Scheduling NASA's Deep Space Network: Priorities, Preferences, and Optimization

Mark D. Johnston
Jet Propulsion Laboratory, California Institute of Technology
4800 Oak Grove Dr., Pasadena CA 91109
mark.d.johnston @ jpl.nasa.gov

Abstract

NASA's Deep Space Network (DSN) is the primary resource for communications and navigation for interplanetary space missions, for both NASA and partner agencies. Growth in mission demand, both in number of spacecraft and in data return, has led to increased loading levels on the network, and actual demand frequently exceeds network capacity. The DSN scheduling process involves peer-to-peer collaborative negotiation, which consumes significant time and resources in order to reach a baseline version of the schedule, and then to manage and agree to changes. Process delays are exacerbated by the high level of oversubscription experienced by the DSN: it is not unusual for the scheduling process to start with 20-40% more requested time can be accommodated on the available antennas. The other NASA networks make use of a static mission priority list to address a similar problem: missions are ranked in priority order, then the schedule is populated by priority from highest to lowest. Such a mechanism would not work for DSN due to the heterogeneity of the mission set, and to the time-varying mission requirements with mission phase. This paper describes an alternative approach for the DSN that addresses key problems inherent in the current process — oversubsubscription and how to "fairly" reduce it to a manageable level. The main characteristics of the new approach are the use of loading-based limits based on balancing requested time, along with priorities and user preferences as the basis for optimization criteria that can be used by new algorithms.

Introduction

NASA's Deep Space Network (DSN) consists of three communications complexes, located in Goldstone, California; Madrid, Spain; and Canberra, Australia (Figure 1). Each complex contains one 70-meter antenna and three or four 34-meter antennas. These ground antennas are responsible for communications and navigation support for a wide range of scientific space missions, from those in highly elliptical earth orbits, to some beyond the solar system. In future years, DSN will also support human missions to the moon and beyond. The placement of the three DSN complexes allows at least one of them to be in view of any distant spacecraft at all times(Imbriale 2003).

The current DSN scheduling process is lengthy (with a lead time of around four months) and labor intensive. It relies on peer-to-peer negotiation for changes, with frequent proposals and counter-proposals, and so it is a major challenge to add a large number of new missions without impacting the current mission users. This calls for new approaches to scheduling that minimize the impact of adding new missions, while accommodating existing ones. At the same time, it is important to satisfy mission tracking and telecommunications requirements to the greatest extent possible, on an ongoing basis, week after week.

We have investigated, and are in the process of deploying, a novel approach to this problem, characterized by the following:

1. Use advance (>6 months) planning/loading requirements, provided by users, to calculate overall anticipated loading, and then derive limits on the time allowed for submission later in the detailed scheduling process based on what can be supported by the available antenna and other resources
2. Require users to categorize their requests into *normal* and *augmented* priority levels, based on objective criteria that apply across the entire user base
3. Require users to also associate one of a small number (<10) preference levels with each of their scheduling requests, thus providing a relative ranking of their own submissions, and then optimize the generated schedule to satisfy as large a fraction of higher preference requests as possible across the entire mission set.

We have prototyped a system based on this approach and have run an initial series of experiments to investigate the quality of the schedules generated, as well as the degree to which user preferences could be met in practice. Results to date are encouraging, and it is planned to conduct more extensive trials in the near future. Moving to this paradigm could enable the DSN to greatly reduce the time and effort to build and manage schedules, while still allowing for unexpected late changes that are not uncommon in this domain.

In the following, we first briefly describe the DSN scheduling process, and then some of the factors that come

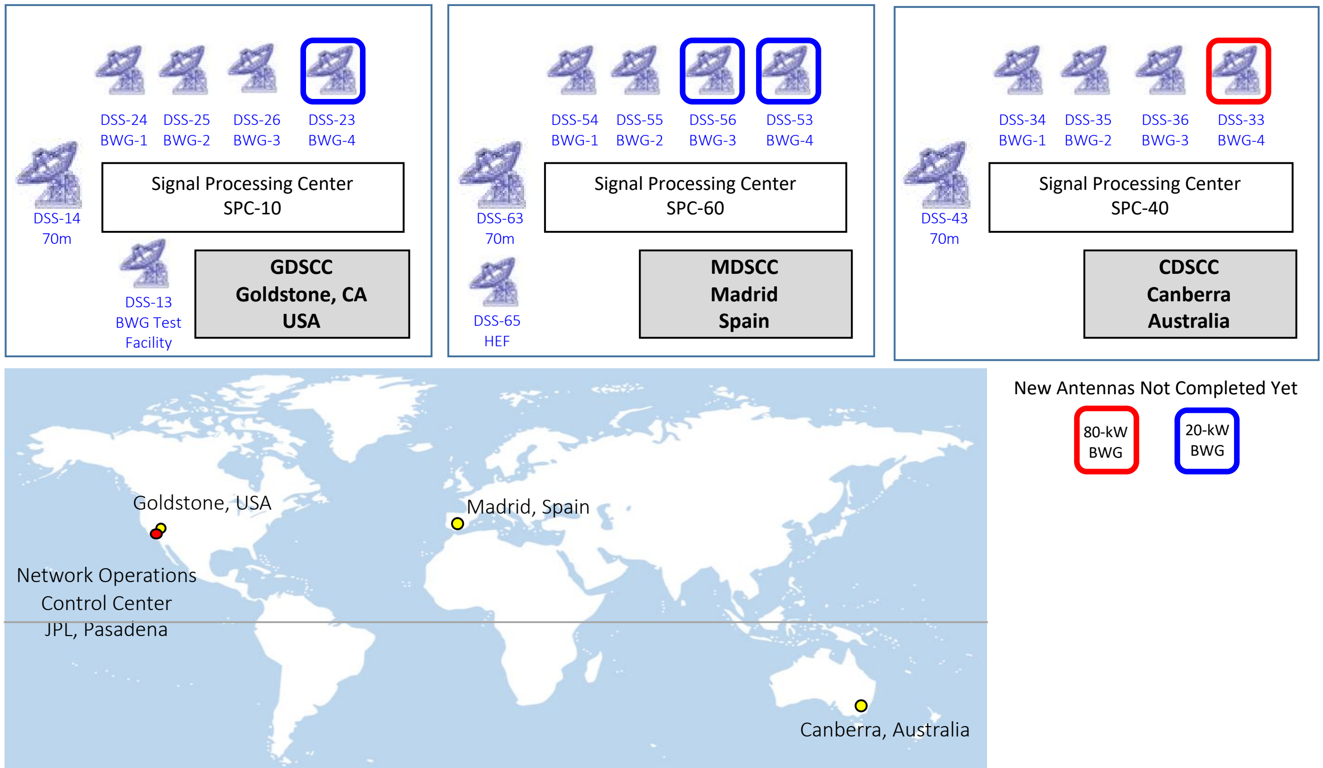

Figure 1: An overview diagram of the NASA Deep Space Network (DSN) antennas at the three complexes in Goldstone, CA, Madrid, Spain, and Canberra, Australia. As of early 2020 there are 12 active antennas, expected to grow to 15 by 2024 with the construction of new antennas and the decommissioning of some older ones. A typical week includes around 500 scheduled activities for about 40 missions or other users.

into play as more missions are included in the DSN processes. This is followed by discussion of the new approach, and then by conclusions and directions for future work.

DSN Scheduling Process Overview

Figure 2 shows a block diagram of the DSN planning and scheduling systems, indicating the key mission interfaces and data flows. On the longest time scale, the Loading Analysis and Planning Software (LAPS) (Johnston and Lad 2018) provides for long-range planning and forecasting, several years into the future, taking into account anticipated mission requirements and planned DSN asset capabilities. LAPS users include analysts and mission planners, as well as others involved in long-term planning of DSN activities. At this stage, the missions provide long-term ephemeris information along with their expected utilization of DSN resources, with as much fidelity as possible.

The Service Scheduling Software (SSS, or S^3) maintains user-provided detailed scheduling requirements and expands them into specific communications and navigation passes, taking into account resource constraints (antennas and equipment) as well as a wide variety of DSN operational constraints and rules. S^3 supports the scheduling negotiation process, followed by a consensus-based change process with workflows for approval of all schedule modifications by authorized mission representatives. More accurate ephemeris information, along with specific communications and navigation requirements, are provided by the missions during this phase. At the conclusion of this phase, missions receive negotiated DSN allocations that they use to plan and sequence their onboard activities, usually far in advance (weeks to months).

The Service Preparation Subsystem (SPS) merges the schedule with mission-provided sequence and ephemeris information to generate detailed DSN ground system sequences of events, along with a variety of predict calculations to support each scheduled activity. These are distributed to the DSN complexes where they are used by the Network Monitoring and Control (NMC) subsystem to handle the execution of each pass.

DSN scheduling differs from the other NASA networks in large part due to the operating characteristics of most of its supported missions. Deep space missions typically do extensive advance planning and scheduling, to the point of building detailed command sequences that are uploaded weeks or months ahead of execution, reflecting a range of mission phases, science events, and engineering activities. Additionally, there are extensive checks on these plans and sequences, as an error can be catastrophic. Long light travel times preclude any significant real-time interaction. As a re-

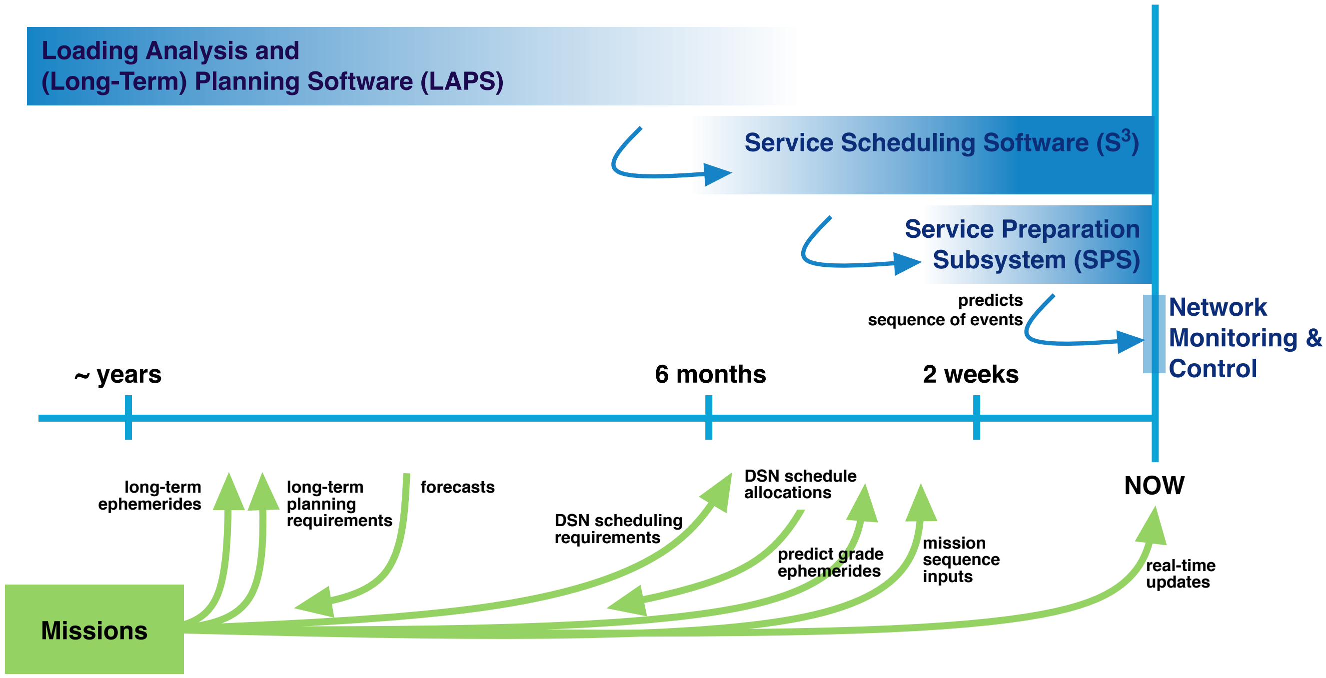

Figure 2: Schematic of the flow of information through the end-to-end planning, scheduling, and execution monitoring and control systems (top), highlighting interactions with mission users (bottom). Priority scheduling makes use of forecast supportable loading levels from LAPS, along with priorities and preferences in S^3, to address a key bottleneck in the process (see Figure 5).

sult, the DSN schedule is baselined months ahead of time, with changes occurring only when agreed to by all involved.

The DSN scheduling process (Johnston et al. 2012; 2014) operates on a rolling weekly basis (see Figure 5): as the deadline for a week approaches (roughly four months before the start of the week), mission scheduling representatives enter the requirements for that mission into S^3. Once all inputs for a week are in, they are integrated into a single schedule and the DSN Scheduling Engine (DSE) (Johnston et al. 2010) is run to deconflict as much as possible, given any specified flexibilities in the input requirements from each mission. In practice, little flexibility is allowed by users in their initial specifications, and the net oversubscription level means that many conflicts necessarily remain in the schedule.

Once the scheduling engine has been run, and conflicts reduced automatically as much as possible, a human scheduler called "Builder of Proposal", or BOP, starts to work on the schedule and makes further changes based on experience and background knowledge of each mission's requirements, plus information about any special needs during the week. These changes include: deleting some activities, shortening tracks below their specified minimums, splitting tracks designated as unsplittable, placing the (now shorter) segments into gaps in the schedule, and moving tracks to different antennas. This a time-consuming and labor-intensive process, requiring a great deal of familiarity with the entire DSN mission set and their typical requirement patterns and unstated flexibilities. The BOP can generally eliminate hundreds of conflicts, but at the end there usually remain 10-20 conflicting activities, at which point the week is released to the full set of mission scheduling representatives to clear the remaining conflicts and to mutually negotiate any adjustments to changes made by the BOP. In this phase, individual mission representatives collaboratively negotiate peer-to-peer to reach a state where all users are agreed (Carruth et al. 2010). In this process, one user will propose a set of changes, to which all affected users must concur before it becomes the new baseline. If any user disagrees with the changes, it falls on him or her to counter-propose an alternative, with a justification (where just undoing a previous proposal is generally not allowed!). This process continues until the deadline is reached, at which point conflicts are either cleared or (rarely) waived, and the schedule is considered baselined and published. From the completion of the automated scheduling run to the baseline conflict-free schedule is typically 2-3 weeks. The overall duration of this process means that multiple weeks are being worked on in parallel, with over ~5 months in the pipeline in normal operations, with about 12 or more weeks negotiated and stable.

Priority Scheduling

DSN has been considering developing a priority scheduling scheme for some time. The motivation for this includes a desire to reduce the time and effort to prepare and publish the baseline schedules, to reduce the effort of the BOP, and to improve consistency in the number of negotiated weeks that are available for missions to plan their detailed activities. It's also been suggested that DSN use a priority system for consistency with the other NASA networks, specifically

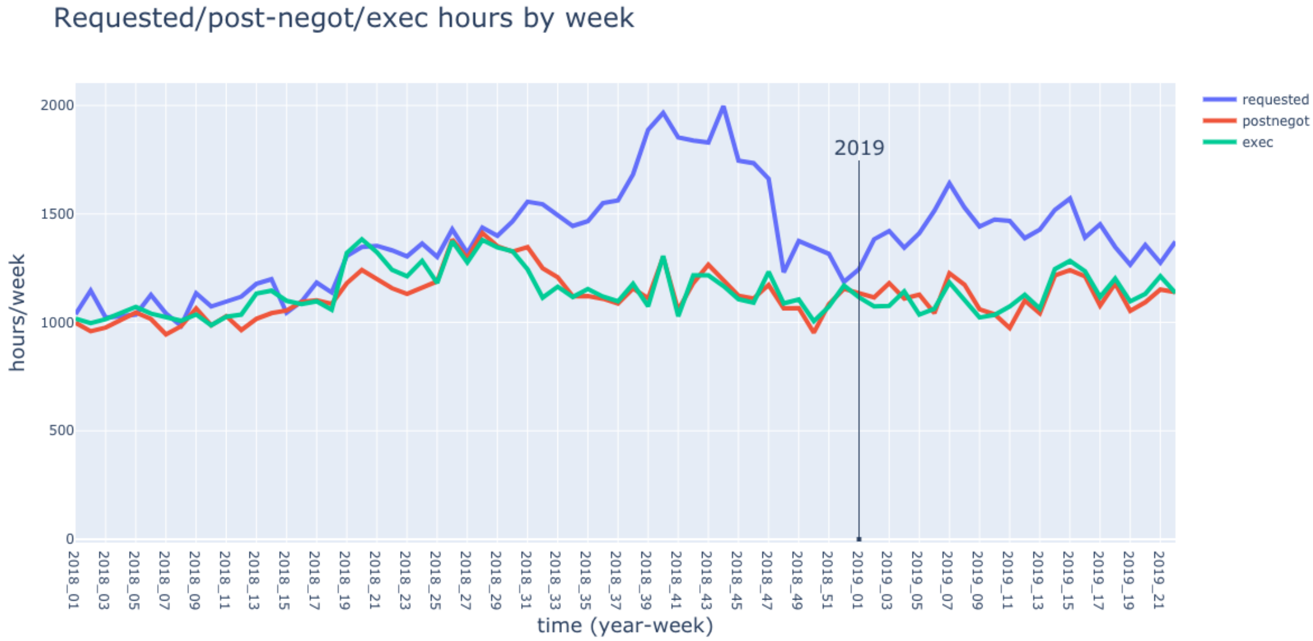

Figure 3: DSN antenna oversubscription vs. time for a period in 2018-2019. The chart shows the ratio of requested (blue) and negotiated (red) time to that actually executed (green), for a range of weeks. The peak overage was about 60% in late 2018, corresponding to the simultaneous occurrence of critical spacecraft events for three different missions.

the Near Earth Network (NEN) and Space Network (SN). Both of these networks use a very similar priority scheme as part of their routine scheduling processes. While DSN has adopted a resource-based prioritization policy for scarce 70m time, in that missions that can only use the 70m antennas are preferentially given time on them, this is very different from a general mission-based prioritization.

Both NEN and SN use a *two-dimensional* priority scheme: the first and overriding dimension is often called *event* or *absolute priority*. It reflects the overall importance of an activity in some absolute and objective way, as a relatively small number of categories and subcategories into which activities can be slotted. The second dimension is a strict *mission ranking*, such that more highly ranked missions have their scheduling requests satisfied first. The absolute priority list includes such categories as human spaceflight, launch and landing services, critical operations, etc. and goes down to normal operations and various kinds of network tests. The scheduling process works to satisfy user requests in order down the absolute priority list, then the mission priority list at the same absolute level, etc., until all requests are satisfied, or there are no further placement possibilities.

NEN and SN have a process for developing and approving their priority lists, run by Goddard Space Flight Center and with concurrence by NASA Headquarters. These lists are updated about once per year, mostly to reflect the changing mission mix as new missions begin and old ones end. There are several important differences between the scheduling processes for NEN/SN and DSN:

- NEN and SN are entirely scheduled by a central authority based at White Sands, New Mexico: there is no mission-to-mission negotiation, and the full schedule is not published to the user community – only that subset directly required for each mission
- NEN and SN both run their processes on a very short timescale compared to DSN: requirements are due two weeks before a week starts execution, and the confirmed schedule is published one week before. In the week before execution, mission priorities are not used and time is allocated on a first-come, first-served, basis

DSN does have an event (absolute) priority schema that specifies types of activities that are more important, as recognized by all participants in the process. These include support for human spaceflight, spacecraft emergencies and survival activities, major and unique scientific events, etc. all of which rank higher than normal DSN non-time-critical science activities. That said, only a small percentage of each week's time falls into these higher priority categories, with the exception of antenna maintenance. Launches and planetary orbit insertions and landings are examples of higher priority activities when they occur, as well as in-flight maneuvers. In the past year, fewer than 1% of over 24,000 scheduled DSN activities were at support levels higher than routine.

What DSN does not have is a ranked mission list: there has never been a process by which agreement could be reached on which missions are higher priority than others. Instead, DSN uses a peer-to-peer collaborative process to negotiate time and changes: even a small extended mission can engage in negotiation with a recently launched flagship to make mutually agreeable schedule changes. There are other reasons why a ranked mission list is problematic for DSN:(Shouraboura, Johnston, and Tran 2016; Johnston 2019)

- Some missions require only a few hours per week to meet their mission goals, while others seek and can use nearly full time coverage. A strict mission ranking could leave some missions completely out of the schedule, which would undoubtedly be viewed as unacceptable
- Many DSN users have requirements that change frequently with mission phase or with planned science activities, week to week or day to day – not reflected well by a static priority list
- Given the variable mix of activities and mission phase updates, a mission priority list would have to change so frequently as to be essentially useless

All of these factors have played into the evolution of the current process and away from the static mission priority list adopted by the other NASA networks.

Previous investigations of priority schemes for DSN have made suggestions for addressing some of these concerns (Shouraboura et al., 2016), which will be considered after examining the role of oversubscription in the next section.

Oversubscription

The DSN is routinely oversubscribed by a variable amount, depending on the current mission set and their activities. An illustration of this is shown in Figure 3, where the oversubscription level is plotted vs. time for a period of time in 2018-2019. The ramp up in late 2018 is due primarily to three factors: the arrival of the InSight mission at Mars in late November 2018, and the arrival of the two asteroid missions at their targets, OSIRIS-REx and Hayabusa2, also in the fall of 2018.

When a mission reaches its destination science target, such as a planet or asteroid, it can go from minimal DSN usage to nearly continuous coverage for some period, sometimes months or years. When oversubscription reaches a level of as little as 10% it is equivalent to having one additional antenna's worth of demand to remove before a feasible schedule can be developed and published. At oversubscription levels of 60%, the scheduling engine algorithms, which are prohibited from dropping requirements, can do little more than try to spread out the overage. Most of the burden of dealing with the oversubscription falls on the BOP, who has to make wholesale reductions to come up with a feasible proposed schedule for negotiation. Negotiation itself is more arduous because missions try to accommodate the reductions and adjust the schedule to their best advantage. In spite of this, consensus is nearly always reached, and the final schedule is regarded as sufficiently fair that the process of escalating disagreements is virtually never invoked.

Categorizing oversubscription by mission reveals some interesting facts (see Figure 4). While DSN supports a wide range of mission users, most of the oversubscription (80%) is coming from a small proportion (25%) of heavy users. Some users submit requests for time up to twice what can actually be accommodated, and in general, 40% come in at 20% more than can be fit. This is in part due to a lack of visibility in what can actually fit, in part to volatility of the DSN downtime plan, but mostly due to the desire of missions to submit what they would "like to have" if possible, even though they routinely settle for much less time.

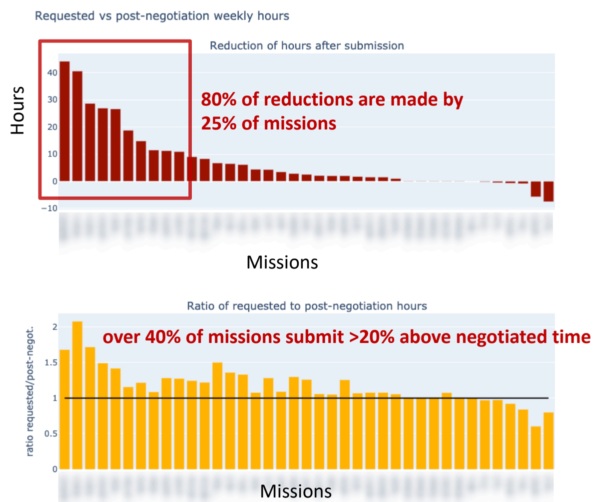

Figure 4: Oversubscription by mission (name obscured) over a one year period of DSN scheduling. Many missions enter the process requesting significantly more time than can be fit into the schedule, necessitating human intervention to reduce the requested time to a supportable level. The top chart shows that most reductions are made by 25% of the mission set, while the bottom chart shows that most missions submit 20% more time than they will obtain on average, with some requesting more by up to a factor of 2.

User Preferences and Loading-Derived Limits

Figure 5 illustrates the overall lifecycle and timeline of a week in the DSN scheduling process. The numbered steps corresponding to the following:

1. Users must enter their scheduling requirements at a deadline per week, roughly 5 months ahead of start of execution of the week.
2. The submitted requirements are integrated and conflict reduction algorithms are run, followed by human expert conflict reduction, in the stage designated "Builder of Proposal", or BOP.
3. When complete, the BOP releases the week to negotiation as a "Negotiation Workspace", or NWS.

When negotiation is complete, the baselined conflict-free schedule can still be changed by mutual agreement in a "Proposal Workspace", or PWS. The actual concurred schedule is stored as the "Master" schedule, which is synchronized to the DSN complexes to control the execution of each tracking pass, engineering, science, and maintenance activity.

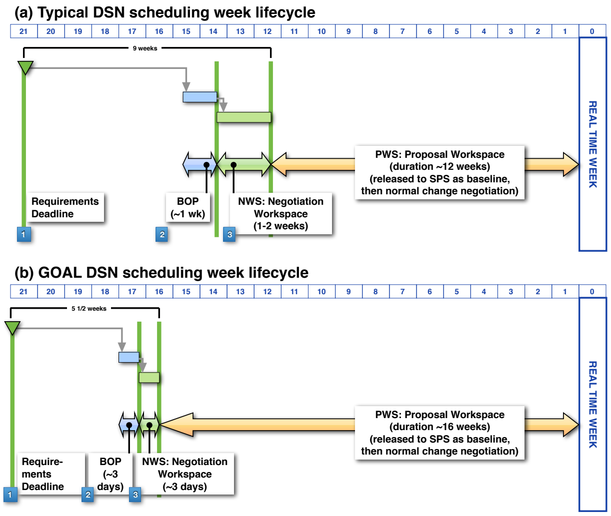

Figure 5: Lifecycle of a week in the DSN scheduling process, from requirements entry to execution. The main stages, as described in the text, including reducing conflicts with human assistance (the "Builder of Proposal", or BOP), and negotiation of any remaining conflicts in order to baseline the schedule. The overall timeline is typically 5 months in duration. (a) the timeline before introduction of requirement limits and priority scheduling, and (b) the goal timeline afterwards. The objective is as much as a 3x reduction in workload while increasing the number of negotiated weeks by 35%.

The concept being developed for infusing priorities, limits, and optimization is called "User Preference Optimization". This name is meant to indicate that "priorities" that pit one mission against another are not involved, but instead that *individual mission user preferences among their own activities are driving the process*. The primary objective shifts from eliminating conflicts, to maximizing the weighted degree of satisfaction of user requirements, while eliminating conflicts by a combination of requirement flexibilities (stated and implicit), limiting input to a near feasible level, and dropping low preference requests when they cannot be made to fit without causing conflicts. These enter into the lifecycle illustrated in Figure 5 as follows:

1. Limit each week's submitted requirements to a cap calculated to be supportable by anticipated network resources (possibly allowing for a small margin to account for uncertainty). The cap is calculated by the Loading Analysis and Planning Software (LAPS), based on projected planning requirements from all missions, along with antenna engineering activities and downtime, and commissioning and decommissioning of antennas. This requires up-to-date long-range planning requirements to be provided by each mission (most do this already), and may require explicit deconfliction if contention levels are too high when projected into the future.
2. Require each mission to provide a *relative prioritization (preference level)* for their own inputs (only). This would be done by each user assigning every request to one of a small number of "levels" or "tiers" (current planning it to use the range 1...7, with 1 being most important). Furthermore, it would be required that time within the cap must be within a certain preference range (currently planned for 1...4)

   The significance of this is that it has users explicitly indicate what is most important to them, while avoiding any attempt to compare importance of one mission relative to another. The specification of user preference levels would be such that all requests at the same level are equally important to that mission, while any request in a higher level is more important than any request at a lower level. User preference levels would not replace the current DSN absolute priority scheme, but it would be expected that critical activities such as launches, orbit insertions, landings, etc. would also be given high levels of user preference, corresponding to their treatment as DSN critical events.

   The incentive to provide this information is the knowledge that lower preference requests will be omitted if they don't fit. This means less work for a user than if a high-preference request was omitted, since it would have to be negotiated back into the schedule, potentially requiring concurrence from numerous other missions for changes that frequently ripple to affect more and more missions.
3. At BOP time, run user preference optimization algorithms that search for feasible (conflict free) schedules that maximize the scheduled preference level for all missions. All provided flexibility will be used to fit as much as possible into the schedule, but requests that cannot fit will be dropped at this point (though may be negotiated in later by the affected missions). This flexibility can including shrinking requested tracking duration, splitting tracks across antennas or complexes, adjusting min/max gaps between tracks, etc. While the BOP is expected to still intervene as a human expert to make explicit adjustments in some cases, it is a goal to reduce the BOP level of effort by a factor of 5 over current levels. Following the BOP schedule release, users would negotiate as they do today. However, the time allowed to conduct negotiation could be reduced to ensure the pipeline of negotiated weeks remains consistently far enough ahead of realtime.
4. During and after negotiation, only *absolute* priorities would be considered and consensus-based changes would remain the norm. This is how the schedule is maintained today, in that late changes for spacecraft emergencies or hardware outages are accommodated, but changes impacting a mission must first be concurred by the affected participants. This is also how the other NASA networks handle changes after the schedule is released, except that changes are made only by the central authority rather than peer-to-peer.

Priority/Preference Algorithms for DSN Scheduling

The DSN Scheduling Engine uses minimizing conflicts (e.g. two activities on the same antenna at the same time) as a

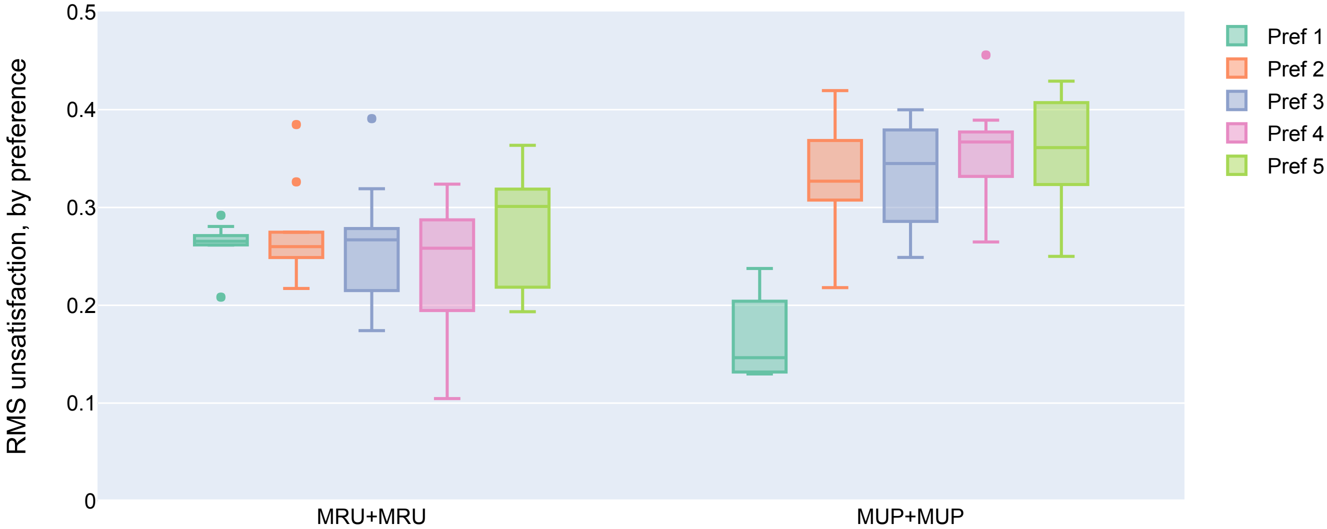

Figure 6: Comparison of two algorithms with respect to satisfying higher-preference bin time scheduled: the left chart (MRU+MRU) uses min RMS unsatisfied time as the heuristic for phases (1) and (3) (Eqn. 1), and the right chart uses a per-bin unsatisfied time heuristic based on $U_{max,j}$ (Eqn. 3). The latter does much better at satisfying preference=1 requests. Statistics for 10 cases with randomized preference assignments are shown, for one oversubscribed DSN schedule week.

primary driving objective, subject to constraints on timing and duration, resources, gaps to other related activities, and a wide range of other factors e.g (Johnston et al. 2010). Up to now, user requirements have only indicated absolute priority, not any degree of relative preference, and so it has not been feasible to deconflict the schedule by omitting lower preference activities.

In introducing relative preferences, we also need to define a new objective function to guide schedule comparison and search. One such metric, to be minimized, is an overall RMS unsatisfied time fraction:

$$U = \sqrt{\frac{1}{N} \sum_{i=1...N} \left(\frac{T_{Ri} - T_{Si}}{T_{Ri}} \right)^2} \quad (1)$$

where i ranges over missions $1 \dots N$, and T_{Ri} and T_{Si} are the tracking time requested and scheduled, respectively, for mission i. While this will reflect an overall measure of unsatisfied requests, it does not distinguish the situation where one mission is "pushed out" of the schedule altogether ($T_{Si} = 0$), an unacceptable situation. So a better metric is:

$$U_{max} = \max_{i \in \{1...N\}} \left(\frac{T_{Ri} - T_{Si}}{T_{Ri}} \right) \quad (2)$$

which indicates the worst case (max) unsatisfaction of any individual mission, where 1.0 means *no* requested time for that mission was included in the schedule. The shortcoming of metrics like U and U_{max} is that they do not distinguish between different user preference levels for activities that make it into the schedule. To address this, we can differentiate by preference level (assuming that absolute priorities are reflected in user preferences, as noted above):

$$U_{max,j} = \max_{i \in \{1...N\}} \left(\frac{T_{Rji} - T_{Sji}}{T_{Rji}} \right) \quad (3)$$

where j ranges over the set of preference levels, $j \in \{1...J\}$. We can compare two schedules through lexicographic comparison of their $U_{max,j}$ vectors. While this emphasizes comparisons based on user satisfaction, there are of course additional metrics such as number of conflicts, number of requirement violations, antenna utilization, etc. that need to be considered in generating and proposing full schedules.

We have investigated several algorithms for generating schedules based on sets of collected request inputs, and have implemented a testbed for comparing different heuristic search techniques. For the moment, we are experimenting with a 3-phase algorithm as follows:

1. **Initial assignment**: greedy initial assignment of all requests to some valid time and resource for that request, avoiding but allowing conflicts
2. **Min-conflict hill climbing**: use a min-conflicts heuristic to repair the schedule and reduce conflicts leaving all requests assigned (Minton et al. 1992)
3. **Deconflict and repair**: remove conflicting activities and (selectively) add back others that may now have feasible places

In this approach, each phase can make use of different heuristics, and we can assess the combinations that provide the most promising results and evaluate them on larger data sets. While this analysis is not complete, some promising results are already seen. As an example, we have selected as a test week 14 of 2019, which was oversubscribed by about

25%. Because there is no historical preference data, we have used a randomized preference assignment to each request, from a set of 5 preference bins. We selected a random subset of requests based on percentage of total requested time as follows: 60% as preference 1, 15% as 2, 10% as 3, 8% as 4, and 7% as 5. We then compared two different heuristics for initial assignment (1) and deconfliction (3) (using the same min conflicts repair stage (2)). We made 10 runs with different preference bin randomized assignments, to see the sensitivity on which specific activities were flagged as higher preference. The results are shown in Figure 6. It is clear that maximizing higher user preference levels does a better job of satisfying those requirements, at the expense of the lower levels.

One of our next steps will be try some different approaches, such as squeaky-wheel optimization as a means to dynamically prioritize requests (Barbulescu, Howe, and Whitley 2006; Joslin and Clements 1999; Lewellen et al. 2017), and to incorporate tabu into the conflict reduction step (2). We are also looking into the potential value of multi-objective optimization (Johnston and Giuliano 2011) as a means to evaluate tradeoffs among missions in the course of generating the schedule. We note that aspects of this problem are similar to those encountered in "fair division" problems — e.g. (Kash, Procaccia, and Shah 2014) — and we are investigating that literature as well.

Conclusions and Next Steps

In this paper we have described a new approach for DSN scheduling that addresses several key objectives, notably the reduction of effort in key bottlenecks in the DSN scheduling process. While not complete, we have made progress in several aspects. In particular, we have:

- implemented the interfaces and logic to transfer limits from the LAPS loading and forecast software into the midrange schedule system S^3, and to edit those limits as needed.
- implemented the association of priorities and preferences with S^3 detailed scheduling requests, and to tally those so users can see where their inputs fall with respect to their limits.
- conducted an initial field test, where users took an existing schedule week and retroactively added priorities and preferences, after which the week was processed just as the original week had been. The results are extremely promising in that with minimal additional effort on the part of the users, the BOP (bottleneck process) was able to substantially speed up processing of the week with about 400 tracking hours (30% of the original submission) removed as over limit, and the remaining time labeled as to what was most important to each user. The estimated speedup is that the work completed in 40-60% of the original time.
- implemented an algorithm testbed and made initial comparison studies of different heuristics to generate end deconflict schedules.

While there remains significant work to go to validate and fully implement this approach, it should provide a significant reduction in effort and cost in the DSN scheduling process.

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

References

Barbulescu, L.; Howe, A.; and Whitley, D. 2006. AFSCN scheduling: How the problem and solution have evolved. *Mathematical and Computer Modelling* 43:1023–1037.

Carruth, J.; Johnston, M. D.; Coffman, A.; Wallace, M.; Arroyo, B.; and Malhotra, S. 2010. A Collaborative Scheduling Environment for NASA's Deep Space Network. In *SpaceOps 2010*.

Imbriale, W. A. 2003. *Large Antennas of the Deep Space Network*. Wiley.

Johnston, M. D., and Giuliano, M. 2011. Multi-Objective Scheduling for Space Science Missions. *Journal of Advanced Computational Intelligence and Intelligent Informatics (JACII)* 15:1140–1148.

Johnston, M. D., and Lad, J. 2018. Integrated Planning and Scheduling for NASA's Deep Space Network – from Forecasting to Real-time. In *SpaceOps*.

Johnston, M. D.; Tran, D.; Arroyo, B.; Call, J.; and Mercado, M. 2010. Request-Driven Schedule Automation for the Deep Space Network. In *SpaceOps 2010*.

Johnston, M. D.; Tran, D.; Arroyo, B.; Sorensen, S.; Tay, P.; Carruth, J.; Coffman, A.; and Wallace, M. 2012. Automating Mid- and Long-Range Scheduling for NASA's Deep Space Network. In *SpaceOps 2012*.

Johnston, M.; Tran, D.; Arroyo, B.; Sorensen, S.; Tay, P.; Carruth, J.; Coffman, A.; and Wallace, M. 2014. Automated Scheduling for NASA's Deep Space Network. *AI Magazine* 35:7–25.

Johnston, M. D. 2019. User Preference Optimization for Oversubscribed Scheduling of NASA's Deep Space Network. In *11th International Workshop on Planning and Scheduling for Space (IWPSS)*, 86–92.

Joslin, D. E., and Clements, D. P. 1999. Squeaky Wheel Optimization. *Journal of AI Research* 10:353–373.

Kash, I.; Procaccia, A.; and Shah, N. 2014. No Agent Left Behind: Dynamic Fair Division of Multiple Resources. *Journal of AI Research* 51:579–603.

Lewellen, G.; Davies, C.; Byon, A.; Knight, R.; Shao, E.; Tran, D.; and Trowbridge, M. 2017. A Hybrid Traveling Salesman Problem - Squeaky Wheel Optimization Planner for Earth Observational Scheduling. In *10th International Workshop on Planning and Scheduling for Space*.

Minton, S.; Johnston, M. D.; Philips, A.; and Laird, P. 1992. Minimizing Conflicts: A Heuristic Repair Method for Constraint Satisfaction and Scheduling Problems. *Artificial Intelligence Journal* 58:161–205.

Shouraboura, C.; Johnston, M. D.; and Tran, D. 2016. Prioritization and Oversubscribed Scheduling for NASA's Deep Space Network. In *Proceedings*.