SCHEDULING THE MAPPING OF A PLANET UNDER GEOMETRICAL CONSTRAINTS*

Adrien Maillard, Steve Chien, Christopher Wells

Jet Propulsion Laboratory, California Institute of Technology, Pasadena, USA
e-mail: name.surname@jpl.nasa.gov

ABSTRACT

Scheduling the coverage of a planet by scientific instruments on board spacecrafts under observational constraints is central in a significant number of current and future missions for the exploration of the solar system. In this paper, we describe the components and algorithms of a software used to study early-phase mission design or to schedule daily operations of currently in-flight spacecraft. The scheduling problem at hand is usually a large combinatorial problem. A discretization process is described and several ordering algorithms are designed and compared. Experiments show that high-quality schedules can be produced by approaches combining reasoning about rolling, coverage and priorities.

1. INTRODUCTION

Scheduling the observation of a planet is a hard combinatorial problem in which usually one or several small sensors on board one or several spacecrafts must observe a set of large areas, under user-set observational constraints, while enforcing hard physical constraints such as energy or memory. This problem is solved with specific software designed to provide *schedules* of observations maximizing a user-set quality criterion. We will see that this type of software is currently used for studying the possibilities offered by specific spacecraft configurations in early mission design phases and for daily operations of in-flight spacecrafts.

Compressed Large-scale Activity Scheduling and Planning (CLASP) is a long-range scheduler [8] for space-based or aerial instruments that can be modelled as push-brooms – 1-dimension line sensors dragged across the surface of the body being observed. It addresses the problem of choosing the orientation and on/off times of a pushbroom instrument or collection of pushbroom instruments such that the schedule covers as many target points as possible, but without oversubscribing memory and energy. Orientation and time of observation is derived from geometric computations that CLASP performs using the SPICE ephemeris toolkit [1].

CLASP allows mission planning teams to start with a baseline mission concept and simulate the mission's science return using models of science observations, spacecraft operations, downlink, and spacecraft orbit. This analysis can then be folded back into many aspects of mission design – including trajectory, spacecraft design, operations concept, and downlink concept. The long planning horizons allow this analysis to span an entire mission.

As a software, there is a distinction between the CLASP *core* and its adaptations to various missions. The CLASP core is designed to be as generic as possible such that a minimal effort to derive it is required to take into account specific needs that are mission-dependent e.g. modelizations of energy or memory or user desires in terms of outputs and schedule quality. CLASP has already been derived several times. Mission planning and mission design inputs on the NASA-ISRO Synthetic Aperture Radar (NISAR), previously DESDynI (Deformation, Ecosystem Structure, and Dynamics of Ice), mission were performed with CLASP [4] [8]. It was also prototyped as a tool for early stage mission planning of the Mars Odyssey THEMIS instrument [11]. CLASP has been used to assess achievement of mission science criteria for the planned Europa Clipper mission and the JUpiter ICy moons Explorer (JUICE) [15]. More recently, the CLASP adaptation for the ECOSTRESS instrument onboard the International Space Station (ISS) [18] concerned long-term campaigns with changing information, mass storage unit operations challenges, and orbit uncertainty. The CLASP adaptation for Orbiting Carbon Observatory-3 Mission (OCO-3) [12] includes models and constraints to enforce on geometric visibility constraints due to occulted zones in the instrument. Notably, the CLASP core contains the scheduling algorithms which generally remain untouched when a derivation happen.

The work whose results are presented in this paper is part of an effort to improve the quality of produced schedules in the core of CLASP. In this paper, we describe in details

An extended version of this work has been submitted to the AIAA Journal of Aerospace Information Systems.

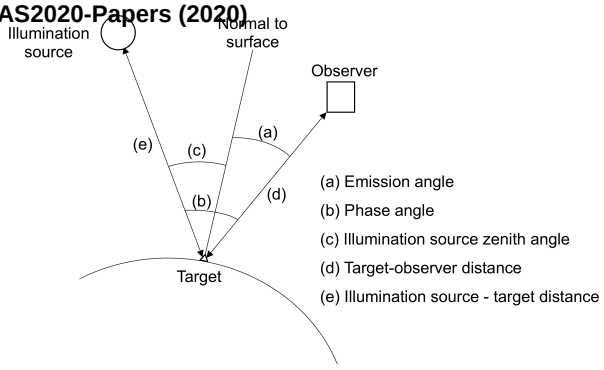

Figure 1: Physical quantities used in observational geometry constraints. Illumination may be the sun or another body.

the current and new algorithms for this software. The remainder of this paper is structured as follows. In Section 2, we describe the Spacecraft Coverage Scheduling Problem (SCSP). In Section 3 and 4, we depict methods to solve the problem, first how it is discretized, reducing it to a selection and ordering problem, and then several ordering algorithms. Finally, we compare these ordering approaches in Section 5.

2. THE SPACECRAFT COVERAGE SCHEDULING PROBLEM

In this section, we present the Spacecraft Coverage Scheduling Problem (SCSP). Users of the system start by submitting *science campaigns* which have to be satisfied. A campaign represents a request to map a small area of the body surface under prescribed conditions. The scientist specifies a polygon on the surface, along with the instrument mode to be used and optional constraints on when data can be acquired (e.g., seasonal, local time, relative position of the sun).

All campaigns are assigned priorities based on preferences specified by the scientists. For example, mapping campaigns are assigned priorities partly based on how close previous observations have met an assigned target allocation for the specified data type. In the end, the generated campaigns and priorities are passed as inputs to CLASP. Science campaigns are also associated with required instrument modes, geometry constraints on the observation (e.g. left or right looking, ascending or descending node) and a desired number of observations over the planning horizon. Note that a single region of interest may have several campaigns attached to it. There are various overflight constraints that we do not describe here. Physical quantities used in these constraints are shown on Fig. 1.

These campaigns are to be realized over a given planning horizon by a spacecraft orbiting or encountering the targeted body (e.g. fly-bys). An important distinction exists between Conventional Observing Spacecrafts (COS) and Agile Observing Spacecrafts (AOS). In this paper, we address the problem of scheduling observations only for the conventional COS, that is, for spacecrafts for which the only degree of freedom is the roll axis. Notably, the 3-axis agility of the agile spacecrafts creates a much larger search space [10] as it creates multiple viewing opportunities for each single pass over a target.

A spacecraft is defined by a provided ephemeris or trajectory model, a target body, and encompasses a state timeline for each of the sub-components such as batteries or solid state recorders. Our notional spacecraft has one or several body-mounted instruments with specific *swaths* modeled as pushbroom sensors. Geometry of the sensors are parameterized by minimum and maximum look angles as angles rotated about the velocity vector of the spacecraft from the nadir look vector, looking 90 degrees off of velocity. Here are some important concepts that we will use in the remainder of this paper:

- *visibility*: rectangular projection of the space of possible swaths on the ground between two instants.
- *observation*: the resulting data from when an instrument is turned on for a period of time at a specific roll angle. The covered targets on the ground will be those which are visible in the swath of the instrument during the period of time.
- *instrument mode*: an instrument may have several modes of operation called *instrument modes*, each constrained by the state of the spacecraft and dictating a data-generation rate on the spacecrafts storage system. An instrument mode may subsume another one depending on the type of instrument.

The observations stored on the onboard memory must then be downloaded to ground stations to be finally be delivered to end users. Projected communications windows between ground stations and spacecrafts are given as input data. When the spacecraft is communicating with a ground station, it downloads as much observations as possible, following a predefined queueing policy. Scheduling must ensure that it is never oversubscribing the memory as it would result in losing observations. The spacecrafts are equipped with batteries and possibly of solar panels for producing power. As instruments are consuming power, it is paramount that the schedule ensures the energy level remains above a certain level, which would otherwise put the spacecraft into a critical safe mode resulting in operational delays.

The problem consists in choosing the orientation and on/off times of a pushbroom instrument or collection of pushbroom instruments possibly distributed among several spacecrafts such that the schedule covers as many target points as possible, but without oversubscribing memory and energy.

From a scheduling point-of-view, single spacecraft problem belongs to single-machine scheduling problem with

sequence-dependent setup effects, job-assembly characteristics, and time window constraints. In our setting, the scheduling problem is clearly oversubscribed, as scientists always want to observe more than the available spacecraft capacity and lifetime, which mean that in addition to the *scheduling* problem (ordering a set of activities satisfying the constraints), there is a *selection* problem (selecting a subset of the set of activities). Existing works

3. DISCRETIZATION AND TASK CLUSTERING

To solve the problem, we first discretize it. We operate a four-step discretization phase during which we: (1) discretize target space by projecting a regular spacing of gridpoints on the target body (this will allow to compute whether an observation covers what portion of a target region of interest); (2) project the spacecraft swaths onto the celestial body for the whole planning horizon and discretize time; (3) intersect the swath, the target polygons and the gridpoints to discard uncoverable target points; and (4) proceed to task clustering (as defined by [17]) to produce a choice of observations to choose from.

3.1. Discretization of target space

First, a regular spacing of gridpoints is projected on the target body. The resolution of this grid is expressed in terms of number of points at equator. Three thousand points at equator will then define a distance of 13.36 kilometers between points.

Then the regions of interest could be either (1) directly intersected with all spacecraft swaths generating a collection of shards or target visibilities over time or (2) projected on a regular spacing of gridpoints on the target body [9]. The former solution gives coverage calculation at resolution limited only by processor memory and floating-point accuracy, however for scenarios of sufficient planning horizon and/or where the swath may intersect with itself, it becomes computationally expensive and the latter solution necessary. In this paper, we consider only the second case. Fig. 2 shows how a polygon is transformed in a set of point targets. The output of this process is given by the set of latitude-longitude targets. Choosing the appropriate grid resolution should take into account the smallest targets of a given scenario as a wrong parameterization might make these targets to be missed.

3.2. Projection of the swaths and discretization of time

For each spacecraft, the swath of each instrument mode (as each instrument mode may have a different swath)

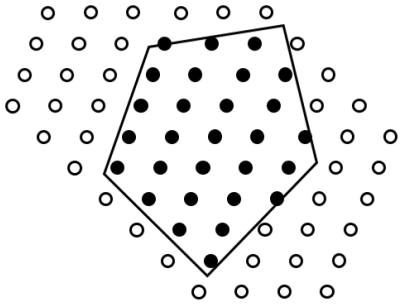

Figure 2: Intersection of a polygon with a regular spacing of gridpoints on a target body.

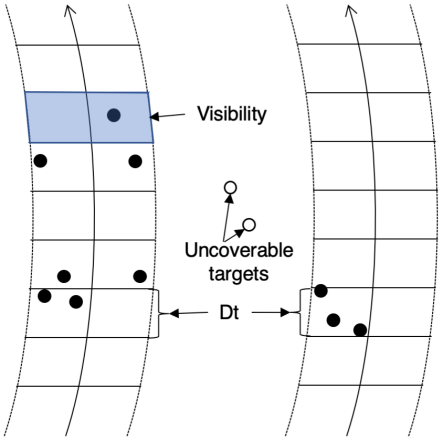

Figure 3: Illustration of the discretization steps. Projection of the spacecraft swath on a geographical region over the whole planning horizon and breakdown into visibilities of fixed timestep and elimination of uncoverable targets.

is projected onto the celestial body for the whole planning horizon at minimum and maximum roll angles. The space between these two extreme roll angles is known to be coverable.

Then these swaths are divided into steps of a parametrized duration (usually 4 or 10 seconds). We define a *visibility* as the polygon resulting from this projection of the swaths of the instruments on the considered target body for all possible rolling angle during a step. This process is shown on Fig. 3 for one instrument mode and two passages (two orbits) over a defined geographical region.

Note that a visibility contain all the possible roll angles for the instrument for a timestep. Once this is done, it is easy to compute which targets are contained in each of these visibility windows. All targets not contained in any visibility window are marked as *uncoverable* and discarded.

3.3. Task clustering

Task clustering is the procedure that will generate a set of *observations* for each spacecraft from the set of *visibilities* previously produced, restricting again the search space.

First, we restrict start and end times of observations to start and end times of visibility windows. The only remaining decision variable here is the roll angle of the observation. For each visibility. we will generate several observations with several rolling angles. Before describing how we generate these roll angles, we will describe how we handle the roll angle sequence constraint.

Angle least-commitment Slewing from one roll angle to another takes some time. Also, we have restricted start and end times of observations to start and end times of visibilities. This avoids any search for target clustering in the along-track axis but it presents an inconvenience: because each observation has a given roll angle, any two consecutive observations without the exact same roll angle are not compatible. In other terms, any roll transition is taking at least one timestep to be completed. But the roll angle is only constrained by the targets the spacecraft is trying to observe. The swath of the instrument might be larger than the geographical area we are trying to observe. In other terms, observing a given set of target might be done with not only one roll angle but an interval of roll angles.

That is why a *least-commitment* strategy is used so that roll angles of observations are not definitely set before the end of the search process but *refined*. When an observation is inserted in the schedule, the intervals of the neighboring observations are propagated with regard to slewing transition times. If an the interval of an already-inserted observation becomes empty, it means that both observations are incompatible. Now we can describe target clustering methods.

Clustering The objective of this step is to produce a set of observations for a given visibility. For each visibility, the search space is the interval of roll angles. Note that for each visibility that can cover at least one target, several observations may be generated, but they are incompatible by definition, at most one can be inserted in the schedule during scheduling. Two observationsover the same visibility are different if they do not cover exactly the same set of targets. It is theoretically possible to explore all possible observations for a given swath, but for the probable large number of visibilities and targets, it is computationally hard. The goal is then to accurately *sample* the space of roll angles. We use two methods:

Target-centric clustering One observation is generated for each target in the visibility. The central roll angle of the observation is set at the target angle in the visibility. An observation generated with this method might physically cover more than one target, but the flexibility interval is set to include only the considered target and thus remain very flexible while ensuring that all coverable target have at least an observation.

K-roll clustering The interval of roll angle is divided in k parts with k for a given instrument mode to cover at least the whole visibility space. For each part, an observation is generated. The highest k is, the more overlaps between observations there is. Compared to the target-centric approach, this approach generates more covering possibilities. On Fig. 4, while the two approaches both generate three observations, only the k-roll clustering produces an observation covering t_2 and t_3. But the resulting flexibility intervals of these observations are smaller, thus restricting compatibility with observations after and before.

All algorithms will use k-roll clustering if not mentioned otherwise.

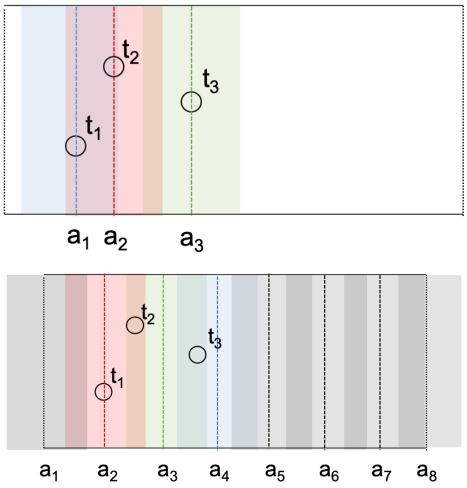

Figure 4: Comparison between target-centric and k-roll clustering for 1 visibility covering 3 targets. Colored observations cover at least one target.

4. ORDERING APPROACHES

In the previous section, it has been shown how a set of observations can be built from the problem data by discretizing time and roll angles. Solving the problem consists now in selecting a subset of observations maximizing the optimization criterion while satisfying the constraints. In this sections, we present several greedy algorithms to achieve that.

Physical constraints concerning resources such as slewing, energy or memory are always enforced during search. All the algorithms use a timeline-based modeling

framework, an approach previously explored in the space domain [2] [3] [16], in which state and resource values are represented by a fully ordered sequence of values. A resource timeline is created for each resource and every time an activity is consuming a resource, the timeline is incrementally updated and propagated. When needed, one can check whether it is possible or not by inspecting the potential effect on all timelines. Actually, none of the algorithms we present are never allowing a constraint to be violated, the current schedule always satisfy the constraints.

4.1. Greedy-in-time scheduling algorithm (PRIOTIME)

The most straightforward algorithm (our baseline) goes through all targets ordered by campaign priority and for each target, tries to schedule it as much as possible (until the required number of occurrences is attained). For each occurrence, the algorithm looks at all potential observations ordered by start times and schedule the first one. Because the algorithm goes through targets by campaign priorities, it enforces strict priority hierarchy: a given priority target will always be favored against any set of lower-priority targets. However, it is myopic in terms of coverage or rolling sequencing as it selects observations only by earliest start times.

4.2. Greedy-in-coverage scheduling algorithm (COVERAGE and K-COVERAGE)

We have seen that the baseline algorithm does not really consider coverage, that is the number of targets covered by observations, and remains myopic in this sense. We introduce a new greedy algorithm that computes a coverage quality heuristic for each observation, aggregating the value of all targets covered by it with the roll angle necessary to observe them, and schedules the best first.

Compared to the previous algorithm, this algorithm changes the decision-making hierarchy as it reasons on observations and not only targets. It reasons on the roll angle resource and tries to favor high-coverage observations that will not prevent subsequent or previous observations to be inserted in the schedule.

There are two variants for this algorithm, one using the k-roll clustering and one using the target-centric clustering.

4.3. Longest strip greedy algorithm (STRIP)

The previous algorithm tries to minimize the consumption of the roll angle resource by observations at individual level. But it does not reason over sequences of observations. As slewing is one of the main limiting factor when scheduling, a natural way of thinking about this is to build strips of consecutive observations that would cover large areas without any variation in slewing. A strip is defined as a sequence of observations without gap for a given spacecraft. The heuristic value of a strip s is computed as the sum of the heuristic values of its observations. Then the algorithm schedules strips by decreasing order of heuristic score. Note that compared to the previous approach, this method is only adding to the search space of observations as it also considers sequence of length 1.

4.4. Greedy maximum-contention algorithm (CONTENTION)

In a highly constrained scheduling problem, a common approach is to analyze conflicts between tasks and to schedule most constrained task first, those which have the least opportunities to be satisfied during the planning horizon. In constraint programming, this is called the Minimum Remaining Values heuristic (MRV) [7] and is used when choosing the order in which the decision variables will be assigned. The decision value with the smallest remaining domain is chosen by this heuristic.

Targets are to be covered by observations. As the number of spacecrafts orbits are limited, there is a limited number of observation opportunities to cover a given target. But measuring the pure opportunities is not enough. There are targets that are very isolated geographically from other targets, which will be relatively easy to schedule even for a small number of opportunities. On the contrary there are areas packed with targets, leading to concurrency between observations for all these targets. Even a target with several opportunities will be hard to schedule in this case. This is why it is necessary to combine these aspects in a measure as it has been already highlighted in [5].

This algorithms greedily schedules observations with the highest contention.

4.5. Chronological slewing optimization algorithm (CHRONO)

When trying to optimize instrument slewing, the approach in section 4.2 computes a local metric penalizing observations slewing far from the swath center of the instrument. The approach in section 4.3 is another extreme case by removing any slewing from long sequences of observations.

It has already been shown that the slewing optimization problem can be approximated to finding the longest path in a Directed Acyclic Graph (DAG) where nodes are made of observations at different timesteps and arcs represents the physical possibility for the spacecraft to go from one slewing angle to another [6]. But this approach considers that observations have fixed roll angles. As seen in Section 3.3, in our case, roll angles of observations are set as late as possible to maximize compatibility between neighboring observations. Each observation has

an interval of roll angles. In the original DAG, an arc represents the compatibility between two roll angles which is easy to compute with the roll rate. And more importantly it is a transitive relation between observations.

The number of paths is lower-bounded by n^t, with n the number of observations at each timesteps and t the number of timestep, thus making this approach intractable even for modest depths.

However, such an approximation can be used in a chronological approach as a *lookahead* heuristic to make a more informed choice, taking into account few subsequent observations that can be inserted in a near future. In a chronological algorithm , progression goes from the beginning to the end of the horizon and selects observations to insert at each timestep. The first p observation of the best n-sequence at a given instant are scheduled, with $n = 4$ and $p = 2$ in our case.

5. EMPIRICAL EVALUATION

In this section, we evaluate the performance of each proposed algorithms on a realistic scenario.

5.1. Parameters

The following parameters are set for every run. Timestep (see Section 3.1) is set at 10 seconds. The scheduling horizon duration is set to 12 days as it is the revisit period of the spacecraft we consider. The grid (see Section 3.1) is set to be of 3000 points at equator which makes a total of 317417 gridpoints for Earth.

5.2. Spacecrafts

There are two different constellations based on a spacecraft with characterics close to the NASA-ISRO Synthetic Aperture Radar (NISAR) mission spacecraft [14] which has a heliosynchronous 98° inclined Earth orbit and a 12-day repeat period. We consider that it only has 1 instrument with 9 independant modes (no mode subsumes another mode). Its orbit and instrument size results in a 240-km swath with an angular visibility of 12° (30-42°). The two studied constellations are constellation `ONE`, which is made of one spacecraft, and constellation `SIX`, which is made of six identical spacecrafts whose orbits are offset by two days each. In other words, if spacecraft 1 nadir is flying over a point on earth at the beginning of the scheduling horizon, spacecraft 2 will fly over the same point exactly two days after with the same orbital position, spacecraft 3 four days after, and so on.

The number of resulting visibilities for each constellation is 103681 and 622080. Onboard memory is limited, data production is associated with each observation and no downlink is possible during the horizon which makes the resource clearly oversubscribed and a limiting factor. As rolling is one of the main limiting factor, we use two roll rates to evaluate the impact on schedule quality: rate `LOW` which is 0.2 degrees per seconds, which makes the maximum roll transition to be 6 timesteps, 1 minute, and rate `HIGH` which is 12 degrees per second, which limits rolling to one timestep in this case as the whole angular range is covered in one timestep

5.3. Campaigns considered

We evaluate algorithms against one simple scenario (more are shown in the upcoming journal version of this paper and show the same tendencies in results), the `Landmass` scenario, which is taken from the studies to address the *Surface Deformation and Change* part of the 2017 Decadal Survey [13], and simplified to suit our need. This scenario has only one priority, and specifies that the whole Earth landmass should be observed. It contains 88226 targets.

5.4. Chosen criterions

For evaluating the algorithms, we use a quality criterion aggregating coverage and campaign priority, whose value is between 0 and 1. To try explaining the results, we also report for each scenario, configuration and algorithm: the runtime decomposed in pre-ordering runtime and ordering runtime, the mean roll transition angle between subsequent observations in the schedule, the number of observations inserted in the schedule, and the mean number of targets in the observations in the schedule, the *target density*.

5.5. Comparison of the algorithms

CLASP is developed in C++. Experiments were ran on a Apple MacBook Pro equipped with an Intel Core i9 processor at 2.9GHz and 32GB of DDR4 ram.

When observing the quality of produced schedule on Fig. 5, we see that the roll rate has, as expected, an impact on the performance. As the spacecraft loses timesteps to roll from observation to observation and that taking observation during rolling is not permitted in this setting, it can be seen on Fig. 6 that schedules for configurations with a low roll rate indeed produce less observations than schedules for configurations with a high roll rate, regardless of the ordering algorithm. This result is also unaffected by the target density.

In all configurations, the best quality is achieved with the `CHRONO` approach which optimizes rolling path separately for each spacecraft. It combines a high target density (see Fig. 6) with a low mean transition roll angle (see Fig. 7) but at a computational cost more than 4 times higher than

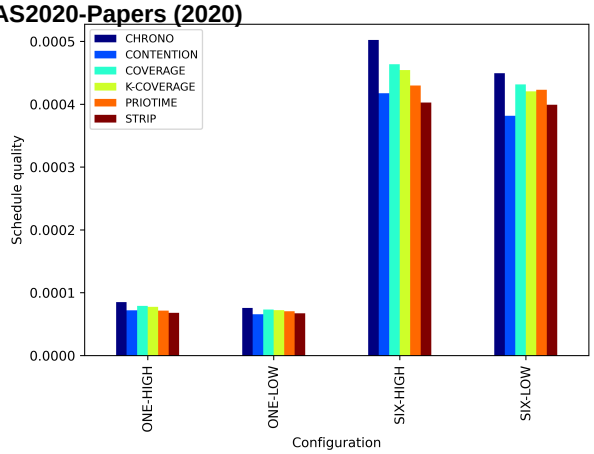

Figure 5: Schedule quality by configuration and algorithm for scenario Landmass

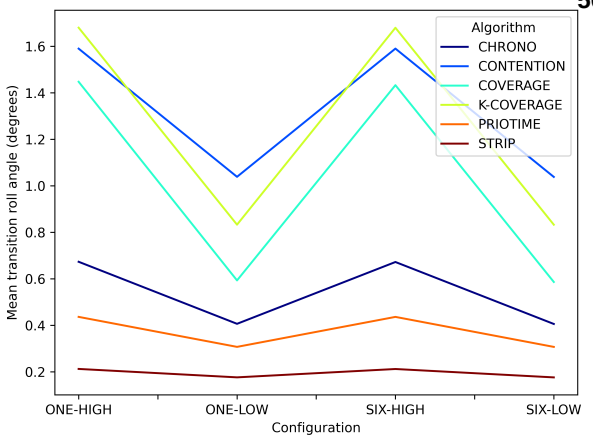

Figure 7: Mean roll transition per configuration and algorithm for scenario Landmass

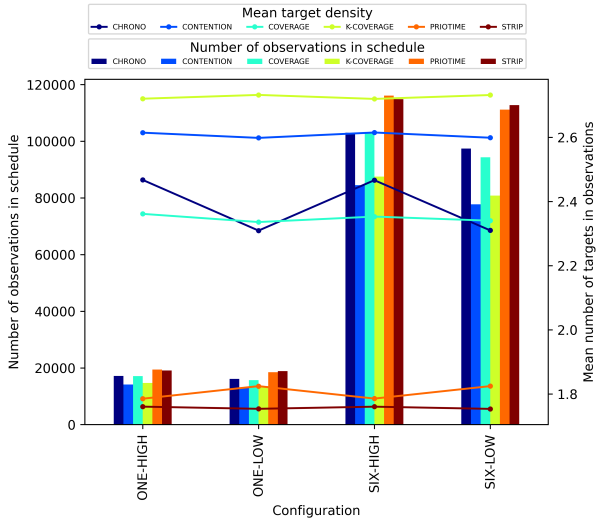

Figure 6: Number of observations in the schedule and mean target density per observation by configuration and algorithm for scenario Landmass

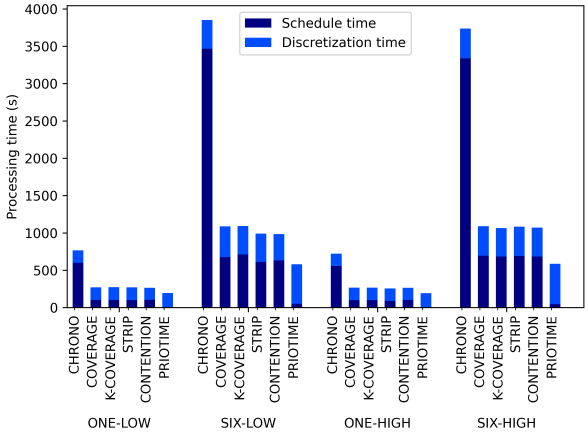

Figure 8: Processing time per configuration and algorithm for scenario Landmass

other approaches. As expected, developing the rolling paths at each timestep is extremely time and memory-consuming(see Fig. 8).

Surprisingly, the `COVERAGE` algorithms performs better than the `K-COVERAGE` approach. As expected the latter has a higher target density as it finds the observations containing the most targets because of its systematic sampling of the roll angle space. But, at the same time, it is able to insert less observations in the schedule, probably because of longer roll transition. By aggregating more targets in the observations, this algorithms reduces the roll flexibility (see Section 3.3) and thus prevents from accommodating with other future insertions.

The `PRIOTIME` algorithm performs better than `STRIP` and `CONTENTION` while being the fastest approach. It is able to insert a large number of observations in the schedules. It can be noted that its mean roll transition is low, probably because its target-centric clustering allows for a greater angle flexibility. `CONTENTION` bad performance can be linked to the fact it does not consider rolling at all. The `STRIP` approach is the least-performing approach with this scenario. As expected, it has the lower mean transition roll angle of all approaches which lead to having second highest number of observations in the schedules, but with the least target density. This shows how maintaining a balance between all these parameters is important.

6. CONCLUSION

In this paper, we have presented a singular spacecraft mapping problem involving observational constraints arising in a variety of missions. A general multi-step discretization approach has been described to deal with the geometric constraints and possibly long planning horizon. Several ordering algorithms have been considered

and experiments have shown that in order to produce high-quality schedules, straigthforward approaches optimizing only one aspect of the problem, such as priority or rolling, are not sufficient and require algorithms combining reasoning on coverage, priority and rolling aspects. This approach can improve the quality of resulting schedules, and thus the overall science return achieved in such missions.

7. FUNDING

This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration (80NM0018D0004).

REFERENCES

[1] Charles H. Acton. Ancillary data services of nasa's navigation and ancillary information facility. *Planetary and Space Science*, 44(1):65 – 70, 1996. Planetary data system.

[2] Amedeo Cesta, Alberto Finzi, Simone Fratini, Andrea Orlandini, and Enrico Tronci. Validation and verification issues in a timeline-based planning system. *The Knowledge Engineering Review*, 25(3):299–318, 2010.

[3] Steve Chien, Daniel Tran, Gregg Rabideau, Steve Schaffer, Daniel Mandl, and Stuart Frye. Timeline-based space operations scheduling with external constraints. In *Twentieth International Conference on Automated Planning and Scheduling*, 2010.

[4] Joshua R. Doubleday. Three petabytes or bust: Planning science observations for nisar. In *SPIE 9881*, New Delhi, India, May 2016.

[5] Jeremy Frank, Ari Jonsson, Robert Morris, David E Smith, and Peter Norvig. Planning and scheduling for fleets of earth observing satellites. 2001.

[6] Virginie Gabrel and Daniel Vanderpooten. Enumeration and interactive selection of efficient paths in a multiple criteria graph for scheduling an earth observing satellite. *European Journal of Operational Research*, 139(3):533 – 542, 2002.

[7] Robert M. Haralick and Gordon L. Elliott. Increasing tree search efficiency for constraint satisfaction problems. *Artificial Intelligence*, 14(3):263 – 313, 1980.

[8] Russell Knight and Steven Hu. Compressed large-scale activity scheduling and planning (clasp) applied to desdyni. In *Proceedings of the Sixth International Workshop in Planning and Scheduling for Space (IWPSS 2009)*, Pasadena, CA, 2009.

[9] Russell Lee Knight. *Solving the constrained coverage problem with flow networks as linear program approximations*. University of California, Los Angeles, 2005.

[10] Michel Lemaitre, Gérard Verfaillie, Frank Jouhaud, Jean-Michel Lachiver, and Nicolas Bataille. Selecting and scheduling observations of agile satellites. *Aerospace Science and Technology*, 6(5):367 – 381, 2002.

[11] D. McLaren, G. Rabideau, S. Chien, R. Knight, S. Anwar, G. Mehall, and P. Christensen. Scheduling results for the themis observation scheduling tool. In *International Workshop on Planning and Scheduling for Space (IWPSS 2011)*, Darmstadt, Germany, June 2011.

[12] A. Moy, A. Yelamanchili, S. Chien, A. Eldering, and R. Pavlick. Automated scheduling for the oco-3 mission. In *International Workshop for Planning and Scheduling for Space (IWPSS 2019)*, pages 195–203, Berkeley, California, USA, July 2019.

[13] National Academies of Sciences, Engineering, and Medicine. *Thriving on Our Changing Planet: A Decadal Strategy for Earth Observation from Space*. The National Academies Press, Washington, DC, 2018.

[14] P. A. Rosen, Y. Kim, R. Kumar, T. Misra, R. Bhan, and V. R. Sagi. Global persistent sar sampling with the nasa-isro sar (nisar) mission. In *2017 IEEE Radar Conference (RadarConf)*, pages 0410–0414, 2017.

[15] Martina Troesch, Steve Chien, and Eric Ferguson. Using automated scheduling to assess coverage for europa clipper and jupiter icy moons explorer. In *International Workshop on Planning and Scheduling for Space (IWPSS 2017)*, Pittsburgh, PA, June 2017.

[16] Gérard Verfaillie, Cédric Pralet, and Michel Lemaître. How to model planning and scheduling problems using constraint networks on timelines. *The Knowledge Engineering Review*, 25(3):319–336, 2010.

[17] Guohua Wu, Jin Liu, Manhao Ma, and Dishan Qiu. A two-phase scheduling method with the consideration of task clustering for earth observing satellites. *Computers and Operations Research*, 40(7):1884 – 1894, 2013.

[18] A. Yelamanchili, S. Chien, A. Moy, E. Shao, M. Trowbridge, K. Cawse-Nicholson, J. Padams, and D. Freeborn. Automated science scheduling for the ecostress mission. In *International Workshop for Planning and Scheduling for Space (IWPSS 2019)*, pages 204–211, Berkeley, California, USA, July 2019. Also appears at ICAPS SPARK 2019.