Investigating a Demand Access Scheduling Paradigm for NASA's Deep Space Network

Timothy M. Hackett and **Sven G. Bilén**
School of Electrical Engineering
and Computer Science
The Pennsylvania State University
101 Electrical Engineering East
University Park, Pennsylvania 16802

Mark D. Johnston
Jet Propulsion Laboratory/
California Institute of Technology
4800 Oak Grove Drive
Pasadena, California 91109

Abstract

NASA's Deep Space Network supports the communications to and from spacecraft, rovers, and landers across our solar system and beyond. The weekly tracking requirements for these spacecraft are scheduled by mission representatives at least eight weeks before the start of the track through a combination of automated algorithms and peer-to-peer negotiations. This process has worked well for traditional users with deterministic science collections, such as those doing mapping and imaging. But, this process will not scale well to accommodate a new class of users whose data collections are not completely predictable (i.e., event-driven science), which includes both traditional, deterministic users with unexpected discoveries as well as autonomous users exploring and monitoring for certain events, such as solar flares. In this paper, we propose a "demand access" scheduling approach in which the spacecraft, rovers, and landers themselves request track time on the network using a beacon-tone system and are scheduled track time "on-the-fly" using pre-scheduled shared-user block tracks. We show through simulation that this demand-access approach can both decrease the mean duration between the time of data collection to the start of the downlink and the number of tracks required compared to the traditional scheduling method for an example mission concept of autonomous SmallSat explorers at near-Earth asteroids. We also show how this demand-access approach can be used in combination with the traditional scheduling method to support legacy users.

Introduction

NASA's Deep Space Network (DSN) supports the communications and tracking for spacecraft, rovers, and landers across our solar system and beyond for a variety of space agencies around the world. The network consists of three main ground complexes in Goldstone, CA, USA; Madrid, Spain; and Canberra, Australia with an experimental station in Morehead, KY, USA. Each main complex consists of one 70-m and three or four 34-m parabolic dish antennas; the Morehead antenna has one 21-m antenna (Nelson 2018; Malphrus et al. 2018). At least one of the complexes is in view of a spacecraft at any given time between the three main ground complexes (Imbriale 2002). The Morehead antenna has been performing technology demonstrations in support of becoming an operational capability for future deep-space SmallSat missions (Martinez 2018).

Currently, the DSN supports a few dozen missions, which are scheduled through a process that starts roughly four months prior to the start of the track start time. This process starts with mission representatives submitting their tracking requirements into the Service Scheduling Software (S3). These tracking requirements include constraints and preferences, such as the number of tracks during the schedule week, the duration of these tracks, and timing and dependence between tracks. The DSN Scheduling Engine (DSE) compiles all of the individual mission requests into one master schedule and then systematically deconflicts the schedule. The level at which the schedule can be deconflicted is dependent on both the flexibility of the submitted requirements as well as the trajectories of the spacecraft. The schedule is then further deconflicted by a human scheduler named the "Builder of Proposal" (BOP), who uses her expert knowledge on DSN scheduling and the context of the various missions and their current mission phases. This process generally results in about 10–20 remaining conflicts that need to then be resolved using a peer-to-peer negotiation process facilitated in S3. Once the remaining conflicts have been negotiated and resolved, the master schedule is baselined. For a typical schedule week, the schedule is baselined eight weeks or more ahead of the start of the schedule week. This process is very labor and time intensive and requires a couple dozen full-time mission representatives working on multiple schedule weeks in parallel (Carruth et al. 2010; Pinover, Johnston, and Lee 2017).

This scheduling process has worked well so far partially due to the relatively low number of missions supported by the DSN at any given time. This process will not scale well to support future mission concepts deploying groups of spacecraft as secondary payloads, such as the SmallSats on Exploration Mission 1 (EM-1). Furthermore, this process has worked well so far because currently supported missions generally have deterministic data collection processes, such as those doing routine mapping and imaging. The amount of data collected in a time period is entirely predictable or constrained by mission representatives on the

ground, so the tracking requirements can be pre-scheduled months in advance. This does not allow for last-minute flexibility in the case of unexpected discoveries. For example, when images from OSIRIS-REx revealed dust plumes eminating from *Bennu*'s surface in March 2019 (Lauretta et al. 2019), the tracking requirements for the following weeks could not be easily adapted based on this discovery as they were already set into the master schedule weeks before the discovery. For missions based on event-driven science, such as heliophysics missions monitoring for major solar activity, pre-scheduling tracks weeks ahead of time could result in a number of those scheduled tracks to yield no useful, new information as no new activity had occurred. The wasted DSN resources could have been used to support other missions instead.

The science community has expressed a need for on-demand communications, where spacecraft and missions can request communications and tracking time in "near real-time". Providing this ability enables a new class of science missions: event-driven science (Shaw et al. 2018). The traditional static pre-scheduling method described above does not suit these types of missions well due to its lack of adaptability and latency to changing events. In response to this need, on-demand concepts termed "demand access" scheduling and "user-initiated services" (UIS) have been proposed (Johnston and Wyatt 2017; Shaw et al. 2018; Israel et al. 2018). The main idea of these proposed concepts is that the spacecraft itself directly requests time on the network at the moment when it needs tracking time, rather than mission representatives requesting time on behalf of the spacecraft in advance. For spacecraft relatively close to Earth (i.e., LEO and MEO users), spacecraft could use a low-data rate, high availability link to request time on the network through a packetized handshake protocol. This could be accomplished by leveraging the Demand Access Service (DAS) using the multiple access antennas on NASA's Tracking and Data Relay Satellite System (TDRSS) fleet (Gitlin and Horne 2012). For deep space spacecraft, the narrow beamwidths of the larger aperture antennas required to close links along with the long round-trip light times makes this impractical. For these types of missions, a "beacon-mode" contact mechanism has been proposed, in which a spacecraft sends a low-complexity modulated subcarrier tone to indicate spacecraft statuses (Johnston and Wyatt 2017). In this paper, we will focus on the beacon-mode mechanism.

In the late 1990s, the Deep Space 1 (DS1) spacecraft pioneered the Beacon Monitor Operations Experiment (DeCoste et al. 2004; Wyatt et al. 1997; 1999; 1998). The spacecraft sends a beacon tone to the ground depending on its state: "nominal", "interesting", "important", or "urgent" as defined in (DeCoste et al. 2004). A nominal tone indicates the spacecraft is operating nominally with no need for downlinking. An interesting tone indicates when a non-urgent event occurs that can be downlinked when convenient for ground operations (e.g., a single event upset (SEU) causes a reset). An important tone indicates that communications with the ground is time sensitive, for which if no action is taken there is potential for data loss or the state of the spacecraft to worsen. An urgent tone indicates a spacecraft emergency for which ground intervention is required because the spacecraft could not recover on its own. Lastly, if no tone is received from the spacecraft, this could indicate many different scenarios including the spacecraft antenna is not pointing towards Earth or an anomaly is stopping the beacon from being transmitted (DeCoste et al. 2004).

Using the beacon tone monitoring service instead of scheduling traditional telemetry tracks can reduce mission risk, mission cost, and network loading. Because of cost and scheduling constraints, missions are limited to the amount of track time allotted per week. This service could allow missions to have more timely health status reports because the spacecraft will notify the ground immediately that there is an anomaly, as opposed to finding out during the next scheduled telemetry track. Mission cost is decreased by reducing track time and reducing telemetry data, which corresponds to staffing fewer operations analysts. Network loading will be decreased by spacecraft using fewer tracks than in a typical week (DeCoste et al. 2004).

With the validation of the beacon monitoring service, the New Horizons mission to Pluto baselined it into their communications design. The mission estimated that, by using the beacon system, they would be able to decrease their 70-m antenna time by approximately 2274 hours over its mission lifetime (Bowman et al. 2004) for two reasons. First, for the data rates required, the 70-m antenna typically would be required for all telemetry tracks beyond Jupiter (about 4 AU) from Earth. The beacon service allowed the 34-m antenna to be used until 25 AU. Second, beacon tracks take significantly less time than typical telemetry tracks (1.5 hours vs. 8 hours). In 2016, it was estimated that New Horizons was able to save about 80% of its track time using the beacon service (Wyatt et al. 2016).

In addition to previous work using the beacon-tone service for signaling the spacecraft's status, it will now be used to indicate when the spacecraft has new data to downlink. A spacecraft will use the beacon-tone service to indicate that it will be using a followup track to downlink its new science data. When the ground receives the beacon, it will schedule the spacecraft into the next reserved-for-demand-access, unallocated track, if possible (Johnston and Wyatt 2017).

A key enabling concept that will be used for efficiently implementing this beacon-tone demand access system is block scheduling based on geometrical alignment (Pinover, Johnston, and Lee 2017; Hackett, Johnston, and Bilén 2018). Originally proposed as a method for decreasing the complexity and required resources for scheduling users within a small ($\approx 5^\circ$) pointing angle of each other, the same basic algorithms will allow us to find demand-access followup tracks that can be shared among many users.

The remainder of this paper is organized as follows. Section II discusses the major algorithms developed for the beacon-tone demand access scheduling system. Section III describes the setup, assumptions, and results of an example demand-access simulation campaign of near-Earth asteroid (NEA) autonomous SmallSat explorer missions and compares these results with the current, static scheduling method. Finally, Section IV outlines conclusions based on

the simulation findings and identifies future work for continuing research into beacon tone–based demand access scheduling.

Discussion of Algorithms and Mechanisms

Demand access scheduling can take on many different forms depending on the application and the system constraints. This section discusses the selected and alternative approaches for transmitting beacon tones and scheduling data tracks. It then describes the algorithms used to schedule these beacon and data tracks, which are used in the simulations in the next section.

Beacon Tracks

In order to convey its health status and request time on the network, each spacecraft transmits beacon tones down to the ground. Each spacecraft is pre-scheduled n beacon tracks per week during which a ground antenna and receiver are actively listening for the spacecraft's beacon tones. It is assumed that these scheduled tracks are uploaded to the spacecraft in advance. We plan to use dedicated smaller, auxiliary antennas for receiving beacon tones, such as the 21-m Morehead DSS-17 antenna (as opposed to the 34-m and 70-m antennas). Because these antennas' schedules are not loaded down with regular downlink tracks, it is assumed pre-scheduled beacon tracks for many weeks (or months) could be uploaded to the spacecraft at once. During these scheduled beacon track times, a DSN beacon antenna will be actively tracking a specific spacecraft listening for a beacon. By providing the spacecraft with the pre-scheduled track times, it only needs to beacon during the duration of the track to assume its status was heard. Power-constrained spacecraft, such as SmallSats, want to minimize the amount of time spent using their transmitters.

An alternative method is when the spacecraft is not given *a priori* knowledge of when the beacon tracks are scheduled and the spacecraft beacons for a longer duration, such as for 36 hours for New Horizons (Wyatt et al. 2016). The network guarantees that if the spacecraft transmits within that 36-hr block, then it will be heard on the ground. This worked well for New Horizons because it was not as power constrained compared to solar-powered SmallSats and the antennas used for receiving its beacons were the standard 34-m and 70-m antennas at each main DSN complex, whose schedules change on a week-by-week basis. Because our simulations assume the use of auxiliary antennas for beacon reception, we chose to use the former method for which beacon tracks are agreed upon *a priori*.

Data Tracks

Once the spacecraft beacons to the ground requesting a communications track on the DSN, it needs to know the start/stop time of its scheduled downlink track. One possible method is that the ground uplinks an acknowledgment (ACK) packet with the scheduled track (or if the request was dropped) using a very low rate service. The issue with this approach is that it would either require the use of the 34-m and 70-m antennas for the uplink ACK track or very high power transmitters installed at the auxiliary smaller beacon antennas to close the link. In either case, the uplink tracks for acknowledgments would either have to be scheduled on demand into any remaining gaps in the antennas' schedules or be reserved in the master schedule when it was created weeks ahead of time.

Like the beacon tracks discussed above, an alternative approach is to agree *a priori* on possible downlink data tracks that the spacecraft *could* use. During the standard DSN scheduling process, these data tracks would be scheduled into the DSN master schedule as reserved but not allocated. They are chosen such that multiple demand-access spacecraft are in view during the track time (i.e., a "blocked" or "shared" schedule track) so that it can be used by any of the spacecraft in view. When a spacecraft sends a schedule request beacon tone to the ground, it assumes that it will be granted one of the next tracks depending on the online scheduling policy. It then transmits during the data track assuming it was granted time on the network. If the spacecraft's request for the data track were fulfilled, then the ground will receive the spacecraft's downlink and will also uplink an ACK during the data track. If the spacecraft does not hear an ACK during its shared data track, it can assume the request was not fulfilled and its data transmission was not received by the ground. If the data have not yet become stale, the spacecraft can re-beacon to the ground during the next available beacon pass for a new request for a data track, and the process repeats.

For the simulations presented in this paper, we chose to use the second approach because it does not require separate ACK tracks to be either pre-scheduled or scheduled in real time into the (very few) open gaps in the master schedule. Furthermore, the request–ACK mechanism works better for spacecraft for which round trip light-travel time is negligible, which is not generally the case with deep space missions.

In our simulations, we use the following scheduling policy. When a spacecraft transmits a downlink request beacon tone, its onboard scheduler schedules the next two shared data tracks (in order of track start time) after the end of the beacon track. The first track acts as the "primary" track, and the second track acts as the "backup" track. The spacecraft assumes that the DSN will schedule its request into either of those two tracks. The spacecraft will first transmit during the primary track. If the spacecraft hears an ACK during the data track from the ground, it knows its data downlink was received and will delete its backup track. If the spacecraft does not hear an ACK during the primary data track from the ground, then it will downlink the same data again during the backup track. If the spacecraft receives an uplink ACK during the backup pass, then it assumes the data were received on the ground. If it does not receive an uplink ACK, then it can assume its schedule request was dropped (due to those tracks already reserved by other demand access users). If the data are not stale by this point, the spacecraft can use its next beacon track to request a data track on the network again.

From the ground perspective, for this initial investigation into demand access, we assume a strictly first-come-first-

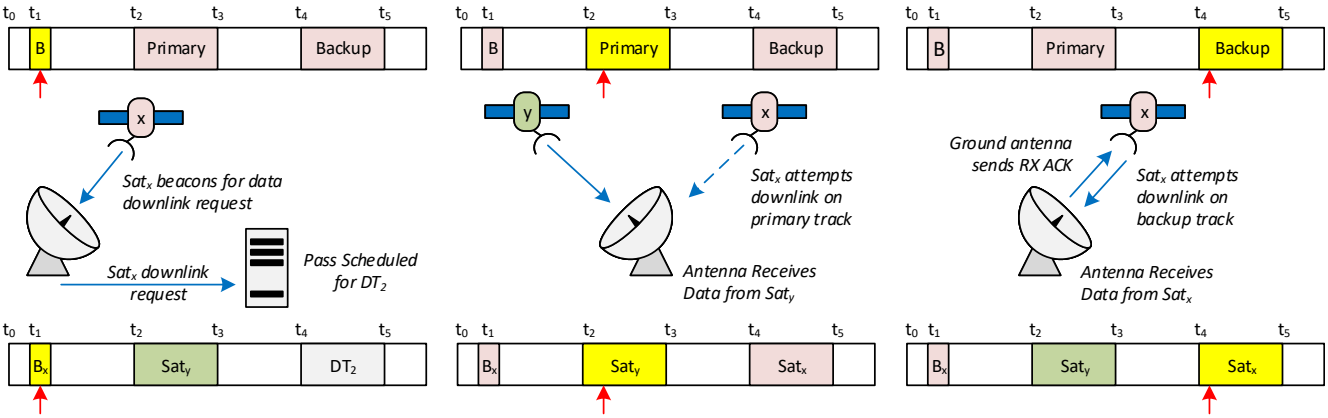


Figure 1: Example concept of operations for when a spacecraft is allocated its backup track.

serve (FCFS) online scheduling policy for allocating the shared data tracks (i.e., the order in which schedule request beacons are received is the order in which their requests are scheduled). When the DSN receives a beacon request, it will try to first schedule the request into the spacecraft's primary track and, if already allocated, into the spacecraft's backup track. If both tracks are full, the request is dropped. If the spacecraft was allocated a data track, then it is blacklisted from requesting another track on the network until the end of its allocated data track. If its request was dropped, the spacecraft is blacklisted from requesting another track until the end of its backup track.

An example scenario for which the spacecraft is scheduled for its backup pass is illustrated in Figure 1.

Shared Data Track Generation

In the previous section, we proposed the use of shared data tracks for which any of the blocked spacecraft assigned to that track can use it (if more than one chooses to use a specific track, then only one spacecraft transmission will be received). With enough shared data tracks reserved in a schedule week to meet each user's individual expected demand, shared data tracks give spacecraft more options on when to downlink. As an example, let us assume we have seven users. If each user required one track per week and the shared track could be shared by all seven users, then all seven shared tracks would be reserved in the schedule and could be used by any of the seven users. If those tracks are spread evenly throughout the week, then each user has a chance to downlink any day of the week. If we schedule each user individually as is currently done, each user would be given one track per week meaning it could be up to seven days later that a downlink would occur after a data collection. Without increasing the number of hours reserved in a schedule week, the duration between the data collection and the start of the downlink can be significantly decreased.

To identify track times that can be shared among multiple users, we employ the blocking algorithm used in Hackett, Johnston, and Bilén (2018). For each schedule week, we task the blocking algorithm to find all time windows when a grouping of spacecraft are in view (above the horizon) at the same time for each antenna. We do this for all combinations of n users from one through n. Each resulting spacecraft grouping with their mutual time windows with each antenna is considered a block. The output of the blocking algorithm is a collection of all "potential" blocks. We use a post-processing algorithm to choose the "best" grouping of blocks to use for each schedule week. The best grouping of blocks is one in which there is a small amount of user groups (with a large number of users in each group) with plenty of available track time. This ensures the maximum flexibility for each user. Figure 2 illustrates the post-processing algorithm for choosing the best grouping of blocks.

Given a minimum duration required for a shared track (e.g., 4 hours), each block is pruned of any time windows with duration shorter than the minimum duration—for users that are clustered closely together, the shared track minimum duration can be higher than for those spread out more uniformly. Next, the list is pruned of any "redundant subset" blocks: if the set of users in a block is a subset of another block's user set, the number of tracks per antenna for the subset and superset blocks are equal, and the number of antennas supporting each user set is the same, then the subset user block is eliminated.

Next, we find the grouping of blocks that will meet all of our criteria with the minimum number of blocks in the grouping necessary. For a grouping of n blocks, the union of all of the user sets across all n blocks must contain all of the demand access users under study. To make scheduling shared tracks easier, we require that each user can only be in one block's user set in the block group. If a user is in multiple block user sets in the block group, it is deleted from the largest group in which it is a member until it is only found once among all user sets that make up a block group. This helps to equalize the number of users per block. The resulting list of combinations of n blocks that meet the previous criteria are then ordered by:

1. the block group whose minimum number of users of a single member block is largest,

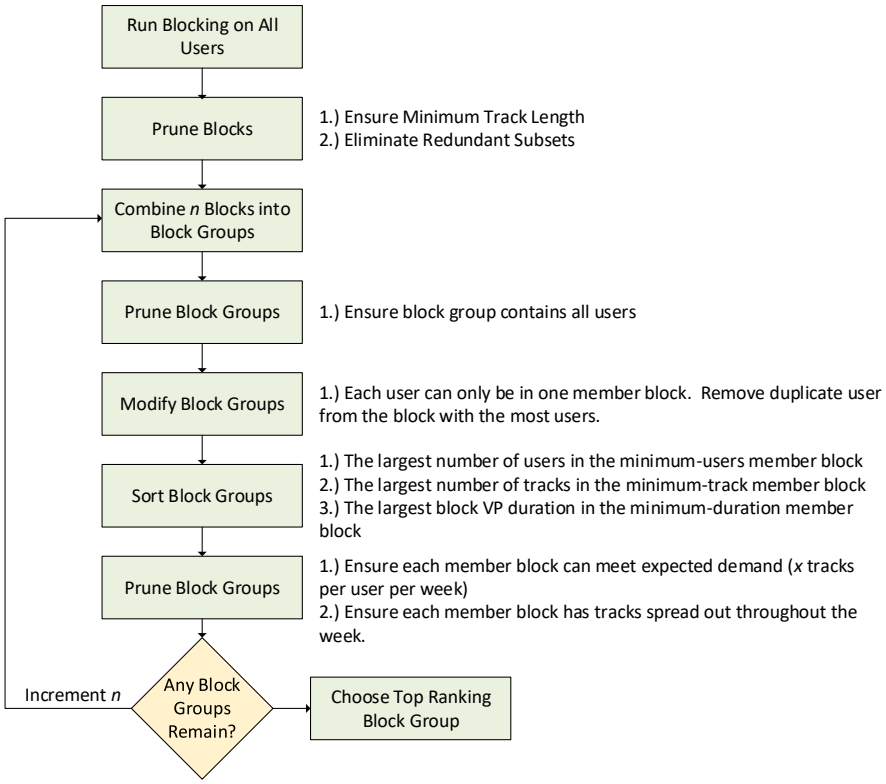


Figure 2: Post-processing algorithm block diagram for choosing the best grouping of blocks for shared data track generation.

2. then by the block group whose minimum number of tracks meeting the required minimum duration of a single member block (across all antennas) is largest, and
3. then by the block group whose minimum total duration across all time windows of a single member block (across all antennas) is largest.

This ordering makes sure that the top-ranked block groups have the most users per block (meaning more flexibility for each user) and a large number of potential time windows (meaning more flexibility by the DSE to schedule these time windows into the master schedule as reserved tracks). Finally, in rank order, the block groups are evaluated to ensure that each member group can meet each individual user's expected demand and that the view windows of each member block are sufficiently spread out through the week. For the former criterion, if a member block has seven users and each user requires two 4-hr tracks per week, then the member block must have 14 4-hr tracks available. For the latter criterion, requiring the end of the last time window of each member block must be at least 5 days later than the start of the first time window will ensure there are time windows at the beginning and end of the schedule week. The "best" block grouping is the first block in the ordered listing that meets these two criteria. If there are no block groups with *n* member blocks that meet all of the necessary criteria, then this process is repeated with all block groups with $n + 1$ member blocks.

Pre-scheduling Shared Data Tracks and Beacon Tracks

The output of the post-processing algorithm for the shared tracks is a group of blocks for each schedule week, for which each block contains all time window intervals in which the block users are in view of each antenna. Because of the relationship of the motion of the Earth around the Sun and the trajectories of each user, the group of blocks will not necessarily be the same for each schedule week.

Now, the group of blocks are scheduled into the master schedule to become reserved shared data tracks. The total number of tracks required for block i in the block group, $\lceil N_{\mathrm{DT},i} \rceil$, is the summation of the number of tracks required to be reserved per user and rounded up to the nearest integer—schedules are created on a week-by-week basis, so there must be an integer number of tracks per week. The scheduler tries to schedule these tracks somewhat evenly throughout the week. It does this by splitting the schedule week into $\lceil N_{\mathrm{DT},i} \rceil$ equally sized time intervals. On each scheduling round, it tries to find the track with the earliest start time in the time interval that meets the block's time window mask. After $\lceil N_{\mathrm{DT},i} \rceil$ rounds, if the number of tracks scheduled is not $\lceil N_{\mathrm{DT},i} \rceil$, then the process starts again but with half as many equally sized time intervals (rounded up to the nearest integer). This process will con-

Table 1: Chosen NEAs for each number of SmallSats in simulation.

# of Users	User Group
1	Daedalus
2	Zeus, Sisyphus
3	Midas, Zeus, Phaethon
4	Hephaistos, Heracles, Toutatis, Daedalus
5	Toutatis, Heracles, Daedalus, Phaethon, Toro
6	Zeus, Hephaistos, Daedalus, Midas, Heracles, Toro
7	Heracles, Wilson-Harrington, Toutatis, Phaethon, Sisyphus, Toro, Hephaistos
8	Wilson-Harrington, Midas, Phaethon, Hephaistos, Toro, Heracles, Zeus, Sisyphus
9	Wilson-Harrington, Zeus, Heracles, Toro, Midas, Daedalus, Sisyphus, Phaethon, Toutatis
10	Toro, Daedalus, Sisyphus, Midas, Hephaistos, Phaethon, Wilson-Harrington, Toutatis, Heracles, Zeus

tinue until $\lceil N_{\mathrm{DT},i} \rceil$ have been scheduled or the number of equally sized time intervals is one and another track could not be scheduled.

Each block track is scheduled in a round-robin order: one track for each block is attempted to be scheduled before a second track for the first block is attempted to be scheduled. During each round-robin round, the blocks are ordered by the least number of successfully scheduled tracks and then randomly shuffled (uniform distribution) if equal—this helps ensure "fairness" when scheduling the blocks.

The beacon tracks are scheduled after the shared data tracks as the potential time windows for the beacons are not as constrained as the blocks—the beacon tracks are exclusive to each user, so they are not constrained to when multiple users are in view of a ground station simultaneously. Given the requested number of beacon tracks per user u per week, $\lceil N_{\mathrm{BT},u} \rceil$, the scheduler splits the schedule week into $\lceil N_{\mathrm{BT},u} \rceil$ equally sized time intervals. On each scheduling round, it tries to find the track with the earliest start time in that time interval. If it cannot find a track, it skips this time interval and moves on. As with the data tracks, each beacon track is scheduled in a round-robin order: one beacon track for each user is attempted to be scheduled before a second beacon track for the first user is attempted to be scheduled. During each round-robin round, the order of the users are randomly shuffled (uniform distribution), which helps to ensure "fairness".

With the pre-scheduling of the data and beacon tracks complete, the demand access users are ready to operate.

NEA Autonomous SmallSat Explorer Fleet Simulation Campaign

One specific type of mission class that this beacon tone–based demand-access scheduling paradigm supports very well is the burgeoning class of deep-space SmallSats. Due to their small size, SmallSats are often power limited, which limits their data rate and track duration. Despite their small physical size, SmallSats can still collect a large amount of important science data. This makes them more apt to adopt many artificial intelligence algorithms for onboard filtering of the raw science data, in addition to having a higher level of onboard autonomy and control. SmallSats generally have financial budgets orders of magnitude less than flagship and other large missions, so funding a mission representative to participate in the current DSN scheduling process is generally not feasible. A demand access scheduling paradigm provides a more financially viable method for SmallSats to be deployed *en masse*.

Simulation Setup

An interesting SmallSat mission concept that we investigate in this paper is the deployment of autonomous deep-space SmallSats to explore and monitor near-Earth asteroids (NEAs) performing event-based science. NEAs could provide necessary raw materials for future in-space development and re-fueling missions. Due to their proximity to Earth, they also pose potential collision hazards and are crucial to monitor. Each SmallSat is assumed to be equipped with autonomous navigation capabilities and a simple set of instruments, such as cameras and spectrometers. We model the data collection of these SmallSats as a stochastic process to better reflect the collection of data in an event-driven science realm. Specifically, the science collection process is modeled as a Poisson process with exponential interarrival times with a mean of 1 week (i.e., on average, every 7 days we will instantaneously collect 1 unit of data). Additionally, the health status of the spacecraft is modeled as a Poisson process with exponential interarrival times with a mean of 30 days (i.e., on average, every 30 days there will be some health event on the spacecraft that will require attention and will store 1 unit of data). The spacecraft onboard storage is assumed to hold 4 units of data and is initialized to be empty at the start of the simulation. If a data collection occurs while the storage buffer is full, the data are dropped and not added to the buffer. Given SmallSat power constraints, we limit each downlink track to 4 hours. In this track duration, we assume that one data collection can be downlinked. With a representative downlink information rate of 2 kbps, this means a data collection is 28.8 Mb and the onboard storage capacity is 115.2 Mb.

NASA's OSIRIS-REx mission is visiting 101955 Bennu (1999 RQ36), an asteroid in the Apollo class of NEAs. In our simulation, we focus on sending one SmallSat to each of the ten largest (in diameter) Apollo NEAs with names by the International Astronomical Union (IAU): 1866 Sisyphus (1972 XA), 2212 Hephaistos (1978 SB), 4179 Toutatis (1989 AC), 5731 Zeus (1988 VP4), 3200 Phaethon (1983

TB), 5143 Heracles (1991 VL), 4015 Wilson-Harrington (1979 VA), 1864 Daedalus (1971 FA), 1981 Midas (1973 EA), and 1685 Toro (1948 OA).

Because the performance of the demand access scheduling system is dependent on the number of users in the system, simulations are run with the number of users between 1 and 10. Table 1 shows the chosen candidate NEA groups using a uniform random sampling. Four antennas at each main complex are assumed to be available for scheduling shared data tracks: DSS-{14,24,25,26} in Goldstone, CA, USA; DSS-{34,35,36,43} in Canberra, Australia; and DSS-{54,55,63,65} in Madrid, Spain. The 21-m Morehead antenna (DSS-17) is assumed to be available for scheduling beacon tracks.

Simulations are run with a 1-minute timestep over a 53-week period: ISO Week 1, 2018 through ISO Week 1, 2019, which provides a spatial diversity of the NEAs with respect to the pointing angles of the ground antennas (this, in turn, creates different block groups throughout the year). This also provides a sufficiently long time-horizon given the mean interarrival times of the data collections. We assume each SmallSat is in a steady-state orbit around its respective NEA; we directly use the ephemeris of the NEA as a proxy for the SmallSat's ephemeris. These ephemerides were acquired through JPL's Small-Body Database (Park 2019).

In addition to varying the number of spacecraft in each year-long simulation, simulations are run for {1.25, 1.5, 2.0} data tracks per user per week. 1.25 tracks per user per week essentially is reserving the number of tracks necessary to meet the mean data collection rate. Providing 1.5 and 2.0 tracks per week over-reserves the number of required tracks in order to provide a buffer against bursts of data collections by multiple users. For the beacon tracks, each user requests seven 1-hr beacon tracks per week (i.e., one 1-hr beacon track per day).

The simulation software used was a custom-built Python DSN network simulator created by the authors using a SimPy3 (Lünsdorf and Scherfke 2018) discrete-time framework and NAIF's SPICE Toolkit (Acton 2018; Annex 2018) for ephemeris calculations. More on this simulator is described elsewhere. For the blocking analysis, the blocking software developed in Hackett, Johnston, and Bilén (2018) was used. For each {number of users, tracks per user per week} configuration pair, between 50 and 60 simulations were run to provide a better distribution of results. A bug in the high computing cluster job scheduler used for these simulations inadvertently canceled a few simulations prematurely. In total, upwards of 1800 simulations were run for the results presented in the next section.

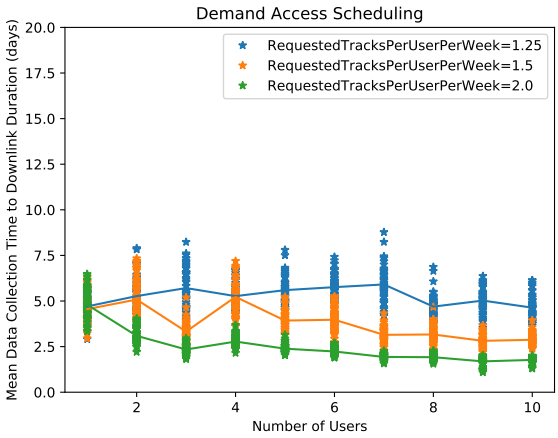
Simulation Results

Three important metrics for measuring the efficacy of our proposed beacon-tone demand access scheduling system are the mean duration between the time of data collection to the start of the downlink (referred to as the mean "data latency" for the remainder of this paper), the percentage of data collections that were dropped because the onboard storage buffer was already full, and the track utilization (the number of reserved data tracks that were actually allocated). The mean data latency provides an insight into the latency of returning science data and addressing spacecraft health issues. The lower the mean data latency, the quicker useful data were returned to mission engineers and scientists. Using a limited onboard storage capacity (as opposed to an infinitely large capacity) results in an upper limit in the data latency because dropped data collections are not included the calculation of the data latency. Because of this, the percentage of dropped collections provides context for the mean data latency values. A low percentage of dropped collection and a low data latency means the spacecraft is able to capture all of its science events and downlink them to the ground in an expedient manner. Finally, the track utilization is directly related to the relationship between the reserved number of tracks per user per week and the expected data collections per week. Although a higher track utilization means more reserved tracks were actually allocated and used to downlink science, it is not necessarily "better". A higher track utilization also means that the network has less *excess capacity* to buffer against unexpected events or scaling up more users.

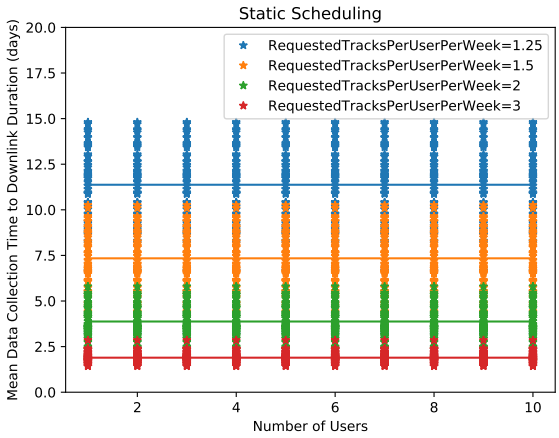
To compare our demand access system against the current static DSN scheduling system, a best-case scenario simulation was run with the same data collection and limited storage capacity assumptions as the demand access simulations. In this simulation, each user is allocated a 4-hr data track at a fixed interval of $7/\mathrm{tracks\_per\_week}$ days. In this case, the track utilization is the percentage of data tracks for which the spacecraft actually downlinked new science data (i.e., at the start of the data track, there were data in the spacecraft buffer). Because each user is independent of each other, the results hold for any number of users in the network.

Figure 3 shows the results of the simulations for each of these three metrics. For each plot, each individual data point is that metric measured after the 53-week simulation period. The solid line represents the mean value across all simulations with the same {number of users, tracks per user} configuration pair.

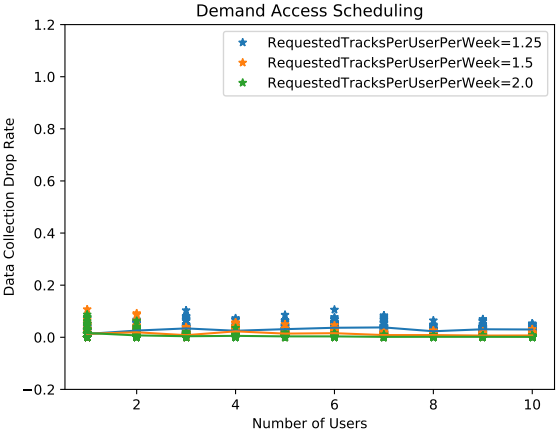
Data Latency and Data Drop Rate Figures 3a and 3b show the mean data latency using the demand access system and the current static scheduling system, respectively. For the same number of tracks per user per week reserved, as the number of users increases, we see that mean data latency becomes roughly half of the value using the static scheduling approach. This decreased latency is the result of the fact that, as more shared users are included in a block, each user has more options to choose from for downlinking. In contrast, with the static scheduling approach, a user only can choose from n tracks per week regardless of the number of users. Figures 3c and 3d show that, for the same number of tracks per user per week, the demand access approach has a lower or the same data drop percentage. Furthermore, the demand access approach with 1.5 tracks per user per week outperforms the static scheduling approach with 2.0 tracks per user per week meaning that our demand access approach was both able to decrease the mean data latency while maintaining a low science collection drop rate but also decrease the amount of total tracks required in the schedule (freeing up more tracks for more users). Comparing the variance be-



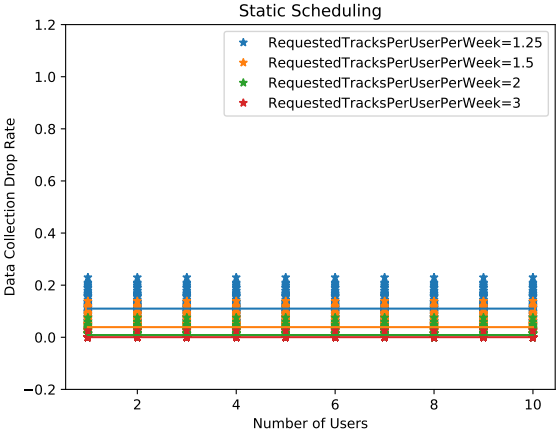
(a)



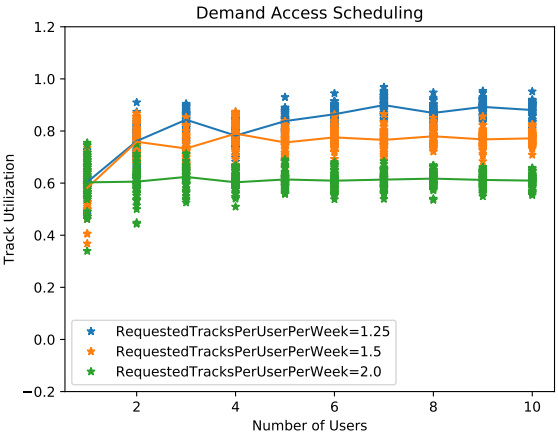
(b)



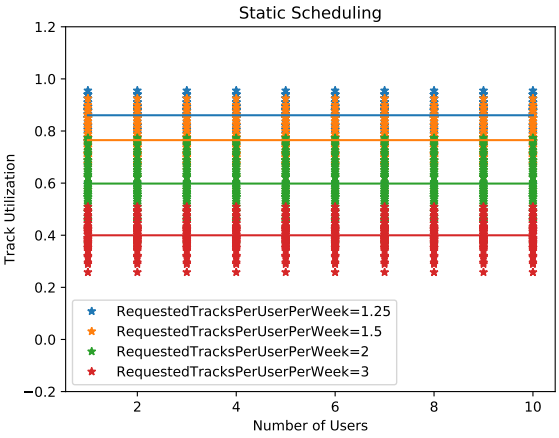
(c)



(d)



(e)



(f)

Figure 3: The (a,b) mean data collection time to downlink duration, (c,d) data collection drop rate, and (e,f) track utilization for a 53-week simulation period for both demand access scheduling and static scheduling, respectively, of an NEA autonomous SmallSat explorer fleet mission concept.

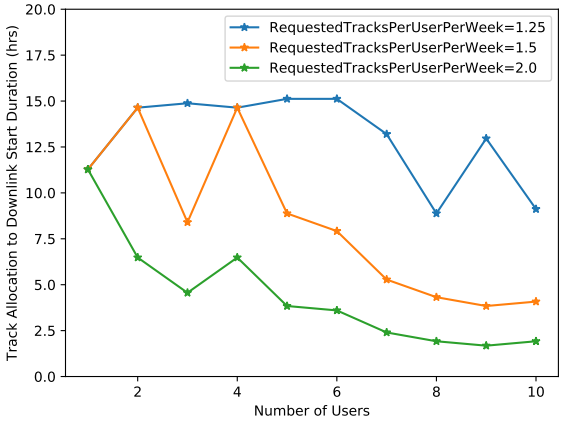


Figure 4: The 5th percentile duration between the allocation time and start of the track as a function of the number of users and number of tracks per user per week from the demand access simulations

tween simulations, we also see that the demand access approach results in a lower mean data latency variance than the static simulation decreasing the data return latency jitter figure of merit.

Data Track Utilization Looking at Figures 3e and 3f, we get the expected result that the track utilization is roughly the same for both the demand access and the static scheduling approaches as this is directly related to the number of tracks per user per week that were reserved. The reason that the track utilization is lower for a smaller number of users for demand access scheduling is because the number of tracks reserved per block per week must be an integer number. For example, if there is only 1 user in a block and it requests 1.5 tracks per week, then the number of tracks reserved will be $\lceil 1.5 \rceil = 2$ tracks. As the number of users increases per block, this rounding artifact goes away.

Spacecraft Health Status In the demand access system, the spacecraft can provide a simple health and status update every day through the use of its scheduled beacon passes. Although full telemetry is not downlinked, New Horizons showed that a beacon tone message is sufficient for general health information. In the static scheduling method, the spacecraft's health and status are only provided whenever it has its scheduled data downlink track. If the mission wanted more frequent health and status updates, it would have to schedule full telemetry tracks on the oversubscribed 34-m and 70-m antennas. In contrast, the beacon tracks can be scheduled on auxiliary, smaller antennas across the world freeing up the main complexes solely for science data tracks. By using the beacon tones, the demand access system allows mission scientists to get more frequent (albeit less detailed) health status updates.

Opportunistic Secondary Users Figures 3a and 3b show that, as the number of tracks per user per week increase, the mean data latency decreases but this is at the cost of reserving more total tracks that do not end up getting allocated. Two methods that can take advantage of these reserved but unallocated tracks are either pre-scheduling other secondary non-demand access users to these tracks and, if the track ends up getting allocated by the primary demand access user, then the secondary user is preempted. With higher over-reservation of track time, the chances of preemption decrease. For example, with 2 tracks per week reserved for demand access users, there is still a 40% chance that the secondary user will be given the track and not be preempted.

A second method is that secondary users are scheduled for reserved but unallocated tracks in a just-in-time fashion. Based on the dynamics of the track utilization and beacon schedule, the network provider can provide secondary users with a duration before the start of a track such that it is highly unlikely (e.g., 5%) that a demand access user will be allocated the track. Figure 4 shows the 5th percentile duration between the allocation time and start of the track as a function of the number of users and number of tracks per user per week from the demand access simulations. For example, for 10 users and 1.5 tracks per week, there is a 95% chance that, if the reserved demand access data track is not allocated by four hours before start of the data track, it will not be allocated at all. At this point, a secondary user could be scheduled just-in-time with a very high likelihood of not getting preempted at the last minute.

Both of these methods can increase the total network utilization and the amount of supported users without affecting the primary demand access user's data latency.

Future Work and Conclusions

We are still in the early stages of analyzing the dynamics and optimizing the use of a demand access scheduling system. This paper presents a preliminary case study, and there are many areas to explore in future work. For example, it would be important to investigate if the number of total data tracks reserved for the demand access users is fixed (e.g., one antenna's worth of schedule time), how does the performance degrade as more users are added to the network? From the blocking perspective, the block group selection process could incorporate load balancing if user spacecraft have different rates of data collection.

In this paper, we detail the algorithms for a beacon tone–based demand access scheduling system and simulate its behavior for an NEA autonomous SmallSat explorer fleet mission concept and compare it to the static scheduling system that the DSN currently uses. Leveraging the spacecraft blocking techniques, we find that the demand access system was able to decrease the mean data latency by about 50% for the same number of schedule reservations compared to the static scheduling system. Furthermore, with 25% less tracks reserved, the demand access system was still able to provide a lower mean data latency with a very low data collection drop rate. By using frequent-but-short beacon tracks, missions are able to receive basic health and status updates more frequently than with the static scheduling method. To further increase the number of users supported and tie in

legacy users, two different primary–secondary user architectures are proposed for utilizing the reserved but unallocated tracks. Furthermore, because the beacon tracks and shared data tracks were pre-scheduled, this demand access system would integrate well with the current static scheduling system to support both new and legacy users. From the simulation results in this paper, we can see that demand access is a promising approach that could enable event-based science missions in the future.

Acknowledgments

This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. This work was also supported by a NASA Space Technology Research Fellowship (grant number NNX15AQ41H).

References

Acton, C. 2018. An overview of SPICE. Online. Available at `https://naif.jpl.nasa.gov/pub/naif/toolkit_docs/Tutorials/pdf/individual_docs/03_spice_overview.pdf`.

Annex, A. 2018. SpiceyPy: The NASA JPL NAIF SPICE toolkit wrapper written in Python. Online. Available at `https://github.com/AndrewAnnex/SpiceyPy`.

Bowman, A. F.; Chacos, A. A.; DeBoy, C. C.; Furrow, R. M.; and Whittenburg, K. E. 2004. New Horizons mission to Pluto/Charon: Reducing costs of a long-duration mission. In *55th International Astronautical Congress*, 1–8.

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 Conference*, 1–12.

DeCoste, D.; Finley, S. G.; Hotz, H. B.; Lanyi, G. E.; Schlutsmeyer, A. P.; Sherwood, R. L.; Sue, M. K.; Szijjarto, J.; and Wyatt, E. J. 2004. Beacon monitor operations experiment DS1 technology validation report. Jet Propulsion Laboratory, California Insitute of Technology.

Gitlin, T. A., and Horne, W. 2012. The NASA Space Network demand access system (DAS). In *SpaceOps 2012 Conference*, 1–11.

Hackett, T. M.; Johnston, M. D.; and Bilén, S. G. 2018. Spacecraft block scheduling for NASA's Deep Space Network. In *SpaceOps 2018 Conference*, 1–14.

Imbriale, W. A. 2002. Large antennas of the Deep Space Network. In Yuen, J. H., ed., *Deep-space Communications and Navigation Series*, number Monograph 4. Jet Propulsion Laboratory. 1–298.

Israel, D. J.; Roberts, C.; Morgenstern, R. M.; Gao, J.; and Tai, W. S. 2018. Space mobile network concepts for missions beyond low Earth orbit. In *SpaceOps 2018 Conference*, 1–11.

Johnston, M. D., and Wyatt, E. J. 2017. AI and autonomy initiatives for NASA's Deep Space Network (DSN). In *AI in the Oceans and Space Workshop, International Joint Conference on Artificial Intelligence*, 1–6.

Lauretta, D. S.; DellaGiustina, D. N.; Bennett, C. A.; Golish, D. R.; and et. al. 2019. The unexpected surface of asteroid (101955) Bennu. *Nature* 568(7750):55–60.

Lünsdorf, O., and Scherfke, S. 2018. Simpy: Discrete event simulation for Python. Online. Available at `https://simpy.readthedocs.io/en/latest/index.html`.

Malphrus, B.; Kruth, J.; Pham, T.; and Wyatt, E. J. 2018. Enabling university-operated tracking and communications for deep space SmallSat missions. In *CubeSat Developers' Workshop*, 1–21.

Martinez, A. 2018. SmallSats supporting deep space human exploration. In *Colloquium on the Radioelectric Spectrum and Its Scientific and Experimental Use in Space*, 1–22.

Nelson, J. 2018. Deep Space Network. Online. Available at `http://deepspace.jpl.nasa.gov`.

Park, R. 2019. JPL small-body database. Online. Available at `https://ssd.jpl.nasa.gov/sbdb.cgi`.

Pinover, K.; Johnston, M. D.; and Lee, C. 2017. Optimizing SmallSat scheduling for NASA's Deep Space Network. In *International Workshop on Planning and Scheduling for Space (IWPSS)*, 124–132.

Shaw, H.; King, J.; Israel, D. J.; Kang, J.; Roberts, C.; and Burke, J. 2018. Space mobile network user demonstration satellite (SUDS) for a practical on-orbit demonstration of user initiated services. In *SpaceOps 2018 Conference*, 1–9.

Wyatt, E. J.; Foster, M.; Schlutsmeyer, A.; Sherwood, R.; and Sue, M. 1997. Optimizing SmallSat scheduling for NASA's Deep Space Network. In *International Symposium on Artificial Intelligence, Robotics, and Automation in Space*, 1–6.

Wyatt, E. J.; Hotz, H.; Sherwood, R.; Szijjarto, J.; and Sue, M. 1998. Beacon monitor operations on the Deep Space One mission. In *5th International Symposium on Space Mission Operations and Ground Data Systems*, 1–10.

Wyatt, E. J.; Sherwood, R.; Sue, M.; and Szijjarto, J. 1999. Flight validation of on-demand operations: The Deep Space One beacon monitor operations experiment. In *International Symposium on Artificial Intelligence, Robotics, and Automation in Space*, 1–6.

Wyatt, E. J.; Abraham, D.; Johnston, M.; Bowman, A.; and Malphrus, B. 2016. Emerging techniques for deep space CubeSat operations. In *5th Interplanetary CubeSat Workshop*, 1–17.