegetation, and fuel moisture. Figure 1 shows the USGS WFPI-based Large Fire Potential (WLFP) forecast for August 1, 2020 [USGS, 2025]. This heatmap shows a 1-km grid of observation target locations and defines each target's science reward. Each of the ∼47,000 grid points is assigned a WLFP value between 0 and 7, yielding ∼23,000 targets with non-zero rewards. WFPI is the Wildland Fire Potential Index, a unit-less number related to vegetation flammability. WLFP (figure 1) is derived from WFPI. They are both USGS terms. Optimal measurements are expected to improve forecasts using neural network methods offline [Roy-Singh *et al.*, 2025], [Nag *et al.*, 2024].

2 Related Research

There is extensive research that addresses agile satellite scheduling for Earth Observation [Liu *et al.*, 2017; Kangaslahti *et al.*, 2024; Lemaitre *et al.*, 2002], but far less on formulations that guarantee optimality applied to real missions [Wang *et al.*, 2021]. Near-optimal solutions typically involve a single satellite [Herrmann and Schaub, 2023; Kucuk and Yildiz, 2019], or observations only [Chen *et al.*, 2019; Frank *et al.*, 2016; Levinson *et al.*, 2022; Nag *et al.*, 2018], or downlinks only [Spangelo *et al.*, 2015], or scheduling them independently [Cho *et al.*, 2018]. There is less work on an integrated model for simultaneously planning data collection and downlink, and the few that enforce data storage constraints use synthetic, randomly generated targets such as [Xiao *et al.*, 2019]. We present work using large-scale, real-world data for orbits, target locations, science rewards.

[Cho *et al.*, 2018] provide a good survey of related work. They propose constructing and solving separate MILP models for planning observations and downlinks in a 2-step process. Step 1: Solve a downlink MILP to assign satellites to downlink windows. Step 2: Solve a second MILP to assign observations, given the downlink assignments from Step 1. They use orbits from historical missions, with up to 700 *randomly generated targets* for 1 satellite, or 500 for 3 satellites.

[Xiao *et al.*, 2019] present a variation of the "flow shop" problem where there are two machines: observe and downlink. They present an interval-based MILP model for multi-satellite observation and downlink scheduling, where each *task* is an observation and downlink pair. They integrate observations and data downlinks but have *no storage limit*, and all targets must be observed. They present experiments with 10 tasks in a four day plan horizon, or 20 tasks in 8 days, with up to two satellites, and use synthetic data only. Their objective is to minimize completion time (makespan).

[Spangelo *et al.*, 2015] present a MILP model for *downlinks only* (no observation planning) and a *single satellite*. Their objective is to maximize the total amount of data downlinked. Different ground stations may have different data and energy consumption rates, so it may save energy to skip over one GS and wait for the next one. They split the plan horizon for each satellite into intervals delimited by passes over ground stations, similar to our model.

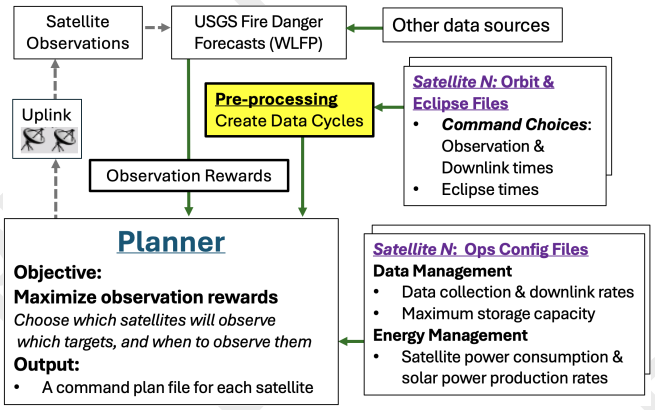

Figure 2: Planner Inputs and Outputs

[Levinson *et al.*, 2022] present methods for scheduling observations for multiple agile satellites which have multiple instruments (P-band and L-band radar) and can slew to 62 different viewing angles, and target rewards depend on the instrument type and viewing angle. It involves observation scheduling only, with no downlinks or storage limits.

[Valicka *et al.*, 2019] solve a stochastic MIP model for scheduling observations only, but with observation uncertainty, for multiple satellites.

[Monmousseau, 2021] presents a system designed to match Planet Lab's heuristic simulated annealing method, and requires solving a sequence of 4 different MILPs: First it optimizes for one objective, then another, until finally maximizing rewards. It assumes events are continuously rescheduled (modifying an existing plan), so first it minimizes the number of plan changes required (minimum perturbation), tracking which "events" must be rescheduled (turned on/off).

[Lee *et al.*, 2024] maximize observation count rather than rewards, and do not consider observation modes which have different energy and data consumption rates. Their evaluation includes 3 satellites and 5 targets.

3 Planner Model and Methods

Our ground-based, centralized planner produces plans to maximize the aggregate science rewards of observed targets for multiple agile satellites. Figure 2 shows the inputs and outputs. Pre-processing, highlighted in yellow, converts raw orbital data into a data cycle abstraction used by the Planner. For improving wildfire forecasts, key inputs include the WLFP forecast shown in Figure 1, observation and downlink opportunity times, and eclipse times for each satellite. Key performance indicators are the total science rewards collected over all observations, and the number of targets and images observed (each image covers multiple targets). Each observation collects one *image* which covers multiple targets, and different images may cover the same targets.

Observation Modes: The satellites are capable of switch-

Command Choices for Satellite N		
Time	**Command Choices**	**Targets**
21	Oberve(L3)	12,4,39,66,17,29
21	Observe(S7)	4,39,66
643	*Downlink*	Hawaii
---	---	---
849	*Downlink*	Hawaii
2020	*Downlink*	**Australia-1**
---	---	---
2641	*Downlink*	**Australia-1**
2132	*Downlink*	**Australia-2**
---	---	---
2750	*Downlink*	**Australia-2**
3585	Observe(L4)	88,3,10,92,55,16
3585	Observe(S5)	3,10,92

Figure 3: The *Choice File* defines each satellite's choices

ing between *multiple observation modes* that represent combinations of different instruments, footprints, viewing angles, etc. Some viewing angles may degrade the maximum reward value which is based on WLFP (Figure 1). The model is capable of supporting any combination of instruments, angles or other operational parameters because the details for data collection and downlink rates are calculated during preprocessing and treated as application-specific input parameters, appearing as constants in the MILP model. Our experiments use observation modes combining two different sensor footprints: *Large* and *Small* with a range of viewing angles that require setup time to change. *Large* images use the CYGNSS RawIF mode, and consume 1/60 of buffer capacity. *Small* images cover half the targets covered by *Large* images, and consume half the storage (1/120 of capacity). The Large images correspond to the actual RawIF images used on CYGNSS, while the small images were introduced for experimental purposes for the generalized model.

Data Storage Model: Our data storage model is based on the CYGNSS satellites [Ruf *et al.*, 2018], which use First-In-First-Out (FIFO) buffers, that hold 60 seconds of *Large* image (RawIF) data or 120 seconds of *Small* image data. Observations take one second to collect, and ∼20 seconds to downlink. It takes ∼20 downlink minutes to empty a full buffer.

Constraints: The planner enforces these constraints:

1. No data collection allowed when storage is full
2. No downlinks allowed when storage is empty
3. Satellites can downlink to only one GS at a time
4. Each GS can connect with only one satellite at a time
5. Battery charge can never dip below a minimum level
6. Target rewards are counted only once in the objective, even when observed by multiple satellites

Plan Choices: Figure 3 shows the *Choice File*, which defines the command choices for one satellite. All choices are optional, and the satellite may remain idle at anytime. Each row defines a time-indexed command option for every second when the satellite may collect or downlink data. There may be multiple observation or downlink choices at one time, but only one command can be executed at a time. The first two rows show a choice between two observation modes, *L3* or

Cycle	Observation Phase	Time Gap	Downlink Phase
1	Times: 4695-5410 (716 sec) Image IDs: 1-667 (667)	5411-8277 (2867 secs)	Times: 8278-10423 (2146 s) Available Downlink secs: 965
	Plan: *Selected Images: 60, Downlink Secs: 963* *Storage: Avail 100% - Used 100% (60 images) + Freed 80% (963 secs) = 80%* *Downlink: AUS: 8282-8821 (540 secs) + HI: 9999-10421 (423 secs)*		
2	Times: 10683-11426 (744 s) Image IDs: 668-1369 (702)	11427-14437 (3011 secs)	Times: 14438-16363 (1926 s) Available Downlink secs: 360
	Plan: *Selected Images: 48, Downlink Secs: 344* *Storage: Avail 80% - Used 80% (48 images) + Freed 29% (344 secs) = 29%* *Downlink: AUS: 14445-14649 (205 s) + HI: 16224-16362 (139)*		
3	Times: 16736 - 17129 (394) Image IDs: 1370-1669 (300)	17130-22276 (5146 secs)	Times: 22277-70133 (47857 s) Available Downlink secs: 4548
	Plan: *Selected Images: 17, Downlink Secs: 662* *Storage: Avail 29% - Used 28% (17 images) + Freed: 55% (662 secs) = 56%* *Downlink: AUS: 63801-63941 (141s), CHI: 29834-30302 (469s),* *CHI: 41861-41912 (52s)*		
4	Times: 72640 - 72674 (35) Image IDs: 1670-1682 (13)	72675-75609 (2934 s)	Times: 75610-76165 (556 s) Available Downlink secs: 556
	Plan: *Selected Images: 1, Downlink Secs: 550* *Storage: Avail 56% - Used 1.7% (1 image) + Freed: 46% (550 secs) = 100%* *Downlink: AUS: 75619-76168 (550 secs)*		
5	Times: 78383 - 78870 (488) Image IDs: 1683-2132 (450)	78871-81669 (2798 secs)	Times: 81670 - 83816 (2147 s) Available Downlink secs: 928
	Plan: *Selected Images: 46, Downlink Secs: 912* *Storage: Avail 100% - Used 77% (46 images) + Freed: 76% (912 s) = 100%* *Downlink: AUS: 81681-82207 (527secs), HI: 83433-83818 (386 secs)*		
6	Times: 84186 - 84866 (681) Image IDs: 2133-2782 (650)		*NO DOWNLINK PHASE* *(24-hour horizon ends)*
	Plan: *Selected Images: 60, Downlink Secs: 0* *Storage: Avail 100% - Used 100% (60 images) + Freed: 0% (0 secs) = 0.0 %*		

Figure 4: Data Cycles for one satellite over 24 hours

S7, at time 21. Mode "L3" means a *Large* image at viewing angle 3, covering targets 12, 4, 39, 66, 17, 29. Mode "S7" is a *Small* image at angle 7, covering targets 4, 39, 66. Setup time is required for satellites to change viewing angle, during which no observations or downlinks can occur. At time 643 a downlink window to a GS in Hawaii begins, with a row for each second during the window. Hyphens indicate a contiguous sequence of seconds to the same GS. A new downlink window to Australia-1 begins at time 2020, which overlaps another downlink window to a second GS (Australia-2) beginning at time 2132. More observation opportunities begin at time 3585. The large scale of this problem is a challenge. A 24-hour plan for 8 satellites involves ∼92,000 seconds when there is a binary choice between collecting data or not, or between downlinking data or not, and there are ∼23,000 potential observation targets. This 1-second command granularity is related to the CYGNSS satellite's speed (∼7km/s).

3.1 Data Cycles

From time-indexed orbital data to intervals: We developed a novel interval-based abstraction which enables optimal solutions to be found quickly. Instead of decision variables indexed at the 1 second granularity of the choice file (Figure 3), we model temporal intervals called *data cycles* (Figure 4). Each data cycle is a sequence of 2 phases: First the *observation* phase fills up storage, followed by the *downlink* phase

which frees up storage. Cycles are repeated for the duration of the plan horizon.

Figure 4 shows one satellite's data cycles for 24 hours. The first observation phase is from time 4695 through time 5410. It lasts 716 seconds, with 667 image opportunities, meaning there are gaps when there are no targets. The downlink phase lasts from 8278 to 10423 (2146 seconds), including multiple separated GS overpasses, for a total of 965 seconds of downlink opportunity. This satellite is assigned image ID's 1-2782. These are *possible* observations, only a subset of which will be selected in the plan. The satellite may collect images 1-667 during Cycle 1. Images 668-1369 may be collected during Cycle 2. The buffer can fit up to 60 *Large* or 120 *Small* images, or a mix of the two. Each cycle's plan appears in the shaded boxes, showing the number of images collected, storage consumption and production. Some cycles consume all storage or use all available downlink seconds, and others don't. Note that cycle 3 uses only 662 downlink seconds out of 4548 available, freeing only 55% of storage. In contrast, 100% of the storage is freed in the next two cycles (4 and 5). The last cycle starts with empty storage, fills it with 60 images, then the 24 hour horizon ends.

Pre-processing: The *Data Cycles* in Figure 4 are extracted from the *Choice File* (Figure 3). Algorithm 1 shows pseudocode that creates the Data Cycles for each satellite. Each `cycle` contains lists of possible observation and downlink choices. We also define a downlink window, `dnlWindow`, which is a contact interval for a given GS. In Algorithm 1, each row in the choice file is parsed to identify the `time`, `command` and `targets`. Each possible observation is assigned an image ID. Cycles start with the observation phase, ignoring downlink choices until after data has been collected. Each observation is appended to the `cycle.obs` list. If the command is an observation and there is a prior downlink window, that marks the end of the prior cycle, and a new cycle is started. If the command is a downlink and the cycle already contains observations, then we check if this command continues the current downlink window. As long as the GS remains unchanged and time increments by only 1 second, the duration of `dnlWindow` is extended. If the next row in the choice file has a break longer than 1 second or the GS changes, then a new downlink window is started. All cycles are collected into `satCycles`, which is returned as the result.

Data Cycle Properties: Data cycles organize the flat Choice File into a sequence of knapsack-like problems for each satellite. Algorithm 1 converts the Choice File into a sequence of intervals (cycles), each containing a sequence of sub-intervals for repeatedly filling and unfilling the data storage. This imposes hierarchy (temporal containment) and sequence constraints which are not in the flat Choice File. The start and end of each cycle define the sparse timepoints where the MILP enforces resource constraints (data and energy), rather than enforcing on every timepoint listed in the Choice File.

Data Cycles may be viewed as a novel *refillable* variant of the *knapsack problem*, where each satellite repeatedly fills, then (partially) empties its data storage *knapsack*. The knapsack problem is not usually iterative, while our satellites may repeatedly empty and refill the knapsacks. Our variant may be described as an: iterative, multiple, refillable, and semi-continuous knapsack. It's *iterative* because each satellite fills up its single knapsack multiple times, and it's *multiple* because there are multiple satellites, each with its own knapsack. *Continuous* means fractions of an item can be added to a knapsack. Our model is *semi-continuous* because only whole images can be added but fractions may be removed.

Figure 4 shows how data cycles help visualize the overall high-level problem constraints, making it easy to see the range and upper bounds on the available observations and downlink seconds per cycle, and thus the aggregate upper bounds on the whole 24 hour plan horizon. For example, cycle 1 has 667 image opportunities, while cycle 4 has only 35. Figure 4 also shows how sparsely distributed observation and downlink events are within the 24-hr horizon. This visualization can help inform future mission design by identifying feasible usage bounds on resources, given the operational constraints and mission requirements.

Algorithm 1: Create Data Cycles from Choice File

```
Input  : Choice file for satellite s, firstImageID
Output: The data cycles for satellite s
cycle ← new Cycle()
cycle.obs ←{}, cycle.dnl ← {}
dnlWindow ← None
imageID ← firstImageID
for row in choice file do
    time, command, targets ← row columns
    if command is observation then
        if cycle contains downlinks then
            /* Save cycle, then start a new cycle */
            cycle.dnl.append(dnlWindow)
            satCycles.append(cycle)
            cycle ← new Cycle()
            cycle.obs ←{}, cycle.dnl ← {}
            dnlWindow ← None
        end
        imageID = imageID + 1
        o ={time,command,imageID,targets}
        cycle.obs.append(o)
    else if cycle contains observations then
        /* Command is downlink, but only include
           downlinks after cycle's observation phase */
        gs = targets.first /* single DNL target */
        if (dnlWindow is not None)
           and (time == dnlWindow.end + 1)
           and (gs == dnlWindow.gs) then
            /* Extend end time of DNL window */
            dnlWindow.end = time
        else
            /* Save dnlWindow & start new one */
            if dnlWindow is not None then
                cycle.dnl.append(dnlWindow)
            end
            dnlWindow ← new DnlWindow()
            dnlWindow.start ← time
            dnlWindow.end ← time
            dnlWindow.gs ← gs
        end
    end
end
cycle.dnl.append(dnlWindow)
satCycles.append(cycle)
return satCycles, imageID
```

3.2 Generalized MILP Formulation

In this section, we present the generalized MILP formulation constructed for data cycles.

Input Parameters

S = Set of satellites
K_s = Set of data cycles for satellite $s \in S$
$I_{s,k}$ = Set of images visible by sat s during cycle k
$dur_{s,k}$ = total # of seconds in cycle k on sat s, $1 \leq k \leq |K_s|$
I = Set of image opportunities for all satellites, $i \in I$ is an image id, $1 \leq i \leq |I|$
J_i = Set of targets covered by image i
$J = \bigcup_{i=1}^{|I|} J_i$ =Set of all targets covered by all images
I_j = images which cover target j
T = set of observation modes/types (a set of any symbols)
o_i^M = observation *mode* for image i, $o_i^M \in T$
o_i^T = the observation *time* for image i
m_j = observation modes available for viewing target j
$r_{j,m}$ = reward for observing target j with observation mode m, $\forall j \in J, \forall m \in m_j$ ***(used in objective)***
s^{init} = initial storage available (%), default = 100
G = Set of all ground stations (GS)
W_g = Set of time windows when GS g may downlink data from any satellite
$D_{s,k}$ = Set of downlink windows between sat s and any GS during cycle k
$D_{s,k,g}$ = Set of downlink windows between sat s and GS g during cycle k, $D_{s,k,g} \subset D_{s,k}$
$d_{s,k,g,n} \in D_{s,k,g}$ = the nth downlink window in $D_{s,k,g}$
$d^L_{s,k,g,n}$ = the *Lower* Bound (earliest start) of window $d_{s,k,g,n}$
$d^U_{s,k,g,n}$ = the *Upper* Bound (latest end) of window $d_{s,k,g,n}$
Δs_i^- = Storage % *consumed* by observation mode o_i^M
Δs_g^+ = Storage % *produced* / downlink second for GS g
Δe^+ = Energy % *produced* / second (except during eclipse)
Δe^- = Energy % *consumed* every second of the mission
Δe_i^{o-} = Extra energy % *consumed*/sec for obs mode o_i^M
Δe_g^{d-} = Extra energy % *consumed*/sec of downlink to GS g
e^{init} = initial energy available (%) default = 100
e^{min} = minimum energy (battery state of charge)
$eclipse_{s,k}$ = number of eclipse seconds in cycle k on sat s
c^G = setup time for GS to change between satellites
c^S= setup time for satellite to change between GS
$c^O_{i,j}$= setup time to change obs modes from o_i^M to o_j^M (sensor setup time + slew time to change view angles)

Binary Decision Variables

$x_i = 1 \Leftrightarrow$ image i is in the plan, $\forall i : 1 \leq i \leq |I|$
$y_{j,m} = 1 \Leftrightarrow$ target j is planned to be viewed with observation mode $m \in T, \forall j : 1 \leq j \leq |J|$ ***(used in objective)***
$z_{s,k,g,n} = 1 \Leftrightarrow$ sat s downlinks to g in window n of $D_{s,k,g}$
$v_{g,s1,k1,n1,s2,k2,n2} = 1 \Leftrightarrow$ GS g sees satellites s_1 and s_2 at the same time: Slots $d_{s1,k1,g,n1}$ & $d_{s2,k2,g,n2}$ overlap, and both slots are in the plan (requiring deconfliction)
$w_{g,s1,k1,n1,s2,k2,n2} = 1 \Leftrightarrow v_{g,s1,k1,n1,s2,k2,n2} = 1$ and downlinking in $d_{s1,k1,g,n1}$ ends *before* downlinking in $d_{s2,k2,g,n2}$ begins (sequence for satellite deconfliction)
$q_{s,k,g1,n1,g2,n2} = 1 \Leftrightarrow$ Sat s sees GS g_1 and g_2 at the same time: Slots $d_{s,k,g1,n1}$ & $d_{s,k,g2,n2}$ overlap, and both slots are in the plan (requiring deconfliction)
$u_{s,k,g1,n1,g2,n2} = 1 \Leftrightarrow q_{s,k,g1,n1,g2,n2} = 1$ and downlinking in $d_{s,k,g1,n1}$ ends *before* downlinking in $d_{s,k,g2,n2}$ begins (sequence for GS deconfliction)

Continuous Decision Variables

$sa_{s,k}$ = Storage % *available* for sat s on cycle k, $0 \leq sa_{s,k} \leq 100$
$sc_{s,k}$ = Storage % *consumed* for sat s on cycle k, $0 \leq sc_{s,k} \leq 100$
$sp_{s,k}$ = Storage % *produced* for sat s on cycle k, $0 \leq sp_{s,k} \leq 100$
$e^S_{s,k}$ = Energy % *available* on sat s at *start* of cycle k, *after* being capped at 100%, $e^{min} \leq e^S_{s,k} \leq 100$
$e^E_{s,k}$ = Energy % *available* on sat s at *end* of cycle k, $0 \leq e^E_{s,k} \leq 100 + \Delta e^+ dur_{s,k}$ (solar panel energy production may exceed 100%)

$p_{s,k,g,n}$ = number of downlink seconds planned for $d_{s,k,g,n}$ $0 \leq p_{s,k,g,n} \leq d^U_{s,k,g,n} - d^L_{s,k,g,n} + 1$
$t_{s,k,g,n}$ = start time of downlink during $d_{s,k,g,n}$ $0 \leq t_{s,k,g,n} \leq d^U_{s,k,g,n}$

Objective: The objective is to maximize the sum of rewards (quantified using WLFP forecasts shown in Figure 1) for all observed targets.

$$Maximize \sum_{j=1}^{|J|} \sum_{m \in m_j} r_{j,m}\, y_{j,m} \tag{1}$$

Constraints:

$$\sum_{m \in m_j} y_{j,m} \leq 1 \quad \forall j \in J \tag{2}$$

$$y_{j,m} \leq \sum_{i \in I_j, where\ \ o_i^M = m} x_i \quad \forall j \in J \tag{3}$$

$$sa_{s,1} = s^{init} \quad \forall s \in S \tag{4}$$

$$sa_{s,k} = sa_{s,k-1} - sc_{s,k-1} + sp_{s,k-1} \quad \forall s \in S, \forall k \in K_s \tag{5}$$

$$sc_{s,k} \leq sa_{s,k} \quad \forall s, k \tag{6}$$

$$sc_{s,k} = \sum_{i \in I_{s,k}} \Delta s_i^- x_i \quad \forall s \in S, \forall k \in K_s \tag{7}$$

$$sp_{s,k} = \sum_{\forall g \in G, \forall n < |D_{s,k}|} \Delta s_g^+ \ p_{s,k,g,n} \quad \forall s, k \tag{8}$$

$$sp_{s,k} \leq 100 - (sa_{s,k} - sc_{s,k}) \quad \forall s, k \tag{9}$$

$$e^S_{s,1} = e^{init} \quad \forall s \in S \tag{10}$$

$$e^S_{s,k} = min(e^E_{s,k-1}, 100) \quad \forall s, k \tag{11}$$

$$e^E_{s,k} = e^S_{s,k} + \Delta e^+ \ (dur_{s,k} - eclipse_{s,k}) - (\Delta e^- dur_{s,k}) - \textstyle\sum_{i \in I_{s,k}} \Delta e_i^{o-} x_i - \sum_{\forall g \in G} \Delta e_g^{d-} p_{s,k,g,n}, \forall s, k \tag{12}$$

$$z_{s,g,k,n} \leq p_{s,k,g,n} \quad \forall s, k, g, \forall n \in 1, ..., |D_{s,k,g}| \tag{13}$$

$$p_{s,g,k,n} \leq M \;\; z_{s,k,g,n} \;\; \forall s,k,g,\forall n \in 1,...,|D_{s,k,g}| \quad (14)$$

$$t_{s,k,g,n} \leq M \;\; z_{s,k,g,n} \;\; \forall s,k,g,\forall n \in 1,...,|D_{s,k,g}| \quad (15)$$

$$d^L_{s,k,g,n} - t_{s,k,g,n} \leq M(1 - z_{s,k,g,n}) \quad \forall s,k,g,n \quad (16)$$

$$t_{s,k,g,n} + p_{s,k,g,n} - d^U_{s,k,g,n} \leq M(1 - z_{s,k,g,n}) \forall s,k,g,n \quad (17)$$

$$v_{g,s1,k1,n1,s2,k2,n2} \leq z_{s1,k1,g,n1} \quad \forall v \;\; variables \quad (18)$$

$$v_{g,s1,k1,n1,s2,k2,n2} \leq z_{s2,k2,g,n2} \quad \forall v \;\; variables \quad (19)$$

$$v_{g,s1,k1,n1,s2,k2,n2} \geq z_{s1,k1,g,n1} + z_{s2,k2,g,n2} - 1 \, \forall v \quad (20)$$

$$w_{g,s1,k1,n1,s2,k2,n2} + w_{g,s2,k2,n2,s1,k1,n1} = v_{g,s1,k1,n1,s2,k2,n2} \quad (21)$$

$$t_{s1,k1,g,n1} + p_{s1,k1,g,n1} + c^G - t_{s2,k2,g,n2} \leq M(1 - w_{g,s1,k1,n1,s2,k2,n2}), \;\; \text{where } s1 \neq s2 \quad (22)$$

$$q_{s,k,g1,n1,g2,n2} \leq z_{s,k,g1,n1} \quad \forall q \;\; variables \quad (23)$$

$$q_{s,k,g1,n1,g2,n2} \leq z_{s,k,g2,n2} \quad \forall q \;\; variables \quad (24)$$

$$q_{s,k,g1,n1,g2,n2} \geq z_{s,k,g1,n1} + z_{s,k,g2,n2} - 1 \, \forall q \quad (25)$$

$$u_{s,k,g1,n1,g2,n2} + u_{s,k,g2,n2,g1,n1} = q_{s,k,g1,n1,g2,n2} \quad (26)$$

$$t_{s,k,g1,n1} + p_{s,k,g1,n1} + c^S - t_{s,k,g2,n2} \leq M(1 - u_{s,k,g1,n1,g2,n2}), \quad \text{where } g1 \neq g2 \quad (27)$$

$$x_i + x_j \leq 1 \quad (28)$$

$\forall$ images $i, j \in I_{s,k}$ which cannot be scheduled together due to setup time requirements (see Algorithm 2).

Equation (1) shows the objective, which maximizes the aggregate reward for all observed targets, where the reward depends on the observation mode. Constraints (2) ensure each target contributes to the objective at most once. This is a maximization problem, so optimal solutions select the most rewarding observation mode m for each target. Constraints (3) ensure: If target j is observed with mode m, then at least one image capturing j with mode m must be the plan. Note the *objective* is a function of the *targets* selected, but the storage and energy *constraints* are driven by the number of *images* selected.

Storage Constraints: Constraints (4) set the initial storage available to s^{init}. In our example all satellites start with 100% storage. Constraints (5) ensure the storage available at the start of cycle k = (storage available at the start of prior cycle) - (storage consumed on prior cycle) + (storage produced on prior cycle). Constraints (6) ensure no cycle uses more storage than is available. Constraints (7) track the storage *consumed* by satellite s on cycle k, where storage consumed per image depends on each image's observation mode. This constraint leverages the fact that image ids are contiguous integers for each satellite (see Algorithm 1). Constraints (8) calculate storage produced during data cycle k as the sum of storage produced from all downlinks in cycle k, where the downlink rate depends on the GS. Constraints (9) ensure the plan never downlinks when storage is empty.

Energy Constraints: Constraints (10) set the initial energy level to e^{init}. In our example all satellites start with 100% energy. Constraints (11) set each cycle's available energy to the energy remaining at the end of the prior cycle, unless that exceeds 100%, in which case it is capped at 100. Constraints (12) calculate the total energy which would be available at the *end* of the cycle, assuming infinite battery capacity, but Constraints (11) ensure it never exceeds 100%. Constraints (12) say: energy available for satellite s the *end* of data cycle k = (the energy available at the cycle start) + (energy produced by solar panels when not in eclipse) - (energy consumed by nominal operations) - (energy consumed for data collection which depends on the observation mode m) - (energy consumed downlinking which depends on the GS g). Energy can never dip below the minimum energy threshold because the lower bound of the decision variable $e^S_{s,k}$ is set to e^{min}.

Algorithm 2: Create Command Mutex Constraint

Input: $I_{s,k}$ = Set of images visible by sat s on cycle k
Output: Mutex constraints are created to enforce setup time to change observation methods

```
for i ∈ I_{s,k} do
    for j from i + 1 to max(I_{s,k}) do
        if o_i^M ≠ o_j^M and o_j^T − o_i^T <= c^O_{i,j}
        then
            add mutex constraint (28): x_i + x_j ≤ 1
        end
    end
end
```

Downlink constraints: Constraints (13) through (17) track which downlink windows are used, and ensure downlinks are scheduled within selected windows. The variable $z_{s,g,k,n}$ indicates satellite s is scheduled to downlink to GS g during the nth downlink window in cycle k. Constraints (13) and (14) ensure $z_{s,g,k,n}$ = 1 *if and only if* downlink time is scheduled in window $d_{s,k,g,n}$. Constraints (15) ensure the downlink start time t is 0 if the downlink window $d_{s,k,g,n}$ is not selected. Constraints (16) and (17) say: If downlink window $d_{s,k,g,n}$ is in the plan, then the start time t for downlinking must be within that window. Conditional constraints (14) through (17) use *Big-M* notation where M is a large constant. The *Big-M* notation refers to a way to model conditional constraints in MILP. It is used to "activate/deactivate" constraints based on binary variables being one or zero. These conditional constraints are similar to [Cho *et al.*, 2018], but designed to work with our unified data collection and downlink model. [Cho *et al.*, 2018] must commit to imposing a total order on all overlapping downlink windows, while we only enforce those constraints when the overlapping time slots are in the plan.

Satellite Conflict Resolution: Constraints (18) through (22) resolve conflicts *when multiple satellites see the same GS at the same time*, by enforcing a sequence so that GS g is assigned to only one satellite at a time, and includes setup time for the GS to switch between satellites (c^G). Constraints (18) through (20) ensure indicator variable v = 1 *if and only if* overlapping downlink windows n_1 and n_2 are both in the plan (using the z indicator variable). Constraints (21) use the indicator variables w to ensure: if v = 1, then either n_1 precedes n_2 or the other way around. Constraints (22) enforce that sequence constraint only if w indicates it is required.

GS conflict Resolution: Constraints (23) through (27) resolve conflicts *when a satellite can see multiple GS at the*

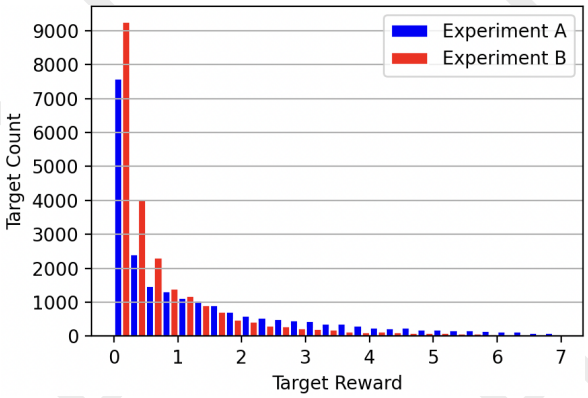

Figure 5: Target Reward Histograms (based on WLFP forecasts)

				Rewards			Targets			Images	
Test: Exp. A/B	Min. Reward	MIP Gap	Solve Time	Plan (objective)	Avail-able	Rwd %	Plan	Avail-able	Target %	Plan	Avail-able
1: A	0.0	0.0009	12 h	31,336	31,374	99.9	21,955	22,536	97.4	1,841	42,934
2: A	0.0	0.001	12 h	31,334	31,374	99.9	21,945	22,536	97.4	1,852	42,934
3: A	0.05	0.0005	12 h	31,304	31,326	99.9	20,515	20,667	99.3	1,846	42,787
4: A*	**0.05**	**0.0008**	**1.8 h**	**31,293**	**31,326**	**99.9**	**20,463**	**20,667**	**99.0**	**1,845**	**42,787**
5: A*	**0.1**	**0.0001**	**28 m**	**31,171**	**31,180**	**99.9**	**18,728**	**18,745**	**99.9**	**1,847**	**42,107**
6: B	0.0	0.0037	12 h	20,130	20,366	98.8	21,433	23,275	92.1	1,843	42,804
7: B	0.1	0.0023	12 h	19,963	20,123	99.2	18,330	19,036	96.3	1,833	41,897
8: B	0.15	0.0017	12 h	19,768	19,882	99.4	16,685	17,113	97.5	1,826	41,208
9: B*	**0.2**	**0.001**	**7.1 h**	**19,514**	**19,577**	**99.7**	**15,172**	**15,363**	**98.8**	**1,820**	**40,151**
10: B	0.25	0.0003	12 h	19,252	19,280	99.9	13,969	14,038	99.5	1,807	39,128
11: B*	**0.25**	**0.0009**	**9.1 m**	**19,243**	**19,280**	**99.8**	**13,946**	**14,038**	**99.3**	**1,820**	**39,128**
12: B	0.275	0.0001	12 h	19,115	19,135	99.9	13,442	13,485	99.7	1,822	38,684
13: B*	**0.28**	**0.0001**	**52 m**	**19,087**	**19,105**	**99.9**	**13,339**	**13,378**	**99.7**	**1,817**	**38,598**
14: B*	**0.3**	**0.0002**	**3 m**	**18,966**	**18,983**	**99.9**	**12,919**	**12,956**	**99.7**	**1,816**	**38,168**

Figure 6: Test results (* = optimal, solver stops before 12-hr limit)

same time, by enforcing a sequence to ensure the satellite is assigned to only one GS at a time. The temporal separation includes a setup time for the satellite to switch between GS (c^S). These constraints are very similar to constraints (18) through (22) described above.

Setup Times for Changing Observation Modes: Constraints (28) create mutex constraints between any two observations that occur too close together for the required setup time to switch modes. If the setup time to switch between modes is 5 seconds, then plans cannot include observations with different observation modes less than 5 seconds apart. Algorithm 2 shows how constraints (28) are created, leveraging knowledge that image id's are created in chronological order and are contiguous within a cycle (see Algorithm 1). Algorithm 2 works as follows: Loop through all pairs of observations for satellite s on cycle k. If the observations use different modes, and the time between them is less than the required setup time between observations, $c^O_{i,j}$, then create a mutex constraint to ensure at most one of the observations is in the plan.

Linearity Assumption: [Spangelo *et al.*, 2015] identify a linearity assumption shared by our model because both models check energy and storage constraints at the end of the same interval. Under some eclipse-related edge-case conditions, it is possible for energy to dip below minimum mid-cycle and then recover to nominal before the interval ends (because recharging is prevented during an eclipse). Their proposed solution incrementally discretizes the interval into subintervals to constrain energy at finer time scales.

Model abstractions and efficiencies: The model tracks storage and energy as percentages of capacity available at the beginning and end of each cycle. Instead of tracking ∼92,000 seconds with time-indexed variables, this model tracks resource constraints only at the start and end of ∼49 data cycle intervals. It does not track specific times or the order images are collected. Downlink times are modeled only when deconfliction is required, when multiple satellites see the same GS, or vice versa. The model doesn't track when specific images are downlinked. The planner chooses the optimal number of downlink seconds, which is not the same as a greedy downlink policy that always downlinks data when possible. Figure 4 shows cycle 3 does not downlink all the data, because that would not improve the optimal objective. During *post-processing*, each selected observation and downlink command in the plan is mapped to a specific time-indexed choice in the *choice file* (Figure 3). We can trace the downlinked image IDs to specific ground stations and times because the data is stored in a FIFO buffer.

4 Evaluation

All experiments were run on a 2023 MacBook Pro, with an M3 Max chip, 36 GB Memory, using Python 3.8. Our MILP solver was Gurobi version 11.0.3 [Gurobi, 2025]. Our data sets and code are available upon request.

We evaluate our system with two data sets. Figure 5 shows the target reward histograms for Experiments A and B. This shows there are many more low-value targets than high-value targets. Experiment A is a *high* fire danger case from 8/1/2020, which was a very active fire season ("A" for "Active"). Figure 1 shows the USGS Wildfire Danger Predictions for that date, which provides the basis for the target rewards in Experiment A. Experiment B is a *lower* fire danger case from 7/15/21. That year was less active and consequently the target rewards were generally lower in Experiment B. The total available rewards for Experiment A are ∼30K vs. ∼20K for Experiment B.

Test Cases: Figure 6 shows 14 tests: Test 1 through 5 use Experiment A data, and Tests 6 through 14 use Experiment B data. Three experiment parameters are used: *Minimum reward*, *MIP gap* (Mixed Integer Program gap), and *Solve Time* (columns 2-4). *Minimum Reward* specifies a minimum target reward threshold used to filter out low-reward targets. Figure 5 shows there are thousands of low-value targets with rewards less than 0.3 (the largest reward threshold used in our experiments). *Time limit* specifies the maximum amount of solve time allowed before returning the best answer found. All of our tests use a 12 hour time limit, but the solver terminates early when it finds a solution within the MIP Gap tolerance which defines an optimal solution. The MIP gap tolerance specifies a minimum gap tolerance between the objective and the solver's best estimate of an

upper bound for the objective. The MIP gap is defined as: $|upperBound - objective|/|objective|$. By default, Gurobi considers a solution to be optimal when the MIP Gap is less than 0.0001. It may take many hours to reduce the gap from 0.001 to 0.0001, or it may never get there.

Analysis: Figure 6 summarizes our results, sorted (and numbered) from highest to lowest objective. The objective is the column labeled *Plan (Objective)* and highlighted in yellow. Optimal results are indicated by the (*) and highlighted in blue bold text, and are the only tests which terminate before the 12-hour limit (see Solve Time). In these cases the MIP gap tolerance was set to 0.001, except test 13, which reached an optimal solution in 52 minutes using the default Gap of 0.0001. The **Rewards** columns are: the plan rewards (objective), the total available reward (sum of rewards for all available targets), and the reward % is the percentage of available rewards covered in the plan. The **Targets** columns are: the number of targets covered in the plan, the number of targets available, and the % of available targets covered by the plan. The **Images** section shows the number of images collected in the plan and total number of available images.

Our key performance metric is *Rwd %*, highlighted in yellow. $Rwd\% = objective/availableRewards * 100$. This shows more than 99% of available rewards are captured in all tests, except test 6 which captures 98.8%. All tests also capture more than 96% of available targets, except test 6, which captured 92%. The optimal plans include only *Large* images, but when the planner is stopped before reaching optimality, plans do sometimes include *Small* images.

Plan Quality Trades: Notice in Figure 6 that the *objective is higher for the suboptimal cases* (see Test 1 vs. Tests 4 and 5). Recall the objective is the sum of all rewards for all observed targets. By definition, a solution is optimal if the solver reaches the specified minimum gap tolerance before reaching the time limit. These are the tests marked with (*) and highlighted in blue in Figure 6. The trade is: We can reach the minimum gap tolerance by removing some targets, thus reducing the maximum possible objective. Figure 6 shows that in both experiments, the objective increases as the number of selected targets increase, while the gap decreases as the number of available targets decreases. For example, tests 1-3 have higher objectives than the optimal results for tests 4 and 5. Test 4, with a minimum reward threshold of 0.05 (and MIP Gap tolerance of 0.001) achieved the optimal score in 1.8 hours, and Test 5 achieved optimality in 28 minutes with the default MIP Gap tolerance of 0.0001. Tests 4 and 5 have lower *optimal* objectives than the *suboptimal* objectives of tests 1-3. In the experiment B tests, as solve time increases, so does the number of observed targets because the number of targets per image increases.

This shows the *trades between solver time, the number of targets, optimality and objective*. Increasing the value threshold helps to reach an optimal solution faster, but reduces the objective's upper bound because we are removing some available targets. In Test 14, with the largest reward threshold of 0.3, an optimal solution is found in only three minutes. Mission scientists may regret filtering out low-reward targets. This raises science questions about if/when to ignore low-value targets. Note that test 12 timed out after 12-hours with a minimum reward threshold 0.275, but test 13 found an optimal solution in 52 minutes with a minimum reward of 0.28, which filtered out an extra 107 targets. This shows how sensitive solve times are to the number of available targets.

5 Prior Work

The data cycle abstraction and generalized model presented above represent the latest evolution of remote sensing planners for a constellation of satellites. In this section we summarize that prior work to compare results before and after data cycles. In general, each evolution involved increasing the number of satellites and increasing the planning horizon. Data Cycles enabled us to find optimal solutions for the first time, and with more satellites and a longer plan horizon. The prior work used the Choice File (figure 3) as planner input (no data cycles), and created time-indexed binary decision variables for each timepoint (row) in the Choice File.

Soil Moisture Model Error Reduction: The initial science problem was to reduce model error associated with soil moisture predictions [Levinson *et al.*, 2021], [Levinson *et al.*, 2022]. The objective maximized the aggregate error reduction in the soil moisture model for all observed targets. Each observation mode was associated with a quantified model error. *The goal was to observe the targets with the highest model error, using observation modes with the least error.* Observation modes included two sensor types (P-band and L-band radar) and 62 pointing angles per sensor. Setup time was required to change instruments and slew time was required to change angles. The system planned observations only (no downlinks or storage limits). *Problem Size*: 3 Satellites, 6 hour plan.

[Levinson *et al.*, 2021] presented Dynamic Constraint Processing (DCP) with heuristic search methods for this problem, and then [Levinson *et al.*, 2022] introduced a MILP formulation to compare the DCP vs. MILP solutions. PDDL was not considered for this quantitative optimization problem. Results showed DCP achieved objective scores that were ∼58% of the optimal scores provided by MILP, but the MILP scaled poorly due to the large number of conflict resolution constraints required to block mutually exclusive operations. With a reduced plan horizon of only 2 hours, MILP required 24 hours to find the optimal score (174.7), while DCP found the best approximate solution (101.2) in 2.5 minutes (58% of the optimal score). With the full 6 hour plan horizon, the MILP required 90 million mutex constraints, and failed to produce any reasonable solution after a 50 hour time limit.

Wildfire Danger Prediction: The present science problem is improving wildfire danger models. The prior soil moisture planners (DCP and MILP) were adapted for new science rewards, objective function, and satellite operations. The new objective maximizes the aggregate science reward for whole constellation by observing targets with the highest wildfire risk based on the USGS Large Fire Probability Forecast (Figure 1). For this problem, the DCP and MILP planners were extended to plan for both observations and downlinks, with storage limits. Additionally, the search engine used by DCP for the soil moisture application was replaced with Monte Carlo Tree Search (MCTS). This resolved scaling problems

with the prior search engine which maintained all search nodes in memory, compared to MCTS which maintains far fewer nodes in memory.

[Levinson *et al.*, 2024] is an unpublished paper (available online) that explains how the DCP planner was integrated with the MCTS search engine for this application. Before data cycles, the percentage of each image downlinked was tracked at every decision point (every row in the Choice File), which scaled poorly with MILP. Even for the small problem of 1 satellite and a 3 hour plan horizon, Gurobi's MIP Gap remained 1,907% after 3 days. This is largely due to the 10's of millions of constraints required to track the percentage of each image downlinked at each decision point. *Problem Size*: 4 satellites, 12 hour plan horizon. There were ∼28k binary decision variables, timepoints when the planner might choose to collect or downlink data. The MCTS *objective score* was 6,131 (20% of the highest score found with data cycles).

[Levinson, 2024] presents a detailed discussion of *Propel*, the DCP system which uses MCTS as the search engine behind nondeterministic Python (Python with choice points). It describes how MCTS works in this context, and includes the present application as an example. *Problem Size*: 8 satellites, 24 hour plan, 92,000 binary choice points = $2^{92,000}$ states in the search space (before data cycles). The *objective score* was 12,717 (41% of the highest score found data cycles).

[Roy-Singh *et al.*, 2025] consider two classes of observation targets, *active-fire* and *pre-fire*, with different science reward calculations. See this paper for discussion of the various neural network methods used to calculate science rewards for these two different target classes. The objective cannot be directly compared with the present paper because it includes both active fire and pre-fire target rewards, while the present paper includes only pre-fire targets. This work did not use the generalized model. Experiments involved real-world data collected by seven CYGNSS satellites over three recent active fires, and a 24 hour plan horizon.

The present paper builds on the prior work described above, generalizes the satellite operations model, and introduces the data cycle abstraction. To handle multiple observation modes, setup and slew times, and conflict resolution for the wildfire case, we adapted the models originally developed for the soil moisture case. The data cycle abstraction reduces the problem size, enabling optimal 24-hour plans for 8 satellites to be generated for the first time. *Problem Size*: 8 satellites and a 24 hour plan horizon. The highest *objective score* is 31,336 (Figure 6, Test 1).

The Data Cycles approach was originally developed for a very different application: generating flight plans for a drone to collect image data with *limited battery instead of limited data storage*. The drone's limited 15-minute battery life required it to return to a charging station repeatedly between flights. *Battery cycles* for the drone were the original model before *data cycles*, which demonstrates how this refillable resource knapsack method applies to different applications.

6 Conclusion

We presented an *"orbital to optimal"* data pipeline, which flows from orbital trajectories to optimal solutions. Raw orbital projections are converted into the flat Choice File (Figure 3) which is at 1 second granularity, which is then converted by Algorithm 1 into the Data Cycles abstraction, which is used as input to the MILP model, which produces optimal solutions.

Our generalized model applies to tasking most state-of-the art remote sensing satellite constellations (CYGNSS is only an example). User-defined observation modes may vary in data volumes/quality and/or sensor field of view (sensor details are abstracted away during pre-processing). More generally, this approach applies to many problems where multiple agents repeatedly consume and replenish a limited resource such as data storage or energy (battery charge).

We presented a parameterized and general MILP formulation with many practical elements for remote sensing applications, including: multiple satellites, integrated data collection and downlinks, limited storage, observation modes, conflict resolution and setup times. We demonstrated results using large-scale real-world data (8 satellites, 24 hour plan horizon, ∼23,000 targets, ∼92,000 seconds when choices must be made) and an environmental application driven objective (wildfire probability). We presented a novel interval-based abstraction for integrated observation and downlink planning without time-indexed decision variables, and demonstrated this problem can be solved optimally quickly, producing plans that collect 99% of available targets and science rewards. We identified trades between target value threshold (filtering out low-reward targets), optimality, and objective score. Optimality may be achieved at the cost of reducing available science rewards marginally. These trades raise science questions about the importance of observing low-value targets. Additional contributions include making our large, real-world data sets and code publicly available.

Acknowledgements

This work is supported by NASA's Earth Science Technology Office and the FireSense Program. We thank the reviewers for helpful feedback and suggestions.

References

[Chen *et al.*, 2019] X. Chen, G. Reinelt, G. Dai, and A. Spitz. A mixed integer linear programming model for multi-satellite scheduling. In *European Journal of Operational Research*, volume 275, 2019.

[Cho *et al.*, 2018] Doo-Hyun Cho, Jun-Hong Kim, Han-Lim Choi, and Jaemyung Anh. Optimization-based scheduling method for agile earth-observing satellite constellation. volume 15, 2018.

[Frank *et al.*, 2016] Jeremy Frank, Minh Do, and Tony Tran. Scheduling ocean color observations for a geo-stationary satellite. In *Proceedings of the International Conference on Automatid Planning and Scheduling (ICAPS 2016)*. ICAPS, 2016.

[Gurobi, 2025] Gurobi. Gurobi Optimization. https://www.gurobi.com/, 2025. Accessed online 8-Jan-2025.

[Herrmann and Schaub, 2023] Adam Herrmann and Hanspeter Schaub. Reinforcement learning for the

agile earth-observing satellite scheduling problem. In *IEEE Transactions of Aerospace and Electronic Systems*. IEEE, 2023.

[Kangaslahti *et al.*, 2024] Akseli Kangaslahti, Alberto Candela, Jason Swope, Qing Yue, and Steve Chien. Dynamic targeting of satellite observations incorporating slewing costs and complex observation utility. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024.

[Kucuk and Yildiz, 2019] M. Kucuk and S. T. Yildiz. A mixed integer linear programming model for multi-satellite schedulinga constraint programming approach for agile earth observation satellite scheduling problem. In *9th International Conference on Recent Advances in Space Technologies*, 2019.

[Lee *et al.*, 2024] Minkeon Lee, Seunghyeon Yu, Kybeom Kwon, Myungshin Lee, Junghyun Lee, and Heungseob Kim. Mixed-integer linear programming model for scheduling missions and communications of multiple satellites. In *Aerospace*, volume 11, 2024.

[Lemaitre *et al.*, 2002] M. Lemaitre, G. Verfaillie, F. Jouhaud, J.M. Lachiver, and N. Bataille. Selecting and scheduling observations of agile satellites. In *Aerospace Science and Technology*, volume 6, 2002.

[Levinson *et al.*, 2021] Richard Levinson, Sreeja Nag, and Vinay Ravindra. Agile satellite planning for multi-payload observations for earth science. In *Proceedings of the International Workshop on Planning and Scheduling for Space (IWPSS)*, 2021.

[Levinson *et al.*, 2022] Richard Levinson, Samantha Niemoeller, Vinay Ravindra, and Sreeja Nag. Planning satellite swarm measurements for earth science models: Comparing constraint processing and milp methods. In *Proceedings of the Int'l Conference on Automated Planning and Scheduling (ICAPS 2022)*, 2022.

[Levinson *et al.*, 2024] Richard Levinson, Sreeja Nag, and Vinay Ravindra. Planning coordinated observations and downlinks for multiple satellites with limited data storage for wildfire danger prediction, (unpublished). https://brainaid.com/pubs/dshield.MCTS.24.pdf, 2024.

[Levinson, 2024] Richard Levinson. Monte carlo tree search for integrated planning, learning, and execution in non-deterministic python. In *Proceedings of the Int'l Conference on Automatic Planning and Scheduling, Workshop of Bridging the Gap Between AI Planning and Reinforcement Learning (ICAPS 2024 Workshop on PRL)*, 2024.

[Liu *et al.*, 2017] X. Liu, G. Laporte, Y. Chen, and R. He. An adaptive large neighborhood search metaheuristic for agile satellite scheduling with time-dependent transition time. In *Computers Operations Research*, volume 86, 2017.

[Monmousseau, 2021] Philippe Monmousseau. Scheduling of a constellation of satellites: Creating a mixed-integer linear model. In *Journal of Optimization Theory and Applications*, 2021.

[Nag *et al.*, 2018] Sreeja Nag, Alan Li, and James Merrick. Scheduling algorithms for rapid imaging using agile cubesat constellations. volume 61, pages 891–913, 2018.

[Nag *et al.*, 2024] Sreeja Nag, Vinay Ravindra, Richard Levinson, Mahta Moghaddam, Kurtis Nelson, Jan Mandel, Adam Kochanski, Angel Caus, Amer Melebari, Archana Kannan, and Ryan Ketzner. Distributed spacecraft with heuristic intelligence to monitor wildfire spread for responsive control. In *IGARSS 2024 - 2024 IEEE International Geoscience and Remote Sensing Symposium*, 2024.

[Ravindra *et al.*, 2021] Vinay Ravindra, Ryan Ketzner, and Sreeja Nag. Earth observation simulator (eo-sim): An open-source softward for observation systems design. In *IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pages 7682–7685, 2021.

[Roy-Singh *et al.*, 2025] Sreeja Roy-Singh, Vinay Ravindra, Richard Levinson, Mahta Moghaddam, Jan Mandel, Adam Kochanski, Angel Caus, Kurtis Nelson, Samira Alkaee Taleghan, Archana Kannan, and Amer Melebari. Optimal planning and machine learning for responsive tracking and enhanced forecasting of wildfires using a spacecraft constellation. In *34th International Joint Conference on Artificial Intelligence (IJCAI 2025), Workshop on Artificial Intelligence for Sustainability*, 2025.

[Ruf *et al.*, 2018] Christopher Ruf, Clara Chew, Timothy Lang, Mary Morris, Kyle Nave, Aaron Ridley, and Rajeswari Balasubraminiam. A new paradigm in earth environmental monitoring with the cygnss small satellite constellation. In *Scientific Reports*, 2018.

[Spangelo *et al.*, 2015] Sara Spangelo, James Cutler, Kyle Gilson, and Amy Cohn. Optimization-based scheduling for the single-satellite, multi-ground station communication problem. volume 57, 2015.

[USGS, 2025] USGS. WFPI-based Large Fire Probability. https://www.usgs.gov/fire-danger-forecast/wfpi-based-large-fire-probability-wlfp, 2025. Accessed online 8-Jan-2025.

[Valicka *et al.*, 2019] C. Valicka, D. Garcia, A. Staid, J.P. Watson, G. Hackebeil, S. Rathinam, and L. Ntaimo. Mixed-integer programming models for optimal constellation scheduling given cloud cover uncertainty. In *European Journal of Operational Research*, volume 275, 2019.

[Wang *et al.*, 2021] X. Wang, G. Wu, L. Xing, and W. Pedrycz. Agile earth observation satellite scheduling over 20 years: Formulations, methods, and future directions. In *IEEE Systems Journal*, volume 15, 2021.

[Xiao *et al.*, 2019] Yiyong Xiao, Siyue Zhang, Pei Yang, Meng You, and Meng Huang. A two-stage flowshop scheme for the multi-satellite observation and data-downlink scheduling problem considering weather uncertainties. volume 188, 2019.