Deconfliction of Spacecraft Communication Schedules with Preference Optimization

John L. Bresina[1], Matthew D'Ortenzio[1], Paul H. Morris[1], K. Brent Venable[2,3]

[1]NASA Ames Research Center, Moffett Field, CA, USA; [2]Tulane University, New Orleans, LA, USA; [3]IHMC, Pensacola, USA
John.L.Bresina@nasa.gov, matthew.dortenzio@nasa.gov, Paul.H.Morris@nasa.gov, kvenabl@tulane.edu

Abstract

In this paper, we describe Jigsaw: a suite of AI software tools for deep-space nano-satellite ("NanoSat") communication scheduling. Jigsaw is intended to support the 13 NanoSat missions set to launch on NASA's Space Launch System (SLS) / Orion Exploration Mission 1 (EM-1) in 2019.

Rather than have each of these missions independently submit requests to the Deep Space Network (DSN) process and negotiate with all other competing missions, Jigsaw can be used to generate a joint request that has already eliminated conflict among the 13 missions. This can greatly reduce the cost of scheduling and negotiating for each mission. This is especially useful during the first week of the EM-1 mission when all the spacecraft have critical events and are all competing for the same DSN assets.

In order to automate this deconfliction process, the communication pass requests from the individual missions include both the minimal requirements and the ideal preferences, and each pass is assigned a priority. The constraints and preferences specified in the requests are used to guide the generation of a joint schedule that satisfies all constraints, maximizes the achievement of the preferences, and is fair across the missions.

We describe the specification of the communication requests, which contain both hard constraints and preferences, and then we describe our current approach to solving this scheduling problem. The approach is exemplified on a realistic example involving three of the EM-1 NanoSat missions.

Introduction

Missions involving small spacecraft are attracting an increasing amount of attention by the space research community due to the many advantages they bring to the table. Small spacecraft provide a means of performing scientific observations, as well as, demonstrating new technologies, with limited risk and cost.

The first cadre of 13 *interplanetary* NanoSat missions, is set to launch on NASA's Space Launch System (SLS) / Orion Exploration Mission 1 (EM-1) in 2019. Many of the missions will need to rely on the powerful antennas of NASA's Deep Space Network (DSN) to provide communication between their NanoSat and ground controllers. Each of these stand-alone missions will be faced with the prospect of a lengthy and costly communications scheduling negotiation for their early mission critical event coverage. This will be a process that the missions likely haven't allocated sufficient budget for, yet one that they will be required to participate in, such that some of their most critical events (e.g. trajectory correction maneuvers) will be successful.

The DSN's existing scheduling process and software is highly successful at balancing the needs of 30+ missions when those missions are distributed throughout the solar system and when data return needs can be balanced over long periods of time. However, their processes are not optimized to support the situation that EM-1 will present, where 13 NanoSats will be in the same area of the sky, continually competing for the same assets, and where they rely on the missions themselves to negotiate time collaboratively. Similar negotiations on a pair of co-manifested Moon-bound missions in 2009 took 2 months, and only yielded about 15% contingency case coverage. While the DSN is planning some system improvements to support NanoSat missions, intense over-subscription issues are likely to persist.

The Jigsaw software effort has started to address this problem by adapting artificial intelligence scheduling and preference optimization techniques. The goal is to allow the DSN and its NanoSat mission customers to efficiently and accurately construct early mission communication schedules, when contention for ground antennas will be highest. If successful, employing Jigsaw to help manage early-mission EM-1 critical events will drastically reduce the amount of time (potentially up to 100 times), and hence funds, that it will take to create a viable early-mission DSN schedule for the EM-1 launch.

Rather than have each of these missions independently submit requests to the DSN process and negotiate with all other competing missions, Jigsaw can be used to generate a joint request that has already eliminated conflict among the 13 missions. In order to automate this deconfliction process, the communication pass requests from the individual missions include both the minimal requirements

and ideal preferences, and each pass is assigned a priority. The constraints and preferences specified in the requests are used to guide the generation of a joint schedule that includes as many requests as possible, does not violate any hard constraints, maximizes the achievement of the preferences on pass durations and start times, and is fair across the missions in terms of resource allocation.

Our current approach involves two phases. In the first phase, a schedule is generated that tries to satisfy the minimal requirements specified in the requests from all the missions. In the second phase, this minimal-requirements schedule is improved as much as possible, based on the ideal preferences expressed in the requests. In both phases, the schedules respect the request priorities and are fair across all missions. In this paper, we first present background information on the EM-1 mission, on the DSN's structure, and its scheduling process. Secondly, we describe our specification language and process for communication requests. We then describe our two-phase approach to solving this scheduling problem and for the second phase, we present two alternative techniques for optimizing the preferences. We illustrate Jigsaw's operation with a realistic set of requests based on three of the EM-1 NanoSat missions.

The EM-1 Mission

In late 2019 NASA will launch its new heavy-lift rocket, the Space Launch System (SLS), carrying the Orion crew capsule. This being the first test flight of the SLS/Orion beyond low earth orbit, NASA has designated the mission "Exploration Mission 1 (EM-1)". While the crew capsule will be unmanned on EM-1, the rocket will be carrying along 13 NanoSats. These NanoSats, each only about the size of a shoe-box, are entire missions onto themselves, and will be deployed from the upper stage of the SLS at various points along the rocket's trajectory between the Earth and the Moon (see Figure 1).

These 13 missions, some NASA sponsored, others Industry or International sponsored, are groundbreaking. While academia and the aerospace industry have been developing and deploying NanoSats of increasing capability over the last decade, these missions represent the first cadre of interplanetary NanoSat missions, tasked with exploring regimes outside of the Low Earth Orbit (LEO). The Moon, an asteroid and interplanetary radiation are some of the research targets for these missions.

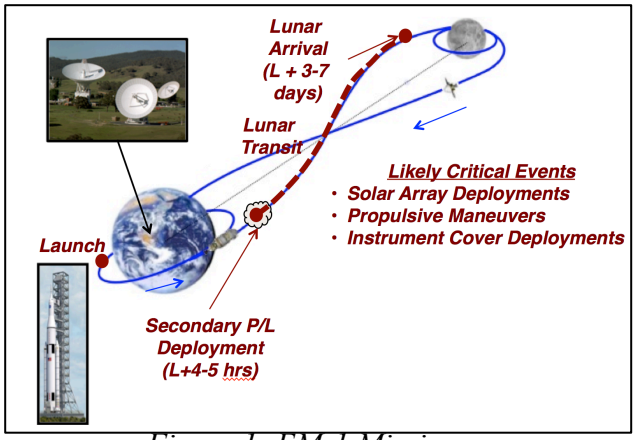

Figure 1: EM-1 Mission.

Of course, with any new paradigm, comes a new set of challenges. In the case of the EM-1 NanoSats, scheduling of ground-based antenna dishes and communications equipment, is one of them. The only dishes large enough to support talking with these tiny spacecraft (with correspondingly tiny radios) are contained within NASA's Deep Space Network (DSN). The DSN is already one of NASA's communications workhorses supporting all of NASA's deep space missions including New Horizons, the Mars Rovers, and Kepler. Successfully integrating up to 13 new missions for the DSN to support in the matter of a day, presents a unique resource scheduling challenge.

Deep Space Network Scheduling

The DSN comprises a collection of large diameter antenna dishes or "stations", and radio frequency processing equipment at three geometrically distributed locations or "complexes". The DSN complexes, in Goldstone, CA (USA), Madrid (Spain) and Canberra (Australia) are located approximately 120 degrees of longitude apart, such that nearly any deep space location (e.g. Mars, Pluto) is above the local horizon, and hence can be "viewed" by an antenna of at least one of the complexes at any time of the day. Each complex has between three and five 34-m diameter antenna dishes, as well as a single 70-m dish.

Each of the stations can support a variety of "services" according to the needs of the mission it is supporting; the most common services include:

- Telemetry Downlink (data communication from the spacecraft to the ground antenna),
- Command Uplink (data communication from the ground antenna to the spacecraft),
- Doppler tracking (measurement of frequency shifts in the downlink signal, from which relative velocity between the spacecraft and the ground antenna can be derived), and
- Range tracking (time of flight measurement of a signal sent up by the ground antenna, received and retransmitted by the spacecraft, from which distance between the spacecraft and the ground antenna can be derived).

Each DSN mission customer is unique with respect to which combinations of services are required, and with respect to when and for how long the services are required, with those needs usually being dependent on the phase of the mission. For example a mission attempting to enter the orbit of another planet, might require Doppler and range tracking several hours before the time of the planetary encounter, then uplink, downlink and tracking

through the time of orbit capture and for an hour afterwards. Later in the mission however, the requirement might relax to just two hours of concurrent uplink and downlink time per week, such that all of the data collected on-board during that week can be transmitted to the ground. Frequently, antenna size and/or capability are also factors in how a mission uses the network (e.g. spacecraft at or beyond the outer planets require use of the 70-m station due to its higher receive and transmit gain).

Production of a network-wide conflict-free schedule is the responsibility of the entire DSN mission set. That is, the scheduling of assets is community-driven rather than priority driven. While the established processes by which the community arrives at the network schedule are well outside the scope of this paper, some key terminology and concepts are useful to understanding the Jigsaw context. In the end, DSN stations are allocated to missions for blocks of time, referred to as "tracks". Each track is created with a combination of the services described above. The minimum track time is usually on the order of 30 minutes, maximum track lengths are sometimes as long as 10 or 11 hours, corresponding to horizon-to-horizon tracking of the spacecraft by a station (think of the Moon "rising", then "setting" about 10 hours later).

In addition to tracking time, the schedule also includes time between tracks on individual stations for the station operators to transition from supporting one mission to supporting another. A variable-length "set up" is always scheduled prior to the start of the track, and a 15-minute "tear down" scheduled after the end. The duration of the "set-up" is dependent on which services are selected for the given track; minimum of 30 minutes, maximum of 75 minutes.

Jigsaw Approach

In this section, we first describe how the mission teams specify the communication requests and how they express their preferences regarding the durations and start times of the requested passes. Secondly, we describe the first phase of our approach, which involves generating a schedule that attempts to satisfy the minimum requirements implied by the set of requests. The second phase involves optimizing this minimum requirement schedule in terms of the preferences. We present two different approaches to this preference optimization process; the first alternative only tries to optimize the duration preference, and the second alternative attempts to optimize both duration and start time preferences. These approaches are illustrated on an example problem based on the BioSentinel mission, the Lunar Flashlight mission, and the Near-Earth Asteroid (NEA) Scout mission.

Communication requests with preferences

We use the Open Source Scheduling and Planning InterFace for exploration (OpenSPIFe) (for details see https://github.com/nasa/OpenSPIFe/wiki), an open source software managed by NASA Ames Research Center, to create communication pass requests, which are represented as activities in a timeline. The important events of each NanoSat mission and the EM-1 mission are also represented on timelines to give context. In addition, the SPIFe resource displays show the following state resources to help the mission teams place their requests:

- For each DSN station (antenna), when has the station been allocated for use by the NanoSat missions; determined by the DSN.
- For each DSN complex and NanoSat mission pair, when is the complex in view of the NanoSat for uplink communication; determined by the predicted mission trajectory.
- For each DSN complex and NanoSat mission pair, when is the complex in view of the NanoSat for downlink communication; determined by the predicted mission trajectory.

The Activity Dictionary includes an activity type to represent each mission's pass requests (e.g., BIOS_PASS_REQUEST for requests from the BioSentinel mission), and each of these request activity types include the following parameters:

- Ideal and Minimum pass duration
- Pass Type (Uplink-Downlink with/without ranging, or Downlink only)
- Antenna Type (Band Wave Guide (BWG), High Efficiency (HEF), Any)
- DSN Complex (Canberra, Goldstone, Madrid, Any)
- Event reference or date reference
- Largest, ideal, and smallest *factor before*
- DSN criticality
- Mission priority
- Whether to add a backup pass
- Backup tolerance factor.

The earliest, the ideal, and the latest that the requested pass can start are computed as offsets from the reference date. The date reference is an absolute date; thus, it specifies the reference date explicitly. The event reference specifies one of the mission's critical events (e.g., deployment from EM-1); the derived reference date, in this case, is the start time of the specified event. More specifically, the offsets from the reference date are computed as the product of the pass duration and the appropriate "factor before" parameter: earliest start time uses the largest factor before, ideal start time uses the ideal factor before, and the latest start time uses the smallest start time before.

The offset is subtracted from the reference date (either the absolute date or an event's start time). Note that these three factors can be positive or negative; if positive, the

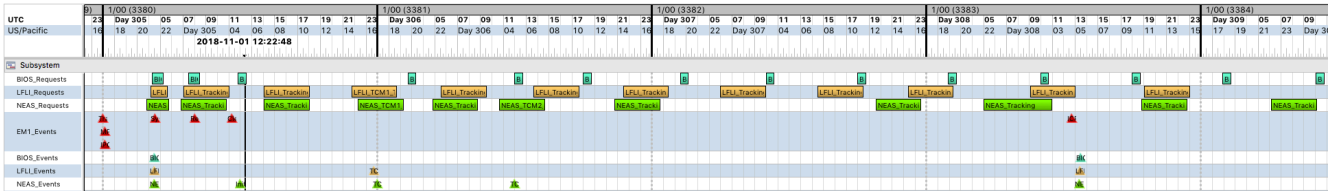

Figure 2: Pass requests and mission events in SPIFe.

computed start time constraint (earliest and latest) or preference (ideal) is before the reference date, and if negative, they will be after the reference date.

As an example, suppose the reference date is noon on a particular date, and the request parameters are as follows:

- Largest factor before = 1.0
- Ideal factor before = 0.50
- Smallest factor before = -0.20
- Minimum pass duration = 1 hour
- Ideal pass duration = 2 hours

If we consider the minimum pass duration, then the pass is constrained to start within the interval [11:00, 12:12], with an ideal start time of 11:30. Likewise, if we consider the ideal duration, then the pass is constrained to start within [10:00, 12:24], with an ideal start time of 11:00.

The backup tolerance factor determines how far the backup pass can be shifted (right or left) from the associated primary pass. The start time offset and end time offset is computed as the product of tolerance factor and the pass duration.

Note that in this problem domain, the start time preference and the pass duration preference are not independent, and this dependence between preferences is a unique challenge in the work that's been done on planning and scheduling with preferences.

Example problem

The example problem that we have been using to demonstrate feasibility of our approach uses representative schedule requests from three of the EM-1 NanoSat missions: BioSentinel (a deep-space radiation biology experiment), Lunar Flashlight (a lunar science mission), and NEA Scout (a near-Earth asteroid mission). Figure 2 shows the full set of requests from these missions, in the context of their key mission events (e.g. trajectory correction maneuvers) and overall EM-1 mission events (e.g. launch time). The requests were based on preliminary mission operations plans for each mission. For example BioSentinel doesn't have any propulsive maneuvers, so its request is for three short passes per day for state of health checking. On the other hand NEAScout and Lunar Flashlight do have maneuvers, so their requests are more substantive, with longer passes both before and after the maneuvers to gather tracking data.

Figure 2 shows all of the requests in the Example Problem, with each mission on its own row in the timeline. The pass duration and start times shown are the ideal values, but preference information for each pass, which includes a minimum duration and earliest or latest start time were also captured.

As an example of preference specifications, we consider a request for the NEA Scout mission, whose purpose is to observe a maneuver. In this case, the minimum and ideal durations are 2.5 and 4 hours, and the largest, ideal, and smallest factors are 0.7, 0.5, and 0.2. This means, assuming a duration of 3 hours, that ideally the event will occur in the middle of the pass and the pass will not be scheduled to start earlier than 126 minutes before the event and no later than 30 minutes before the event.

There is a total of thirty-seven requests in the Example: fourteen from BioSentinel, twelve from Lunar Flashlight, and eleven from NEA Scout, plus 3 backups.

Generating minimum requirement schedules

The first phase of our current approach involves generating a schedule that attempts to satisfy the minimum requirements of all the missions' requests. The pass durations used are the minimum durations specified in the requests. No attempt is made to schedule a pass close to its ideal start time; in fact, the passes are scheduled as early as possible, in order to avoid leaving gaps in the schedule. Any backup passes are considered only after all primary passes have been attempted to be scheduled.

The scheduler uses a random sampling approach, where on each sample, a schedule is incrementally constructed via a sequence of random choices, and then it is checked for validity. If the schedule is invalid, it is discarded, if it is valid, it is compared to the current best schedule to determine whether it became the new best schedule. During the incremental construction, the sequence of choices are as follows:

1. Randomly choose a requested pass to schedule.
2. Randomly choose a possible station on which to schedule the pass.
3. Randomly choose a possible schedule window to insert the pass.
4. Randomly choose a predicted DSN setup duration to use (unless one already exists in the schedule).

In Step 2, the set of possible stations that can service a request is restricted by the request's antenna type and complex parameters. It is also restricted by what stations have been allocated by the DSN for the EM-1 missions and by the in-view windows (uplink and down view periods, depending on the request pass type parameter). Lastly, it is restricted based on the current schedule for

each station; that is, there has to be an open slot in the station schedule big enough to fit the minimal-duration pass. There could be more than one open slot in a station schedule that can accommodate the pass. In Step 3, one of these open slots is randomly chosen and the pass is scheduled at the earliest time in that slot.

If there already is a setup activity scheduled for a prior pass of the same mission and the gap between the two passes is not larger than one hour, then no additional setup is inserted. Otherwise, a setup activity is scheduled before the pass and the setup duration is randomly chosen between the duration required by the pass and the maximum setup duration of one hour (Step 4). We insert a potentially longer duration setup than is currently required because, later in the scheduling process, another pass from the same mission may be scheduled after the current pass, serviced by the same setup, and this new pass may require a longer setup time than the current pass does. The actual setup durations can only be determined in the context of the complete schedule; this poses a unique challenge to constructing the schedule in an incremental fashion.

A schedule is valid if the predicted setup durations are at least as large as the actual setups needed. The schedule's evaluation is based primarily on the priorities of the requests that did not fit into the schedule and secondarily on the fairness across the different missions with respect to the percentage of requests unscheduled. The lower the score, the better the schedule quality.

For each primary pass not in the schedule, the priority score is incremented by the exponential of 10 raised to the (1 – pass priority). Thus, the more important the pass (i.e., the lower the priority integer), the larger the penalty for not scheduling the pass. For the priority factor of the score, the backup passes that are missed are treated as if they have a priority two more (i.e., less important) than the associated prime pass does.

The fairness across prime passes and across backup passes is computed separately. The fairness scores are based on the differences, across the missions, of the proportion of requests that are not scheduled. The ideal fairness has a difference metric of zero.

The factors are weighted and summed as follows: (10000 * priority score) + (100 * prime fairness score) + backup fairness score. The number of setup activities is used as a tie breaker, where the schedule with fewest setups is preferred.

We will now describe two options for phase two of the Jigsaw approach which attempt to improve the quality of the minimum requirement schedule by optimizing the preferences.

Optimizing pass duration via binary search

This approach to optimization combines the (previously described) random sampling scheduler together with binary search to try to uniformly optimize the durations towards the ideal. It does not attempt to optimize the start times. It uses the following steps:

1. Input the minimum requirement schedule and remove from the set of requests any primary or backup passes that did not get scheduled.
2. With only the remaining accepted requests, reformulate the input using durations that are intermediate between minimal and ideal. These are chosen to support a search for improved durations that still satisfy the accepted requests.

The search algorithm iterates over the priority values p from highest to lowest and uses the following scheme to set the request durations in each iteration.

(a) Use the already computed best duration for requests of higher priority than p.
(b) Use the minimum duration for requests of lower priority than p.
(c) Do a binary search for the *uniformly best* duration of requests whose priority is p.

The binary search for the *uniformly best* duration is interpreted as seeking the highest *alpha* between 0 and 1 such that all the accepted requests can be scheduled with the pass durations set to (1 - *alpha*) * minimum duration + *alpha* * ideal duration.

Note, that the optimization is lexicographic with respect to priorities. Thus, it treats durations of higher priority requests as being infinitely more important than those of lower priority requests. An egalitarian alternative could seek the uniform best for all the requests, then seek further improvements for successively higher priorities. This would result in *alpha* values that vary monotonically with the priorities, but it is unclear whether it would better suit mission needs.

Optimizing pass duration and start time

The second optimization technique, which we denote as *Stretch-and-Shift*, takes as input the minimum requirement schedule (described previously). It then performs the following steps:

1. The minimum requirement schedule is scanned for all opportunities of increasing the duration of passes. All passes that can be stretched are binned into groups corresponding to missions and are sorted in decreasing order of mission priority. Whenever possible, tie breaks are applied using the DSN criticality.
2. Passes are extracted from the bins in a round-robin fashion until all bins are empty. For each extracted pass, its duration is stretched to the minimum between its ideal duration and the available time. The minimum increase in duration is set to 5 minutes and stretching is applied bilaterally until either one side gets saturated or the ideal duration is reached. If one side is filled, and the ideal duration is not reached, then, the stretching continues on the other side. Once the duration is optimized the

preference over the start time is computed as described in Section "Communication Requests with Preferences". The pass is then rigidly shifted in order to bring the start-time as close as possible to its ideal time-point, given the duration.

We note that this approach is based on the assumption that increasing the duration of a pass is more important that optimizing the start time. Moreover, fairness in the optimization process is ensured by the round-robin strategy applied to select candidate passes. Finally, priorities are respected by giving passes with a high value precedence in terms of optimization.

Results on example problem

We now describe the results obtained by the optimization approaches on the example problem which we have described earlier.

Both solvers were given in input the same minimum requirement solution, which scheduled 34 out of 40 requests (where 3 of the unassigned requests were backups) assigning minimum durations and earliest feasible start times to the activities.

	Priority 1	Priority 2	Priority 3	Priority 4
Binary Search	100%	0%	75%	100%
Bin. Search Egal.	75%	25%	25%	25%
Stretch-and-Shift -Duration	99%	56%	80%	77%
Stretch-and-Shift - Start	76%	58%	38%	72%

Table 1: Increase in duration and start time obtained by three optimization methods. Averaged by priority groups.

The first three rows of Tables 1 and 2 show the improvements obtained, with respect to the minimum requirement solution, in terms of duration for standard and egalitarian binary search and for the stretch-and-shift approach. The last row of both tables shows the improvement in terms of start time for the stretch-and-shift approach. All improvements are expressed as percentages. In particular, the first three rows of Table 1 and Table 2 show the percentage of increase in duration achieved by the three methods with respect to obtaining the ideal duration. The last row of the tables show how much closer, in percentage, the shifting procedure was able to move the start time of the activities to their ideal values. All percentages are averaged, in Table 1 by priority and in Table 2 by mission.

We note that the quality of the solution found by the binary search approach to optimization is affected by two preset parameters: the number of random trials used by the underlying solver; and the resolution of the *alpha* values in the binary search. The results shown here were obtained with 15,000 trials and 1/8 resolution.

Note the improvement to the ideal duration for the highest priority requests in the case of binary search had the effect of limiting the second-highest to their minimum durations. The alternative, more *egalitarian*, method described in the binary search section achieved lower overall improvements with the same preset parameters but with a fairer distribution across missions.

	NEAS	LFLI	BIOS
Binary Search	66%	69%	80%
Bin.-Search Egal.	39%	33%	29%
Stretch-and-Shift -Duration	67%	81%	90%
Stretch-and-Shift - Start	47%	49%	83%

Table 2: Increase in duration and start time obtained by three optimization procedures. Averaged by mission.

In the case of the Stretch-and-Shift algorithm we also tested the impact of changing the order of the missions in the round-robin. In this example such order turned out to be irrelevant.

We remark that the results we report only pertain to the execution of our method on an example which was designed as a realistic representation involving three missions. Our near future agenda, as discussed later, includes testing the methods on more realistic examples including all 13 missions and on synthetic instances appropriately generated to resemble scenarios relevant to the EM-1 mission.

Related Research

Optimization in the context of communication scheduling has recently been investigated, mainly in response to the current DSN oversubscription problem. The work described in (Pinover et al., 2017), references the same mission as a Jigsaw, and shows the infeasibility of an approach which tries to relegate time slots for small satellites to residual gaps left open by the other 35 missions currently handled by the DSN (opportunistic gap filling). The authors propose block scheduling, which is an approach based on collapsing several small satellites flying in sufficiently tight formation into a single entity in terms of communication scheduling. This has many benefits, among which is the simplification of the final deconfliction phases, which are still tasked to humans, and the optimization in terms of the number of setup and teardown times. This approach is indeed complementary to ours. On one side, it does not address the incorporation of mission manager's preferences in terms of duration and start times and it does not provide any guarantee in terms of fairness among missions. On the other side, it optimizes

with respect to complex global DSN constraints which we treat as strict a priori requirements.

Another particularly interesting paper (Augenstein, et al., 2016) addresses the topic of scheduling constellations of Earth-orbiting satellites using a mixed-integer linear programming (MILP) approach. While the context and constraints are different, and the nonlinear preference dependency aspect of Jigsaw does not lend itself to an MILP formulation, the paper introduces several intriguing ideas, including compiling complex requirements into simpler mutual exclusion constraints. Unfortunately, this technique seems not to be applicable in our case because of global interactions involving the setup and teardown needs of the different requests.

In (Skobelev, et al., 2014) the authors tackle a similar problem to the one considered here, with a multi-agent technique where a schedule emerges from the interaction and trade-offs of many agents which continuously change decisions to improve their objectives and the objectives of the system as a whole. This approach is quite different from ours as it requires the definition of negotiation protocols and tradable resources.

Other papers present in the literature consider variants of the communication scheduling problem. For example in (Abraham, et al., 2016) the focus in on the Multi Spacecraft Per Antenna (MSPA) case, in which a single antenna can support simultaneous communication with multiple spacecraft. Intuitively, MSPA can be modeled in our setting just by duplicating antennas and in-view periods. However, there are several additional constraints which must be imposed to ensure feasibility. We see this extension as a direction for future work.

More in general, the problem that we tackle is an instance of scheduling with temporal constraints and there is a conspicuous amount of work which has been devoted to this topic (Bartak et al., 2014). Temporal constraints (Dechter et al., 1991) are defined as intervals containing allowed durations or interleaving times between activities. They have been extended in several ways: to incorporate different forms of preferences (Khatib et al., 2007; Peintner et al., 2005), to handle uncertainty on the occurrence time of certain events (Vidal et al., 1999), to model conditional information on the execution of activities (Tsamardinos et al., 2003; Bartak et al., 2007), and to handle preferences and uncertainty simultaneously (Rossi et al., 2006). Unfortunately, the problem we have does not directly fit into any of the models present in the literature. In fact, our setting is characterized by the coexistence of priorities over the activities, fairness requirements, and preferences which are conditioned not on exogenous events, such as in (Tsamardinos et al., 2003) but on choices made the scheduler. It is however on our agenda to investigate the application some of these methods as a component of our solving procedure.

Concluding Remarks

This paper presented our preliminary work on Jigsaw, a scheduling system that can help missions like EM-1 with multiple NanoSats being deployed and competing for the same DSN assets. The approach we are taking involves two phases: (1) generating the minimum requirement schedule and (2) improving the schedule by optimizing the pass duration and start time preferences. The first phase has the unique challenge of scheduling the setup activities, given that the correct setup duration can be determined only in the context of the complete schedule. The second phase has the unique challenge of optimizing two preferences (pass duration and pass start time) which are not independent.

In the last year or so, the DSN has become keenly aware of the challenges that NanoSats will present and is subsequently performing research into future capabilities to expand their capabilities to support multiple spacecraft in parallel. However, implementation of these new capabilities is not expected to be ready to deploy for the EM-1 mission. Furthermore, these capabilities are focused on the science phases of the missions, not the first few days in which DSN resource contention is highest. As a result, JPL/DSN has expressed support for the Jigsaw project and sees it as complementary with their efforts.

Thus far, we have demonstrated the feasibility of the Jigsaw approach on a reduced problem including only three of the possible thirteen NanoSat missions. In future work, we intend to extend our test problem to the full set of missions and evaluate how well our approach scales.

We also intend to perform an empirical comparison of the preference optimization techniques as well as a third alternative using both techniques in sequence.

Acknowledgements

We would like to acknowledge Charlie Livaudais and Ethan Bogart, two exceptional undergraduate students, for their contributions to the implementation of the Stretch-and-Shift algorithm, as well as our former collaborators for their foundational work; in particular Alfredo Bencomo for his work to adapt OpenSPIFe for this application, and Ken Galal for his work in modeling EM-1 and NanoSat mission trajectories, and informing our work with his extensive spaceflight mission expertise.

We would also like to acknowledge our sponsors: the Autonomous Mission Operations program led by Dr. Jeremy Frank; and the NASA Ames - Center Innovation Fund (CIF), led by Center Chief Technologist, Harry Partridge, who provided the original funding to get the project off the ground in 2016.

References

Abraham, D., MacNeal, B., Heckman, D. 2016. Enabling Affordable Communications for the Burgeoning Deep SpaceCubeSat Fleet. In *Proceedings of SpaceOps 2016*.

Augenstein, S., Estanisiao, A., Guere, E., and Blaes, S., 2016. Optimal Scheduling of a Constellation of Earth-Imaging Satellites, for Maximal Data Throughput and Efficient Human Management. In *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS 2016).*

Bartak, R., Morris, R.A., Venable, K.B. 2014. An introduction to constraint-based temporal reasoning. Morgan & Claypool Publishers.

Barták, R., Cepek, O. 2007. Temporal Networks with Alternatives: Complexity and Model. *FLAIRS Conference 2007*: Pages :641-646.

Dechter, R. Itay Meiri, I., Judea Pearl, J. 1991. Temporal Constraint Networks. *Artif. Intell.* 49(1-3): 61-95.

Khatib, L., Morris, P.H., Morris, R.A., Rossi, F., Sperduti, A., Venable, K.B. 2007. Solving and learning a tractable class of soft temporal constraints: Theoretical and experimental results. *AI Commun.* 20(3): 181-209.

Peintner, P., Pollack, M.E. 2005. Anytime, Complete Algorithm for Finding Utilitarian Optimal Solutions to STPPs. *AAAI 2005:* 443-448.

Pinover, K., Johnston, M.D., Lee, C. 2017. Optimizing SmallSat Scheduling for NASA's Deep Space Network. In *Proceedings of the 10th International Workshop on Planning and Scheduling for Space (IWPSS 2017).*

Rossi, F., Venable, K.B., Yorke-Smith, N. 2006. Uncertainty in Soft Temporal Constraint Problems: A General Framework and Controllability Algorithms for The Fuzzy Case. *J. Artif. Intell. Res.* 27: 617-674.

Skobelev, P., Simonova, E., Ivanov, A., Mayorov, I., Travin, V., and Zhilyaev, A. 2014. Real Time Scheduling of Data Transmission Sessions in a Microsatellites Swarm and Ground Stations Network Based on Multi-Agent Technology. In *Proceeding Proceedings of the International Joint Conference on Computational Intelligence (IJCCI2014)* - Volume 1 Pages 153-159, SCITEPRESS.

Tsamardinos, I., Vidal, T., Pollack, M.E. 2003. CTP: A New Constraint Based Formalism for Conditional, Temporal Planning. *Constraints* 8(4): 365-388.

Vidal, T., Fargier, H. 1999. Handling contingency in temporal constraint networks: from consistency to controllabilities. *J. Exp. Theor. Artif. Intell.* 11(1): 23-45.