Evaluating a Cognitive Extension for the Licklider Transmission Protocol in a Spacecraft Emulation Testbed

Ricardo Lent
University of Houston
Houston, Texas, USA
rlent@uh.edu

Rachel Dudukovich
Cognitive Signal Processing Branch
NASA Glenn Research Center
Cleveland, OH, USA
rachel.m.dudukovich@nasa.gov

Joseph Downey
Cognitive Signal Processing Branch
NASA Glenn Research Center
Cleveland, OH, USA
joseph.a.downey@nasa.gov

Adam Gannon
Cognitive Signal Processing Branch
NASA Glenn Research Center
Cleveland, OH, USA
adam.gannon@nasa.gov

Ethan Schweinsberg
Flight Software Branch
NASA Glenn Research Center
Cleveland, OH, USA
ethan.e.schweinsberg@nasa.gov

Evan Danish
Software Engineer
ZIN Technologies
Cleveland, OH, USA
danishe@zin-tech.com

***Abstract*—In space communications, particularly when involving regions beyond cislunar space, the development of advanced networking solutions is essential to address the challenges posed by limited connectivity, substantial propagation delays, and radio signal variations. This study explores a data-driven approach to the Licklider Transmission Protocol (LTP), specifically focusing on dynamically adjusting the maximum payload size of segments. Prior research has emphasized the potential benefits of dynamically adjusting this parameter, introducing the concept of Cognitive LTP. This paper presents a software implementation of Cognitive LTP (CLTP) within an open-source Delay Tolerant Networking (DTN) framework, specifically the High-rate Delay Tolerant Networking (HDTN), and experimentally evaluates its performance under realistic space conditions. Leveraging the Cognitive Ground Testbed (CGT), developed by NASA GRC for spacecraft communication emulation, this study effectively bridges the gap between theoretical advancements and practical applications. By thoroughly analyzing CLTP's functionality within the CGT, this research offers insights into the practical implications of adaptive networking strategies, emphasizing the importance of conducting tests in relevant environments for the maturation of space communication technologies.**

***Index Terms*—Delay Tolerant Networking, Reinforcement Learning, Licklider Transmission Protocol, Spiking Neural Networks**

I. Introduction

Space networks are inherently sparse systems operating in extremely harsh environments, posing significant technical challenges. These challenges lead to intermittent network connectivity and substantial propagation delays, affecting individual channels to different extents. To address these issues, the DTN architecture has been developed, departing from the traditional assumptions of classic networking protocols. A notable protocol within DTN is the Licklider Transmission Protocol (LTP) outlined in RFC 5326. LTP serves as a convergence layer protocol supporting both reliable and best-effort bundle deliveries, employing a method of transmitting data bundles as a sequence of segments.

A crucial parameter within LTP is the maximum payload size of segments, which is typically determined manually by the operator. Previous studies have highlighted the significant influence of this parameter on block delivery performance, as well as the potential advantages of dynamically adapting its value to different operational scenarios. Both analytical and simulation outcomes have indicated the advantages of this approach, often referred to as Cognitive LTP [1]. However, an open question, which the present study endeavors to address, concerns the practical performance of this approach when evaluated within the context of standard space protocols and under realistic operational conditions.

The Cognitive Communications Project at NASA Glenn Research Center (GRC) has been focused on increasing the technology readiness level of advanced space communications systems utilizing state-of-the-art algorithms for automation and cognition[2]. A significant gap between the development of many delay tolerant and interplanetary networking optimizations using artificial intelligence, machine learning, and related techniques [3], [4], [5] is the lack of testing in a relevant environment, which is one of the key criteria for technology maturation. To address this need, a realistic spacecraft emulation testbed known as the Cognitive Ground Testbed (CGT) has been developed by NASA GRC [6].

This paper addresses the development of a cognitive extension for LTP (CLTP) that is then tested in both a laboratory computer network and the high-fidelity emulation environment of the CGT consisting of software defined radios, flight computers, channel emulators, modems, orbital analysis, mission operations, and network automation servers. Consequently, this paper's contributions include the development of a software implementation of CLTP integrated into an open-source DTN protocol distribution, namely HDTN. Our study rigorously examines the performance of CLTP through real-world testing under approximated space conditions, enhancing the credibility and practicality of our findings. We believe that the approach taken in both method implementation and test environments represents a novel and substantial contribution towards the advancement of cognitive networking technologies.

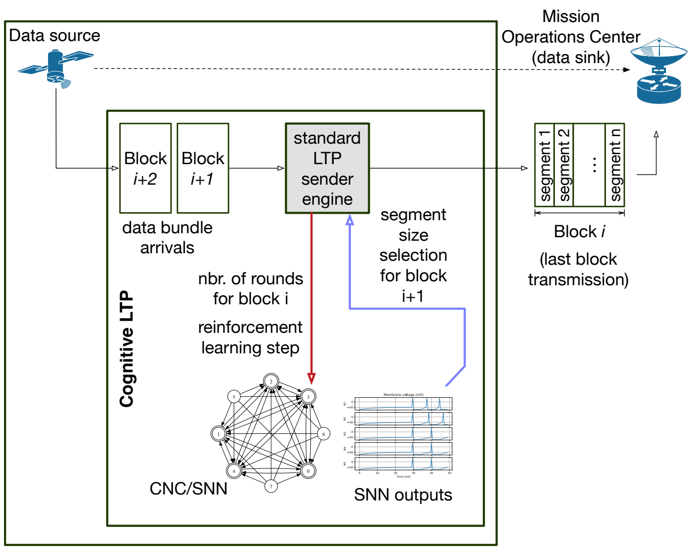
II. Related Works

Several studies have approached topics closely related to this research. Bezirgiannidis and Tsaoussidis [7] conducted experimental investigations on the impact of packet size. Additionally, Lu et al. delved into an analysis of packet sizes for DTN through non-convex optimization [8]. Notably, the latter work relies on knowledge of channel conditions, introducing practical limitations on its applicability. The academic community has also invested substantial effort in evaluating the performance of DTN and space network protocols, with various studies focusing on different facets of LTP. These aspects include assessing LTP's flow control [9], examining bundle aggregation within blocks [10], [11], and investigating the effects of the convergence layer [12], [13]. Recent research endeavors have sought to enhance LTP performance, for instance, by implementing a Reed-Solomon code to mitigate segment loss probabilities [14]. On the other hand, HDTN has been developed by NASA GRC to optimize many aspects of interplanetary networking [15]. The project was originally conceived to support high-rate optical communications onboard the International Space Station [16], however it also serves as a networking framework for a variety of cognitive applications [17], [18], [19], routing methods [20], and forward error correction for CubeSat applications [21]. HDTN uses a modular architecture and high-speed message bus to allow enhancement of basic functions with external third-party modules, simplifying software development and integration.

III. Cognitive Approach to LTP Segment Optimization

A. Problem Formulation

The fundamental concept entails replacing the conventional static data segment size selection methodology of the standard LTP with a dynamic selection approach driven by an intelligent agent or controller. This dynamic selection method presents a sequential resource allocation challenge among several competing alternatives. In this context, these alternatives correspond to distinct segment sizes available for partitioning a data block for transmission. The objective revolves around optimizing the attainable rewards, where a reward can be interpreted as minimizing the delivery delay for bundles. This optimization is achieved by effectively striking a balance between the header overhead of individual segments and the necessity for segment retransmissions in the event of one or more segments being lost during a transmission round. The role of the cognitive extension for LTP is given in Fig. 1.



Data source
Mission Operations Center (data sink)
Block i+2
Block i+1
standard LTP sender engine
segment 1
segment 2
...
segment n
Block i
(last block transmission)
data bundle arrivals
segment size selection for block i+1
nbr. of rounds for block i
reinforcement learning step
Cognitive LTP
CNC/SNN
SNN outputs

Fig. 1. Overview of the proposed method that dynamically selects the segment length for each new LTP block transmission. The number of rounds required by previous sessions serves to adjust the synapse strengths of the SNN, helping the CNC to improve the next segmentation.

The operational dynamics of the LTP communication system, employing a specific segment length, can be characterized through the utilization of two stochastic sequences, denoted as $x(0), x(1), \ldots$ and $R(x(0)), R(x(1))), \ldots$, where $x(n)$ denotes the state of the communication after n block transmissions and $R(x_n)$ the reward obtained for the n-th transmission. The reward can be interpreted as the inverse of the block delivery time. The state transition after the n-th block transmission is given by: $x(n) = f_{n-1}(x(0), x(1), \ldots, x(n-1), W(n-1))$, where $f_{n-1}(.)$ is the transition function and $W(n)$ is a sequence of independent random variables making the process not necessarily Markov.

Given k possible segment sizes (actions) to be used, the problem constitutes a type of multi-armed bandit process where the controller needs to choose every time exactly one action with all other actions disabled. Therefore, the system behavior with each possible action can be described by the sequences $\{(x_i(n_i(t)), R(x_i(n_i(t))))\}$, where $n_i(t)$ denotes the number of times the i-th segment size option was selected until time t. If $U(t) = (U_1(t), \ldots U_k(t))$ represents the action taken by the controller at time t. $U(t)$ takes values in $\{e_1, \ldots, e_k\}$ where e_j is a unit k-vector of zeros except for a 1 at the j-th position.

The system's evolution unfolds according to the following rules: $x_i(n_i(t+1)) = f_{n_i(t)}(x_i(0), \ldots, x_i(n_i(t)), W(n_i(t)))$ if $U_i(t) = 0$. Otherwise, $x_i(n_i(t+1)) = x_i(n_i(t))$. Also, $n_i(t+1) = n_i(t)$ for $U_i(t) = 0$ and $n_i(t+1) = n_i(t) + 1$ otherwise. A given segment selection only yields a reward when it is chosen:

$$R_i(t) = R_i(x_i(n_i(t)), U_i(t)) = \begin{cases} R_i(x_i(n_i(t))) & ; U_i(t) = 1 \\ 0 & ; U_i(t) = 0 \end{cases}$$

The scheduling policy $\gamma = (\gamma_1, \gamma_2, \dots)$ defines the control action $U(t)$ for each time t:

$U(t) = \gamma_t(Z_1(t), \dots, Z_k(t), U(0), \dots, U(t-1))$, where $Z_i(t) = [X_i(0), \dots, X_i(n_i(t))]$. The optimization problem is therefore to find the scheduling policy that maximizes:

$$J^\gamma = \mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t \sum_{i=1}^{k} R_i(x_i(n_i(t)), U_i(t)) | Z(0)\right] \quad (1)$$

B. Cognitive Network Controller

Expression 1 is approximated through a sequential decision-making processes computed with a Spiking Neural Network (SNN) architecture. This architecture comprises of k excitatory neurons, each corresponding to a possible segment size, and their activity serves as an indication of the optimal choice. Additionally, the network includes $k-2$ inhibitory neurons responsible for regulating the membrane potential across the network. The spiking neurons within this framework are characterized by the Leaky Integrate and Fire (LIF) model [22], which models the membrane potential using a resistor–capacitor circuit. A spiking neuron emits a brief pulse or spike upon reaching a specific threshold in its membrane potential. Network connections facilitate the transmission of spikes to other neurons, with amplification given by the weight values associated with specific connections. In this architecture that was initially developed for routing [23], the outputs of all neurons are connected to every excitatory neuron.

Upon the arrival of a spike, the membrane potential of the receiving neuron either increases or decreases depending on whether the source neuron is of the excitatory or inhibitory type. All neurons receive a constant stimulus of sufficient magnitude to evoke an initial spike simultaneously. Given the uniform stimulus across all neurons, the primary differentiating factor influencing the emission of new spikes becomes the weight values of the connections. Consequently, these weight values undergo modification via a reinforcement learning step, which either encourages or discourages the re-selection of the last action. The optimal action is encoded in the timing of the second spike emission.

As the reward signal becomes available only after the LTP sender decides either to drop a data block (resulting in a very low reward signal) or upon receiving confirmation of successful delivery from the receiver, these events determine when the reinforcement learning step is executed. To this end, the controller maintains an average cost G_i associated with each segment size choice i through exponential averaging of the number of transmission rounds C_i required for successful block delivery. In cases where the block was dropped, a penalty term is applied: $G_i \leftarrow \alpha C_i + (1-\alpha) G_i$, where $0 \le \alpha \le 1$ is a hyperparameter. The synapse weights are updated as follows with the third update element added for faster convergence:

$$\begin{array}{lll} w(c_j, c_i) & \leftarrow w(c_j, c_i) + \eta\delta & ; j = 0, \dots, k-1; i \neq j \\ w(c_l, c_i) & \leftarrow w(c_l, c_i) - \eta\delta & ; l = k, \dots, 2k-3; \\ w(c_i, c_j) & \leftarrow w(c_l, c_i) - \eta\delta & ; j = 0, \dots, k-1; i \neq j \end{array} \quad (2)$$

where $\eta > 0$ is the learning rate and $\delta = C - min\{G_i\}$.

IV. Implementation

Because the cognitive extension for LTP only affects the sender engine, it can be easily integrated into existing implementations of the protocol. In the LTP protocol, incoming bundles are aggregated into blocks, and each block is divided into segments for transmission, each with a maximum size denoted as L. These segments are dispatched sequentially when there is a suitable contact opportunity. It is assumed that data blocks are sent using the reliable block delivery mode, i.e., with the retransmission of lost segments, which occurs in retransmission rounds. To monitor the progress of this process, a counter, denoted as *nbr_tx*, is used to keep track of the number of transmission rounds required for the reliable delivery of the block. This value is approximated by incrementing the counter each time a checkpoint (CP) number is received within a report segment (RS). Since LTP allows sending multiple RS for the same CP in some cases, the counter may not be an accurate measurement. However, it still serves to assess the approximate retransmission effort level needed for a block under the current channel conditions.

The typical procedure concludes when either the block is successfully delivered, leading to the transmission of a final report acknowledgment (RA) to the receiver, or when the block is dropped after a session failure. However, with the introduction of the cognitive extension, an additional step occurs at this point: a reinforcement learning process takes place, resulting in the updating of the SNN weights, as elaborated upon in Section III. The update process employs the inverse of *nbr_tx-ref* as the reward signal, where *nbr_tx* is intentionally set to a high value (MAX) as a penalty for block drops. Here, the value denoted as *ref* represents the minimum average number of transmission rounds among the available options. The value of MAX should be equal to or greater than the maximum anticipated number of transmission rounds (usually a protocol configuration parameter). A larger value intensifies the penalty for block drops. Subsequent to the SNN weight update, the SNN is executed to determine the new action, specifically the segment size value to be employed for the next block transmission. These sequential steps are summarized in the flow chart depicted in Fig. 2.

A. Modifications Made to HDTN

HDTN is an implementation of the standard DTN protocols developed at NASA's GRC that is available as open source [15]. The implementation of the cognitive extension logic for HDTN primarily impacts two key classes. Firstly, the *LtpSessionSenderCommonData* class has been extended to incorporate three additional members: a boolean flag that

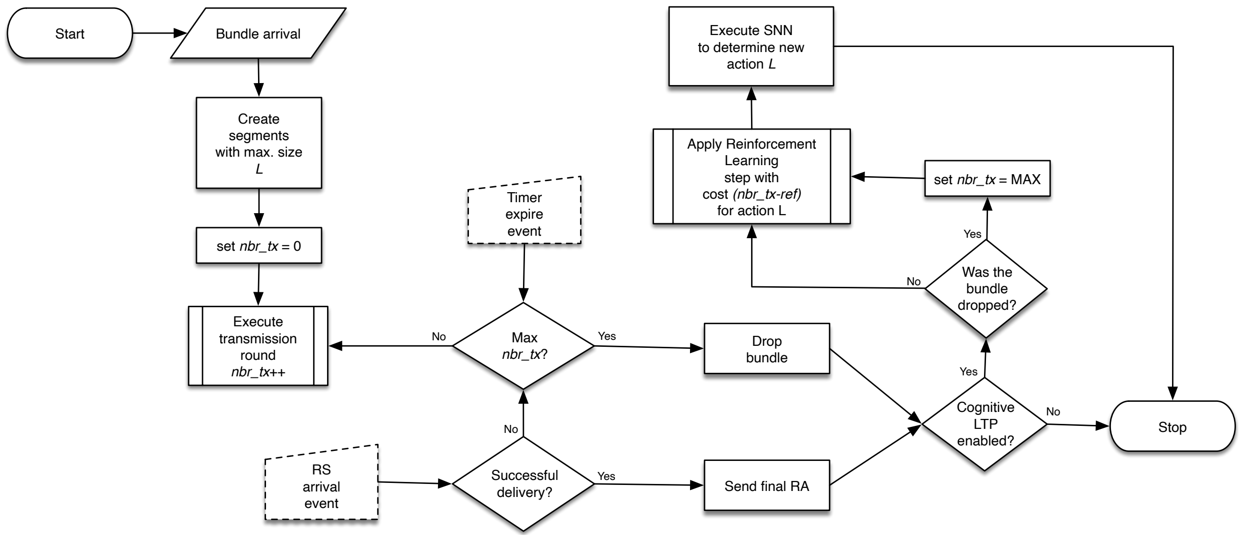


Fig. 2. Simplified flow chart of LTP including the cognitive extension that dynamically optimizes segment lengths. The boxes with dashed borders indicate the two main events that may occur after a transmission round.

indicates whether the cognitive extension should be applied, the count of available actions representing the number of segment length options, and a pointer pointing to the data structure responsible for implementing the Cognitive Network Controller (CNC). Enabling the extension can be achieved through standard HDTN configuration, specifically by zeroing the *ltpDataSegmentMtu* parameter within the LTP JSON data structure.

Secondly, the *LtpSessionSender* class has been expanded to include a counter tracking the number of transmission rounds, which is incremented by the *ResendDataFromReport* member function. Additionally, a set of hyperparameters essential for the training phase of the Spiking Neural Network (SNN) has been included. This training phase can be initiated from either the *LtpCheckpointTimerExpiredCallback* member function or the *ReportSegmentReceivedCallback* member function, contingent upon the successful delivery of the block.

Following the execution of the SNN and the consequent SNN training, the value of *ltpDataSegmentMtu* is updated. Additionally, for exploration purposes, a small fraction of decisions is replaced with random selections, albeit exclusively after the successful delivery of a block.

B. Auxiliary Modifications

A few additional modification were introduced to facilitate the performance measurement of the cognitive extension. The *BpGenAsyncRunner* class was modified to store in a logfile the number of transmission rounds after a block delivery and the *BpSinkAsyncRunner* was changed to send the bundle creation time to the source using a ZeroMQ socket via the control network. The information is received by a background process that calculates the round-trip time for the bundle transmission, which approximates the one-way delivery time as the delay of the return path via the control network is negligible.

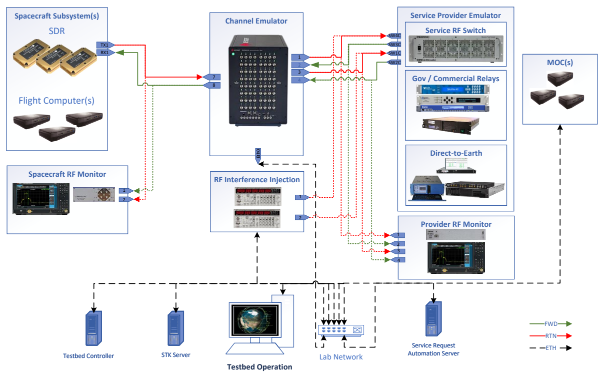


Fig. 3. Cognitive Ground Testbed components. Red and green lines represent Forward (FWD) and Return (RTN) link signals. Red and green dotted lines represent coupled ports for monitoring or interference injection. Further details can be found in [6].

Moreover, to validate the implementation on a standard computer network, specifically in the absence of radio equipment, the *HandleUdpReceive* member function within the *LtpUdpEngineManager* class was modified to introduce random packet drops with a probability given by the formula $1-(1-ber)^{8L}$, where L denotes the segment payload size.

V. Results

Tests were conducted in two settings. The initial code development took place in a network laboratory, where emulated packet drops were used to simplify the initial experimentation. Further validation was carried out on a Cognitive Ground Testbed using software-defined radios and more realistic channel conditions.

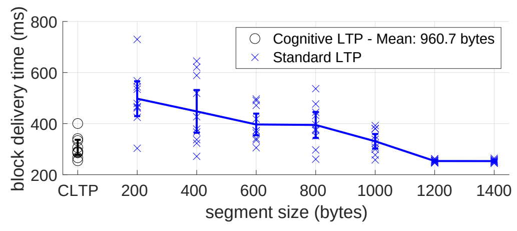
A. Network Laboratory Results

The initial validation was conducted using a wired 100 Mbps wired channel connecting both the sender and receiver, primarily for experimental convenience. To emulate a propagation delay of 100 ms, a *NetEm* queue discipline was applied to the network interfaces at both ends of the link. Packet losses were intentionally introduced to match a predefined Bit Error Rate (BER), as detailed in Section IV (B). This BER value served as one of the key experimental parameters.

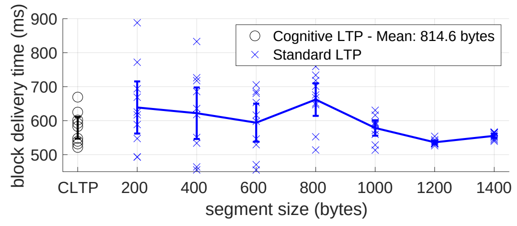
Each experiment involved the transmission of 100 kB bundles over a duration of 5 minutes, at a rate of 1 bundle per second. These experiments were conducted with two different configurations: one with the cognitive extension enabled for dynamic segment size selection and the other using a fixed segment size throughout the entire experiment's duration.

The experiments concluded either when all bundles were successfully received or when there was a lack of reception activity for a specified period. HDTN's LTP convergence-layer adapter (LTP over UDP) was configured to allow a maximum of 10 retransmission rounds, as determined by the parameter *ltpMaxRetriesPerSerialNumber*. For each set of parameters, a minimum of 5 samples were collected, with each sample comprising data on the average bundle delivery time, bundle loss ratio, average number of transmission rounds, and average segment payload length.

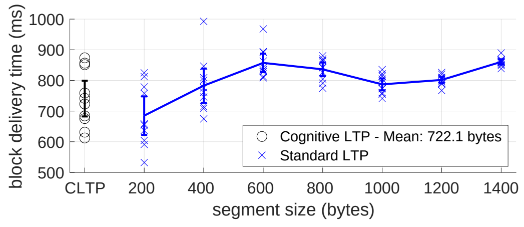
Fig. 4 illustrates the average response time of the test bundle flow in relation to the selected segment payload size across different channel BER values, specifically 10^{-6}, 10^{-5}, 2×10^{-5}, and 10^{-4}. The block delivery times achieved by CLTP are reported on the left side of each chart using circular markers. The cross markers indicate the response times obtained by the standard LTP after fixing the segment payload length to the value shown on the horizontal axis. The response time of the test flow without emulated packet losses closely resembled that of the 10^{-6} case, leading to the omission of that particular data point from the chart. At BER levels of 10^{-6} or lower, it becomes evident that the response time tends to decrease with larger segment sizes. This trend is expected, as larger segments help mitigate the overall header overhead of the bundle flow, and their loss rates are minimal due to the low BER. In this scenario, CLTP discovered an average segment length of 960.7 bytes. However, with a BER of 10^{-5}, the desirability of larger segments persists, albeit the higher loss rate associated with larger segments renders them somewhat less attractive. For this experiment, CLTP determined an average segment size of 814.6 bytes. This trend continues as the BER increases to 2×10^{-5}, where the average segment length identified by CLTP decreases to 722.1 bytes. It is worth noting that in these two tests, we observed higher variability in the results. Finally, with a substantial BER of 10^{-4}, the optimal strategy shifts toward employing much smaller segment sizes, which are less susceptible to loss, thereby conserving retransmission time. For this scenario, CLTP identified an average segment size of 460.2 bytes. Furthermore, the experiment with a BER of 10^{-4} was the only one that resulted in significant block drops as depicted in Fig. 5. In this case, the block loss ratio exhibited a rapid increase for larger segments, reaching approximately 0.12.



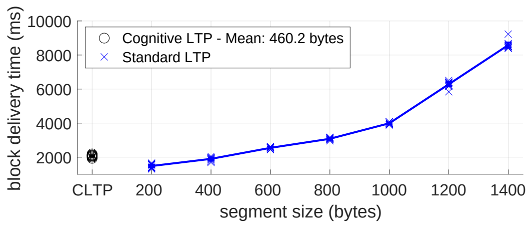
(a) BER 10^{-6}



(b) BER 10^{-5}



(c) BER 2×10^{-5}



(d) BER 10^{-4}

Fig. 4. Block delivery times showing the 95% confidence interval with emulated packet losses according to the indicated target BER values. The markers show the sample values and represent the average block delivery time of individual experiments.

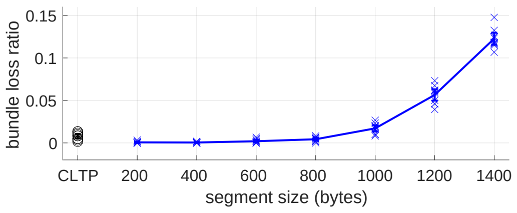


Fig. 5. Block drop ratios for a BER value of 10^{-4}. Smaller BER values did no produce significant block drops.

B. *Cognitive Ground Testbed Results*

Realistic tests were conducted using the CGT (Fig. 3) which aims to emulate end-to-end communications between a spacecraft and its mission operations center (MOC). Hardware used in CGT includes: software-defined radios (SDRs) suitable for smallsat missions (CesiumAstro SDR-1001), single-board computers for spacecraft and MOCs, modems representative of various service providers, and a radio channel emulator (Keysight Propsim F64). AGI's System Tool Kit (STK) is used to model orbital dynamics and estimate received signal parameters. During an active contact, the channel emulator connects RF signals between the spacecraft and the active provider's modem. Link parameters computed in STK (signal-to-noise ratio (SNR), Doppler shift) are added to emulate signal conditions as they would appear on-orbit.

The tested scenario considered a single user spacecraft in low-Earth orbit sending data via geostationary relay satellites. Ephemerides of the International Space Station and the Inmarsat Global Xpress constellation were used for orbital mechanics computations. A Ka-band (26.5-40 GHz) terminal with RF parameters similar to [24] was assumed in link budget calculations. The 2nd Generation Digital Video Broadcasting – Satellite (DVB-S2) standard [25] was used as the physical-layer protocol and a QFlex-400 DVB-S2 modem was used to represent service provider hardware. A corresponding DVB-S2 transceiver was implemented on the SDR which was limited to a MTU of 1,005 bytes.

Since the proposed adaptive technique operates on established contacts, the measurements were obtained without regards of the time required to establish these contacts. No artificial packet loss emulation was introduced during the tests, as the channel naturally produced losses due to the selected waveform and SNR conditions at the receiver.

The test flow comprised 10 kB bundles transmitted from the emulated spacecraft node to the mission operation control (MOC) node at a rate of 2 bundles per second for a duration of 300 seconds. Subsequently, network performance metrics were collected and stored for later analysis. The traffic generation was performed using the *bpgen/bpsink* programs, consistent with the network laboratory tests. In the interference-free scenario, the observed block delivery times for the bundles are presented in Fig. 6 (a). In this context, the values for smaller packets were higher compared to the network laboratory tests due to the increased header overhead associated with smaller segments. Conversely, larger segments proved to be more desirable in this scenario, aligning with the dynamically chosen values by CLTP, which averaged 915 bytes as shown in Fig. 6 (b).

To investigate the protocol's performance under adverse communication conditions, interference was introduced by overlaying a second QPSK signal at 1.65 GHz. Various power levels were tested, revealing that the results tended to exhibit more variability compared to tests with emulated packet losses. Notably, synchronization issues emerged between the transmitter and receiver, worsening with higher power levels. This phenomenon rendered the channel unavailable for certain periods, thereby adding additional delay to block transmissions and causing intermittent disruptions within contacts. When employing lower power levels (typically below -14 dBm), no significant packet drops were observed, and consequently, larger segment lengths remained optimal. As an example demonstrating the tradeoff, Fig. 7 illustrates the results with an interfering power level of -14.29 dBm. These results reveal that CLTP identified an average segment length of 460 bytes, achieving a lower average block delivery time than the regular LTP.

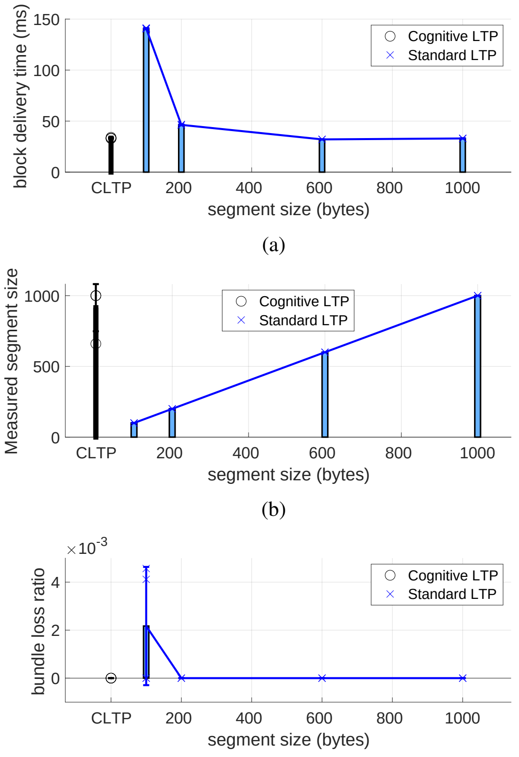
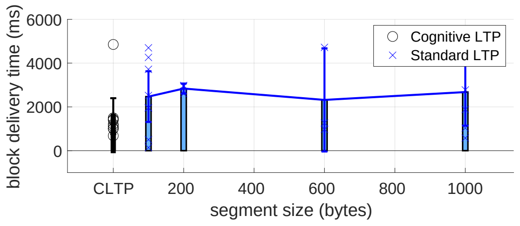


Fig. 6. Performance measurements of the bundle flow over a SDR channel without added interference: (a) average bundle delivery time; (b) mean average segment size; (c) bundle loss ratio.

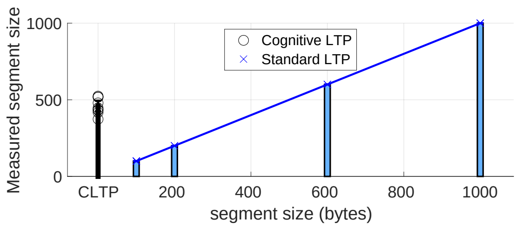
VI. Conclusion

In conclusion, this study has explored the potential of extending the Licklider Transmission Protocol with cognitive capabilities and subsequently validated this approach. By dynamically adjusting the maximum payload size of segments, CLTP has demonstrated its ability to outperform the traditional Licklider Transmission Protocol under specific loss conditions, offering valuable insights into data transmission adaptability within the challenging space environment.

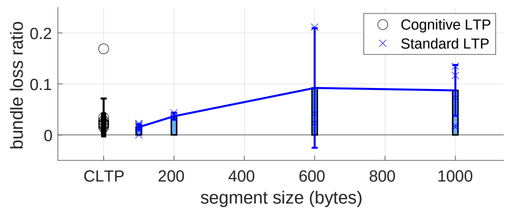
The software implementation of CLTP within the HDTN architecture, contributes with a substantial step forward in the maturation of cognitive networking technologies. Rigorous testing within the CGT provides a realistic and relevant envi-



(a)



(b)



(c)

Fig. 7. Results for a bundle flow over a SDR channel with added interference of -14.29 dBm a QPSK signal at 1.65 GHz: (a) average bundle delivery time; (b) mean average segment size; (c) bundle loss ratio.

ronment, bridging the gap between theoretical advancements and practical applications.

The findings underline the significance of adaptability and highlight the potential of Cognitive LTP in improving the efficiency and reliability of data exchange in space. As humanity's exploration of outer space continues, the insights from this study are poised to contribute to the advancement of space communication technology, enhancing its resilience and adaptability to the dynamic challenges of deep space exploration.

Acknowledgment

This work was supported by grant #80NSSC22K0259 and a faculty fellowship from the National Aeronautics and Space Administration (NASA).

References

[1] R. Lent, "A cognitive networking technique for LTP segmentation," in *2020 International Wireless Communications and Mobile Computing (IWCMC)*, 2020, pp. 263–268.

[2] D. Chelmins, J. Briones, J. Downey, G. Clark, and A. Gannon, "Cognitive communications for NASA space systems," in *37th International Communications Satellite Systems Conference (ICSSC)*, 2019, pp. 1–16.

[3] L. Yao, X. Bai, and Z. Wang, "An overview of intelligent algorithm-based routing in delay tolerant networks," in *2023 8th International Conference on Computer and Communication Systems (ICCCS)*, 2023, pp. 345–350.

[4] X. Sun, C. Li, L. Yan, and S. Cao, "Reinforcement learning for cognitive delay/disruption tolerant network node management in an leo-based satellite constellation," 2022.

[5] P. G. Buzzi, D. Selva, and M. S. Net, *Exploring Reinforcement Learning for Autonomous Delay Tolerant Network Management*.

[6] J. Downey, A. Gannon, A. Smith, M. Koch, and R. Dudukovich, "Emulated spacecraft communication testbed for evaluating cognitive networking technology," in *2023 IEEE Cognitive Communications for Aerospace Applications Workshop (CCAAW)*, 2023, pp. 1–6.

[7] N. Bezirgiannidis and V. Tsaoussidis, "Packet size and DTN transport service: Evaluation on a DTN testbed," in *Int. Congress on Ultra Modern Telecommunications and Control Systems*, Oct 2010, pp. 1198–1205.

[8] H. Lu, F. Jiang, J. Wu, and C. W. Chen, "Performance improvement in DTNs by packet size optimization," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 51, no. 4, pp. 2987–3000, Oct 2015.

[9] R. Wang, A. Reshamwala, Q. Zhang, Z. Zhang, Q. Guo, and M. Yang, "The effect of "window size" on throughput performance of DTN in lossy cislunar communications," in *2012 IEEE International Conference on Communications (ICC)*, June 2012, pp. 68–72.

[10] N. Bezirgiannidis, S. Burleigh, and V. Tsaoussidis, "Delivery time estimation for space bundles," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 49, no. 3, pp. 1897–1910, July 2013.

[11] Z. Wei, R. Wang, Q. Zhang, and J. Hou, "Aggregation of DTN bundles for channel asymmetric space communications," in *2012 IEEE International Conference on Communications (ICC)*, June 2012, pp. 5205–5209.

[12] R. Wang, B. Modi, Q. Zhang, J. Hou, Q. Guo, and M. Yang, "Use of a hybrid of DTN convergence layer adapters (CLAs) in interplanetary Internet," in *ICC*. IEEE, 2012, pp. 3296–3300.

[13] R. Wang, X. Wu, T. Wang, X. Liu, and L. Zhou, "TCP convergence layer-based operation of DTN for long-delay cislunar communications," *IEEE Systems Journal*, vol. 4, no. 3, pp. 385–395, Sept 2010.

[14] L. Shi, J. Jiao, A. Sabbagh, R. Wang, Q. Yu, J. Hu, H. Wang, S. C. Burleigh, K. Zhao, L. Shi, J. Jiao, A. Sabbagh, R. Wang, Q. Yu, J. Hu, H. Wang, S. C. Burleigh, and K. Zhao, "Integration of Reed-Solomon codes to licklider transmission protocol (LTP) for space DTN," *IEEE Aerospace and Electronic Systems Magazine*, vol. 32, no. 4, pp. 48–55, April 2017.

[15] NASA Glenn Research Center, "High speed delay tolerant network," https://github.com/nasa/HDTN. Last retrieved: 09-12-2023.

[16] D. Raible, R. Dudukovich, B. Tomko, N. Kortas, B. LaFuente, D. Iannicca, T. Basciano, W. Pohlchuck, J. Deaton, A. Hylton, and J. Nowakowski, " Developing High Performance Space Networking Capabilities for the International Space Station and Beyond," NASA, Technical Memorandum 20220011407, 2022.

[17] Y. Kirkpatrick, R. Dudukovich, P. Choksi, and D. Ta, "Cooperative clustering techniques for space network scalability," in *2023 IEEE Cognitive Communications for Aerospace Applications Workshop (CCAAW)*, 2023, pp. 1–8.

[18] A. N. Mody, B. Crompton, D. Tran, D. Giger, D. Simpson, D. Gormley, A. Smith, M. Kappes, D. Redelings, and T. Melodia, "Claire: Enabling heterogeneous communication network optimization for robust and resilient operations," in *2023 IEEE Cognitive Communications for Aerospace Applications Workshop (CCAAW)*, 2023, pp. 1–6.

[19] A. N. Mody, J. Islam, B. Crompton, D. Tran, D. Simpson, D. J. Gormley, A. Smith, M. M. Kokar, J. J. Moskal, and T. Melodia, "Inspire – an approach to mission quality management using network slicing for space applications," in *2023 IEEE Cognitive Communications for Aerospace Applications Workshop (CCAAW)*, 2023, pp. 1–6.

[20] R. Lent, "Implementing a cognitive routing method for high-rate delay tolerant networking," in *2023 IEEE Cognitive Communications for Aerospace Applications Workshop (CCAAW)*, 2023, pp. 1–6.

[21] N. P. Douglass, J. Langel, W. J. Moore, L. Ng, R. M. Dudukovich, and S. Mal-Sarkar, "Application of fountain code to high-rate delay tolerant networks," *IEEE Access*, vol. 11, pp. 100 845–100 855, 2023.

[22] W. Gerstner, W. M. Kistler, R. Naud, and L. Paninski, *Neuronal Dynamics: From Single Neurons to Networks and Models of Cognition*. Cambridge University Press, 2014.

[23] R. Lent, "A cognitive network controller based on spiking neurons," in *IEEE International Conference on Communications (ICC)*, October 2018.

[24] M. Piasecki *et al.*, "Development and demonstration of a wideband RF user terminal for roaming between ka-band relay satellite networks," in *Proc. 38th International Communications Satellite Systems Conference (ICSSC)*, Arlington, VA, Sep 2021.

[25] "Digital video broadcasting (DVB) second generation," European Telecommunications Standards Institute, Standard ETSI EN 302 307 V1.2.1, Aug. 2009.