Cloud-Based Demodulation and Data Distribution of a Satellite Downlink

Adam Gannon
NASA Glenn Research Center
Cleveland, OH
adam.gannon@nasa.gov

Joseph Downey
NASA Glenn Research Center
Cleveland, OH
joseph.a.downey@nasa.gov

Cedric Priscal
KBR Wyle Services
Mountain View, CA
cedric.priscal@nasa.gov

Alejandro Salas
Millennium Engineering and Integration
Mountain View, CA
alejandro.j.salas@nasa.gov

Marcus Murbach
NASA Ames Research Center
Mountain View, CA
marcus.s.murbach@nasa.gov

***Abstract*—Ground station networks connected to the cloud allow space missions global communications coverage without the need to operate their own infrastructure. In this work, we describe the communications architecture for the TechEdSat-13 mission, which performed the first in-space characterization of a neuromorphic processor. The mission utilizes a commercial provider for S-band downlinks. A suite of cloud services and open-source software such as GNU Radio are leveraged to demodulate signals received by an AWS ground station during passes with TechEdSat-13 and store recovered data. Once a pass is scheduled, the entire process takes place without human intervention. On-orbit results from the past year of operations are presented, demonstrating the advantages of this approach over traditional operator-owned ground stations with local processing. Use of software-defined radio makes possible custom signal processing. The homogeneity of apertures and their interfaces to the cloud simplifies scaling across many sites. This abundance of candidate links lays the groundwork for intelligent scheduling agents to optimize pass selection across several factors, automatically recover from failed contacts, and gather metrics to learn from past performance.**

***Index Terms*—Ground Station as a Service, software-defined radio, cloud demodulation, autonomous operations, TechEdSat**

I. Introduction

Modeled upon the transformative “as a Service” model, the Ground Station as a Service concept allows mission operators to expand communications coverage of their spacecraft to global scale with zero capital expenditure. Coupling of these ground station providers with cloud services enables greater autonomy in the process of downlinking spacecraft data: from scheduling, to demodulation, data processing, and distribution to end users. Such infrastructure has been leveraged to improve the efficiency of NASA Earth Science data gathering [1] and satellite operations [2]. Cloud-based services are being explored for rapid dissemination of satellite data in military contexts [3] and will feature heavily in future operations of NASA’s Near Space Network [4].

This framework was recently used by the Technology Education Satellite (TechEdSat) program for downlinks from a technology demonstration CubeSat. In this work, we describe in detail the TechEdSat framework for signal reception, demodulation, and data delivery. With the help of cloud resources this process was fully-automated and ran without monitoring from mission personnel. Use of open-source software for signal processing reduced operations cost. Using automation and cloud computing we can reduce the latency between data collection, processing, and delivery to interested parties. Increased automation is important to keep operations costs under control in a future where large numbers of low-cost spacecraft are increasingly performing Decadal-class science [5]. This capability can be leveraged - including on future flights - to automate scheduling and make optimal decisions in a complex environment.

II. Spacecraft Mission

The TechEdSat program based at NASA Ames Research Center aims to prove out advanced technologies with impact to future NASA missions through rapid iteration across flight demonstrations [6]. TechEdSat 13 (TES-13) is a 3U CubeSat launched on January 13, 2022 as a secondary payload onboard a Virgin Orbit flight. The mission performed environmental characterization and performance testing of a neuromorphic computer chip hosting a spiking neural network with future space communications applications. This processing onboard an Intel Loihi represents the first on-orbit demonstration of a processor with a neuromorphic architecture in space [7], [8].

Nominal mission communications are through an Iridium 9602N Short Burst Data modem which intermittently delivers packets up to 270 bytes (to user spacecraft) or 340 bytes (from

This manuscript is a joint work of employees of the National Aeronautics and Space Administration, employees of Millennium Integration & Engineering Company under contract #80ARC021D0001, and employees of KBR Wyle Services, LLC under contract #80ARC020D0010.

user spacecraft). The mission's main use of this link is low-rate telemetry and commanding. Though small data files can be transferred via this link, the primary method of data transfer is through the much higher-rate S-band (2280 MHz) radio.

A. S-Band Radio

The S-band radio on TES-13 is mainly comprised of a Digi ConnectCore embedded processor and Ettus B205mini software-defined radio (SDR). The radio implements a QPSK transmitter following standards by the Consultative Committee for Space Data Systems (CCSDS). Currently, the S-band link is a unidirectional downlink only. The radio transmits through a patch antenna on the +Y (ram-facing) face of the satellite with a half-power beamwidth of 90°.

In advance of a downlink pass, telemetry and experiment data files are transferred into a buffer on the radio. Commands sent over Iridium schedule the transmission by providing time periods to turn on the radio. Due to orbital dynamics and availability of the Iridium satellites the latency of this command can be up to several hours. As order of practice we send this transmit command 24 hours in advance of a downlink. After enabling its transmitter a few seconds before the scheduled start time of a pass, the radio loops over all files in its transmit buffer until the end of the pass.

III. Cloud Framework

For this work, we selected Amazon Web Services (AWS) Ground Station as a service provider. Since 2019 AWS has incorporated antenna sites with data connectivity into the company's cloud infrastructure. Currently 10 operational sites provide global coverage. Data from each site can be processed within its corresponding AWS region or delivered across regions. Antenna time is charged by the minute with rates based on bandwidth and monthly usage. We describe our framework using AWS services and resources, though we note this approach is broadly applicable to similar providers.

A. Service Scheduling

Use of AWS Ground Station begins with creation of a mission profile which describes parameters for how contacts are executed (Table I). The mission profile is agnostic to a specific satellite or ground station site, and will be used for future TechEdSat flights with the same radio configuration. For each satellite the AWS system regularly pulls updated two-line elements (TLEs) published by the Combined Space Operations Center (CSpOC). These TLEs are used to predict potential contacts where a satellite is within line-of-sight of a particular ground station. AWS provides an application programming interface (API) through which contacts can be listed and filtered by ground station availability. In submitting a scheduling request an operator provides the satellite, ground station, and mission profile along with the requested start/stop times of the pass. Several days before the planned pass, the TechEdSat team uses a utility built around the API to submit service requests. After manual scheduling all other steps in the demodulation process are automated.

Parameter	Value
Center Frequency	2280 MHz
Polarization	Right-Hand Circular
Capture Bandwidth	2.5 MHz
Autotrack	Preferred
Minimum Pass Duration	180 seconds
Pre/Post-Pass Notification	120 seconds

TABLE I: Selected parameters specified in mission profile for downlink from TES-13's S-band radio.

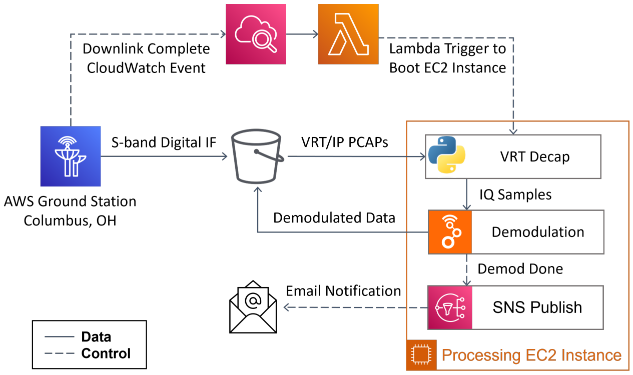


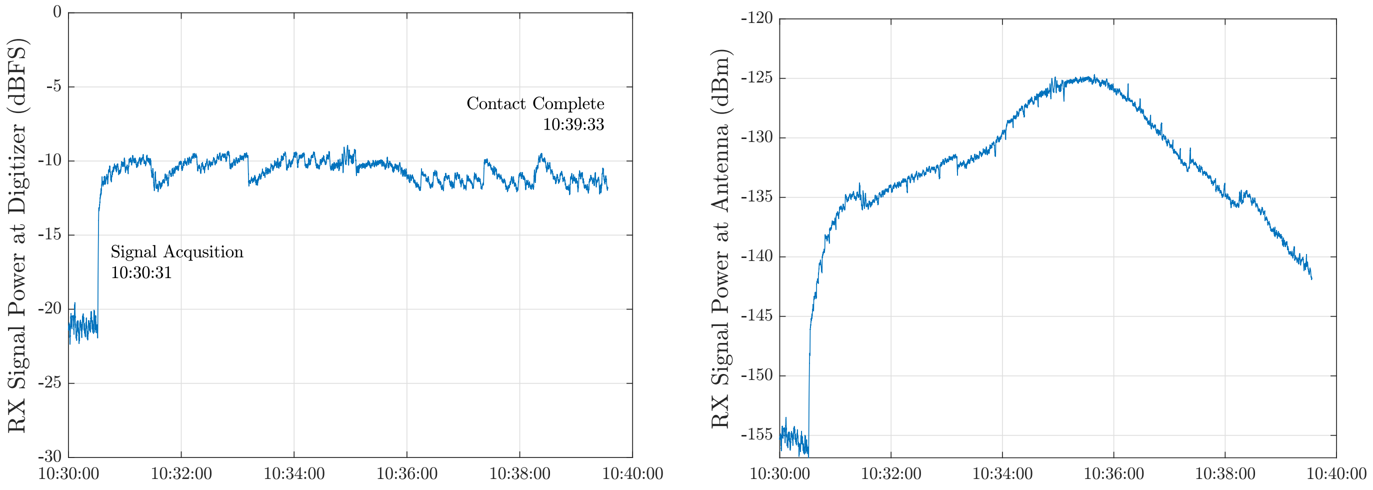
Fig. 1: Overview of services and their connections used to process downlinked signals within the AWS cloud. After initial scheduling, all steps occur without human involvement.

B. Signal Reception

S-band signals received by AWS Ground Station are digitized following the VITA Radio Transport (VRT) standard [9] and encapsulated in UDP/IP datagrams. Satellite operators can stream this traffic into an Elastic Compute Cloud (EC2) instance to process in real-time (the approach taken in [2]). Alternatively, the service can be configured to store packet captures of this traffic in a Simple Storage Service (S3) bucket for post analysis. We take the simpler post-analysis approach and begin processing of VRT-encoded digital intermediate frequency (IF) samples immediately after a pass completes.

C. Processing Automation

Fig. 1 outlines the framework used to automate processing and data distribution. A configurable period of time after the pass completes (Table I) a CloudWatch event triggers a Lambda function which boots an EC2 instance used for processing. Once booted, software pulls all packet capture files from their bucket into Elastic Block Storage attached to the instance. Section IV-A describes the processing to extract digital IF samples from packet captures and convert them into the format required by the GNU Radio framework. The software demodulator processes samples from the entire pass (Section IV-B) and recovered byte-aligned data is stored in S3. Upon completion of processing, Amazon Simple Notification Service (SNS) sends an email to TES-13 operators notifying them a volume of data is ready for download from S3. To save compute costs, the instance then shuts down until the next ground station pass.



(a)

(b)

Fig. 2: Signal power (a) after digitizer and (b) at input to ground station antenna illustrating metrics gathered from VRT packets. Data are from a July 20, 2022 pass with a 1 second moving average applied. All times in UTC.

IV. Signal Processing

As shown in Fig. 1, our EC2 instance uses several open-source frameworks to process the received signal.

A. VITA Radio Transport

The VRT standard handles two packet types relevant to our application: IF data packets of digitized samples and IF context packets containing metadata related to captured IF data. We use the `libvrt` library to parse IF context packets [10]. Table II shows the contents of a context packet for a TES-13 downlink. In this case, digital IF is captured as signed fixed-point complex values. Using this context, a Python script unpacks all samples from IF data packets and store in the GNU Radio format for complex samples (32-bit interleaved floats). The digitized signal power relative to the maximum analog/digital converter (ADC) amplitude is

$$P_{samp,n} = 20 \log_{10} \left(\frac{|s_n|}{2^{m-1} - 1} \right) \quad \text{[dBFS]} \tag{1}$$

where s_n is the nth complex-valued sample in the series represented by m fixed-point bits. Fig. 2a plots this value for a July pass. We notice the effect of the automatic gain control circuit in the receiver to provide a near-constant power signal.

While ADC values do not provide measurements in absolute terms, the reference input signal level $P_{ref,k}$ provided in the kth IF context packet can be used to calculate signal power at input of the ADC

$$P_{adc,n} = P_{samp,n} + P_{ref,k} \quad \text{[dBm]}. \tag{2}$$

Along with AWS-provided parameters of fixed gain in the receive chain G_{rx} and antenna directivity G_{ant} the received power at the input of the antenna is

$$P_{ant,n} = P_{adc,n} - G_{rx} - G_{ant} \quad \text{[dBm]}. \tag{3}$$

The Python script extracts each context packet and performs the calculation of signal level. Fig. 2b shows this value with the characteristic parabola of a ground station contact.

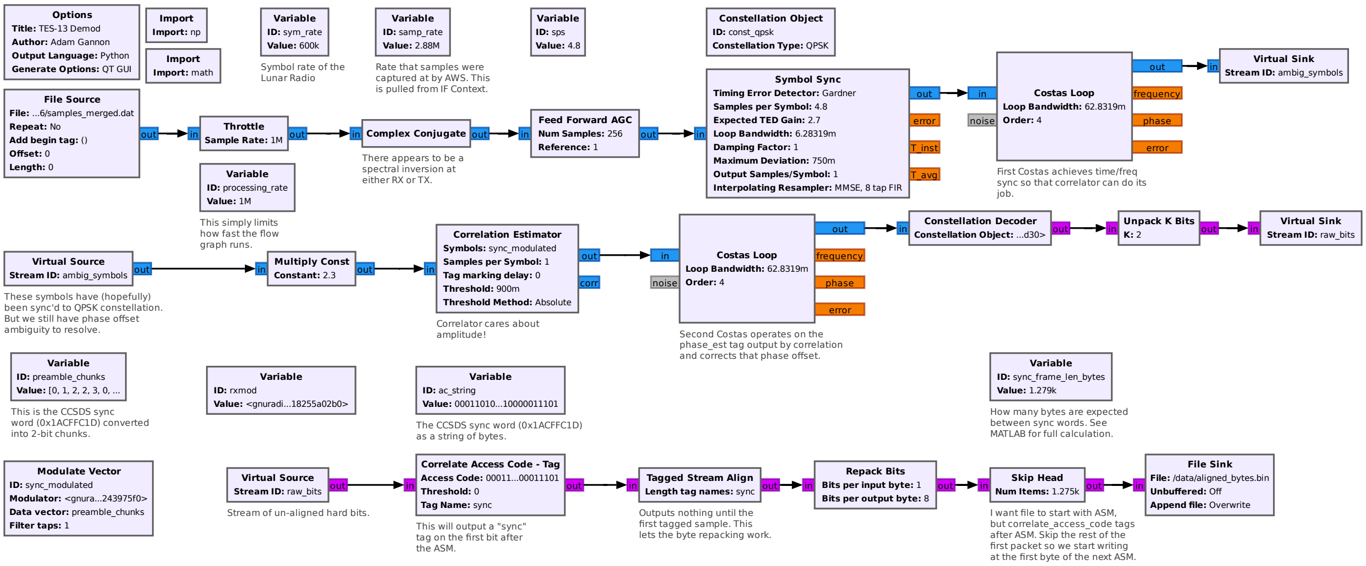
B. GNU Radio

As a result of the above processing, we have a large binary data file which contains extracted complex samples for the entire pass in GNU Radio format. The IF context packets (Table II) provide the rate at which samples were captured which is greater than or equal to the requested bandwidth. Dividing that by the symbol rate of the S-band radio (600 kbaud) provides the number of samples per symbol. Running on recorded data, the rate at which samples are processed is unrelated to sample rate and only limited by the processing power of the EC2 instance.

Software demodulation (Fig. 3) requires only standard signal processing blocks in GNU Radio 3.10. Gardner and Costas loops provide timing and carrier synchronization, respectively. As coherent demodulation is required, we must solve the outstanding phase ambiguity in recovered QPSK constellation points. We compute a phase offset against the symbols which comprise an attached sync marker present at the start of each CCSDS frame. The subsequent block corrects this phase offset allowing us to decode constellation points into bits. Under ideal conditions this phase and frequency offset could be done jointly in a single step. In practice, we find a coarse frequency

Parameter	Value
RF Frequency	2280 MHz
Sample Rate	2.88 Msamples/sec
Input Signal Level	-40 dBm
Sample Format	Signed Fixed-Point Complex

TABLE II: Selected metrics contained in IF context packets captured during a July 20, 2022 pass.



Options
Title: TES-13 Demod
Author: Adam Gannon
Output Language: Python
Generate Options: QT GUI

Import
Import: np

Import
Import: math

Variable
ID: sym_rate
Value: 600k

Symbol rate of the Lunar Radio

Variable
ID: samp_rate
Value: 2.88M

Rate that samples were captured at by AWS. This is pulled from IF Context.

Variable
ID: sps
Value: 4.8

Constellation Object
ID: const_qpsk
Constellation Type: QPSK

File Source
File: ...6/samples_merged.dat
Repeat: No
Add begin tag: ()
Offset: 0
Length: 0

Throttle
Sample Rate: 1M

Complex Conjugate

There appears to be a spectral inversion at either RX or TX.

Feed Forward AGC
Num Samples: 256
Reference: 1

Symbol Sync
Timing Error Detector: Gardner
Samples per Symbol: 4.8
Expected TED Gain: 2.7
Loop Bandwidth: 6.28319m
Damping Factor: 1
Maximum Deviation: 750m
Output Samples/Symbol: 1
Interpolating Resampler: MMSE, 8 tap FIR

Costas Loop
Loop Bandwidth: 62.8319m
Order: 4

First Costas achieves time/freq sync so that correlator can do its job.

Virtual Sink
Stream ID: ambig_symbols

Variable
ID: processing_rate
Value: 1M

This simply limits how fast the flow graph runs.

Virtual Source
Stream ID: ambig_symbols

These symbols have (hopefully) been sync'd to QPSK constellation. But we still have phase offset ambiguity to resolve.

Multiply Const
Constant: 2.3

Correlation Estimator
Symbols: sync_modulated
Samples per Symbol: 1
Tag marking delay: 0
Threshold: 900m
Threshold Method: Absolute

Correlator cares about amplitude!

Costas Loop
Loop Bandwidth: 62.8319m
Order: 4

Second Costas operates on the phase_est tag output by correlation and corrects that phase offset.

Constellation Decoder
Constellation Object: ...d30>

Unpack K Bits
K: 2

Virtual Sink
Stream ID: raw_bits

Variable
ID: preamble_chunks
Value: [0, 1, 2, 2, 3, 0, ...

This is the CCSDS sync word (0x1ACFFC1D) converted into 2-bit chunks.

Variable
ID: rxmod
Value: <gnuradi...18255a02b0>

Variable
ID: ac_string
Value: 00011010...10000011101

The CCSDS sync word (0x1ACFFC1D) as a string of bytes.

Variable
ID: sync_frame_len_bytes
Value: 1.279k

How many bytes are expected between sync words. See MATLAB for full calculation.

Modulate Vector
ID: sync_modulated
Modulator: <gnura...243975f0>
Data vector: preamble_chunks
Filter taps: 1

Virtual Source
Stream ID: raw_bits

Stream of un-aligned hard bits.

Correlate Access Code - Tag
Access Code: 00011...00011101
Threshold: 0
Tag Name: sync

This will output a "sync" tag on the first bit after the ASM.

Tagged Stream Align
Length tag names: sync

Outputs nothing until the first tagged sample. This lets the byte repacking work.

Repack Bits
Bits per input byte: 1
Bits per output byte: 8

Skip Head
Num Items: 1.275k

I want file to start with ASM, but correlate_access_code tags after ASM. Skip the rest of the first packet so we start writing at the first byte of the next ASM.

File Sink
File: /data/aligned_bytes.bin
Unbuffered: Off
Append file: Overwrite

Fig. 3: GNU Radio flow graph for demodulation of the TES-13 downlink signal.

offset correction is required before accurate phase estimate can be computed. Therefore we take a two-step approach. Finally the attached sync marker is again used for byte alignment and recovered bytes are written to a file.

C. Data Processing

Reed-Solomon decoding is applied to the byte-aligned data file. This currently takes place manually but is intended to be integrated into the cloud workflow in future flights. Telemetry is taken every second and multiplexed into the downlink along with data files. Using cross-account bucket access, team members from both Glenn and Ames Research Centers can access all stored data. An example of extracted telemetry is shown in Fig. 4. The increase in current is noticeable when the transmitter completes its power on process, about 60s from the transmit start command.

V. Discussion

A. S-band Contacts

After launch and initial checkout using the Iridium channel, an S-band downlink to the AWS Ground Station in Ohio was completed on 3/23/22. Performance of subsequent downlinks became intermittent in the following months with many receiving no signal from the spacecraft. The most recent successful S-band pass occurred on 7/20/2022 though the satellite continues to communicate over Iridium. We believe the root cause is transmitter power draw causing the bus voltage to drop significantly and resulting in the system successively rebooting during the contact. Future flights will use a larger capacity battery pack that should prevent this issue.

Signal strength for the July pass is plotted in Fig. 5 using the methods described in Section IV-A. This is compared with an orbit mechanics simulation run against a historical TLE from the day of the pass. We see good agreement in the times for signal acquisition though note there is an additional 3-10dB of loss in the measured data compared with our expectation. This is possibly due to additional loss not taken into account in the link budget due to cabling.

Though only exercised several times, the automated processing flow described in this work was successful in reducing labor required to operate a contact and extract useful data from a downlink. Completion of these contacts validated the flow described in this work and also greatly increased the achievable data rate for this mission. Over Iridium the spacecraft could send messages on the order of 300 bytes with successful transmissions occurring approximately every hour, based on signal conditions. The first S-band downlink transferred 35MB of data, a throughput several orders of magnitude greater.

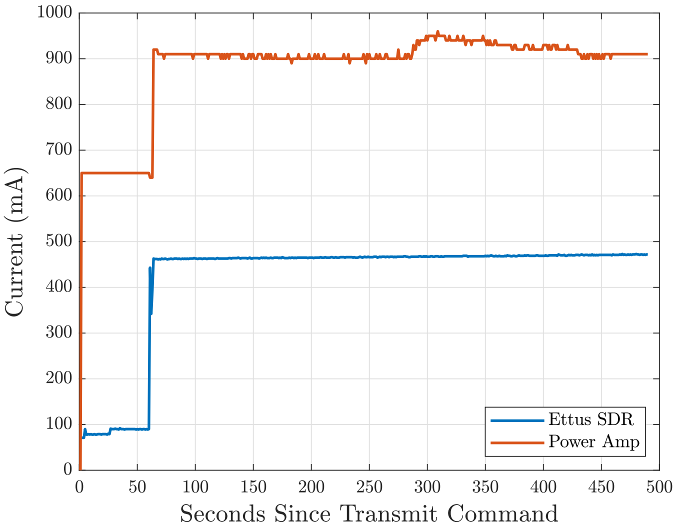


Fig. 4: Current of SDR and power amplifier immediately before and during the pass on July 20, 2022.

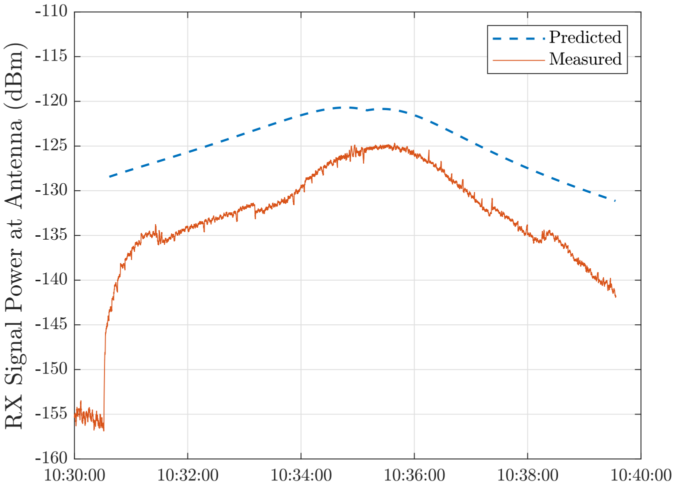


Fig. 5: Predicted (dashed) versus measured (solid) signal power at antenna for the July 20 pass.

B. Enabling Technologies

Though in this work scheduling is done manually by an operator, past demonstrations have shown this process can be automated using machine-to-machine communication by building applications around the AWS scheduling API [11]. A low-rate, high-availability control channel to Earth (such as the Iridium Short Burst Data service used by this spacecraft) can be used to relay messages about the user spacecraft's communications needs. Service on single-access apertures such as AWS Ground Station sites can then be provisioned based on the near-term needs of a vehicle or its instruments [12], [13]. Previous on-orbit demonstrations of this user-initiated service concept [14] have used government relays for the control channel and data service. However, NASA intends to augment and eventually replace government-operated systems with commercial ground stations and space relays [15]. Towards this future, our upcoming TES-11 mission will demonstrate a fully commercial version of automated user-initiated scheduling using Iridium as a control channel and AWS Ground Station as a data service.

As the capacity offered by Ground Station as a Service providers continues to grow, a wide variety of different candidate links will be available to missions. Automation as described above can help manage the complexity of evaluating factors such as capacity, latency, availability, and cost against mission needs. Intelligent agents capable of learning from past selections can refine the scheduling process to make increasingly optimal decisions. Occasionally a pass will fail to deliver the anticipated data volume due to mechanical issues, improper configuration, weather effects (especially at higher frequencies), or inadvertent in-band interference. In this scenario, intelligent agents could reschedule a new pass to route data around a provider site encountering malfunctions. The cloud-based, autonomous approach described in this work accommodates such a scenario without requiring specialized agreements or mission-provided hardware deployed at each ground station site.

Acknowledgments

We thank Dr. Kelso's CelesTrak [16] for providing historical TLE data used in analysis. We appreciate the support of the Mission Cloud Platform office at NASA Goddard Space Flight Center for provisioning and managing access to AWS cloud services. We also thank the AWS Ground Station team for their assistance answering technical questions during our satellite onboarding.

References

[1] L. Nguyen, "Ground stations as a service (gsaas) for near real-time direct broadcast earth science satellite data," in *NASA Earth Science Technology Forum*, Virtual, Jun 2021.

[2] K. Hughes, P. di Pasquale, A. Babuscia, and L. Fesq, "On-demand command and control of asteria with cloud-based ground station services," in *2021 IEEE Aerospace Conference*, Big Sky, MT, 2021.

[3] J. Wilbur *et al.*, "Developing a prototype ground station for the processing, exploitation, and dissemination of pleo sensor data," in *35th Annual Small Satellite Conference*, Logan, UT, August 2021.

[4] P. Baldwin, W. Evans, J. Huebner, M. Saraf, S. El-Nimri, E. Weir, and D. Rodriguez, "Nasa's implementation of cloud services for human space flight," in *17th International Conference on Space Operations (SpaceOps)*, Dubai, United Arab Emirates, March 2023.

[5] M. Seablom *et al.*, "Advancing technology for NASA science with small spacecraft," in *32nd Annual AIAA/USU Conference on Small Satellites*, Aug. 2018.

[6] "What are NASA's technology educational satellites?" https://www.nasa.gov/ames/techedsat, accessed: May 2023.

[7] M. Murbach *et al.*, "Techedsat-13: The first flight of a neuromorphic processor," in *CubeSat Developer's Workshop*, Apr. 2022.

[8] N. Rahman *et al.*, "Spiking neuromorphic software defined networking on the intel loihi running in space," in *Cognitive Communications for Aerospace Applications Workshop*, Virtual, June 2023, in press.

[9] "Vita Radio Transport (VRT)," Standard ANSI/VITA 49.0-2015.

[10] "libvrt," https://github.com/ember91/libvrt.

[11] A. Gannon, S. Paulus, C. Gemelas, and L. Vincent, "Spacecraft-initiated scheduling of commercial communications services," in *39th International Communications Satellite Systems Conference (ICSSC)*, Stresa, Italy, Oct. 2022.

[12] R. C. Reinhart, J. S. Schier, D. J. Israel *et al.*, "Enabling future science and human exploration with NASA's next generation near earth and deep space communications and navigation architecture," in *68th International Astronautical Congress*, Adelaide, Australia, Sep 2017.

[13] D. J. Israel, C. Roberts, R. M. Morgenstern, J. Gao, and W. S. Tai, "Space mobile network concepts for missions beyond low earth orbit," in *International Conference on Space Operations (SpaceOps)*, Marseille, France, May 2018.

[14] D. J. Mortensen, C. Roberts, and R. C. Reinhart, "Automated spacecraft communications service demonstration using NASA's SCaN testbed," in *International Conference on Space Operations (SpaceOps)*, Marseille, France, May 2018.

[15] G. Heckler, B. Younes, J. Mitchell *et al.*, "NASA's wideband multilingual terminal efforts as a key building block for a future interoperable communications architecture," in *Proc. 26th Ka and Broadband Communications Conference*, Arlington, VA, Sep 2021.

[16] "Celestrak," celestrak.org.