Impact Analysis of Parameters in the Licklider Transmission Protocol

Gahyun Kim, Chihyun Song, Seongjin Choi, and Sungrae Cho
School of Computer Science and Engineering, Chung-Ang University, Seoul 06974, South Korea
Email: {ghkim, chsong, sjchoi}@uclab.re.kr, srcho@cau.ac.kr

***Abstract*—The Licklider Transmission Protocol (LTP) is a core convergence-layer protocol for deep-space Delay/Disruption Tolerant Networks (DTNs), yet its performance is highly dependent on parameter settings such as segment size, session count, and transmission rate. In this study, we analyze the quantitative impact of these parameters on goodput, delivery latency, and protocol overhead using a simulation framework that models LTP' s selective retransmission and timeout behaviors. Results show that each parameter has a distinct performance threshold, with optimal settings around 6–8 sessions, 4096-byte MTU, and transmission rates above 512 kbps. The empirical findings align with theoretical models, confirming key trade-offs between efficiency and reliability. Furthermore, we discuss the potential of integrating intelligent optimization methods—such as multi-agent deep reinforcement learning and explainable DRL—to support adaptive and interpretable LTP control in future deep-space communication systems.**

***Index Terms*—Licklider Transmission Protocol, LTP, Delay/Disruption Tolerant Networks, DTN, Reinforcement Learning, Deep Space**

I. Introduction

In next-generation space exploration missions, communication is required not only between the Earth, the Moon, and Mars but also across greater interplanetary distances. Space communication networks face multiple challenges such as long signal propagation delays, high data loss rates, and asymmetric channel characteristics, which make stable data transmission difficult to achieve. Consequently, there is an increasing demand for the development of data transmission protocols capable of ensuring reliability in deep-space environments [1].

To overcome these challenges, the Delay/Disruption Tolerant Networking (DTN) architecture has been adopted as a core component of modern space communication systems [2]. DTN utilizes a store-and-forward mechanism that allows data to be transmitted reliably even under intermittent connectivity. It divides an end-to-end communication path into multiple DTN hops, where each hop stores and forwards data locally. This architecture is realized through the implementation of the Bundle Layer and the Bundle Protocol (BP), which handle end-to-end data management and routing.

For deep-space operations, the Licklider Transmission Protocol (LTP) is employed within the Convergence Layer beneath the Bundle Protocol [3]. LTP segments each transmission file into multiple blocks, which are further divided into smaller segments before transmission. By performing block-based data delivery without connection establishment, LTP overcomes the limitations of conventional connection-oriented transport protocols such as TCP. In addition, each block terminates with a checkpoint, which triggers a report segment from the receiver and enables selective retransmission, thereby enhancing data reliability [4].

Due to this architectural design, the performance of LTP is highly influenced by several configuration parameters, including the Maximum Transmission Unit (MTU), number of sessions, block size, segment size, and timer settings. Nevertheless, the current specification provides no explicit guideline for selecting appropriate parameter values. Therefore, this paper conducts a quantitative analysis of how key LTP parameters affect communication performance. Through simulation-based experiments, the behavior of different parameter settings is examined, and the necessity of dynamic parameter control mechanisms for future deep-space networks is discussed.

II. Analytical Model of LTP Performance

This section presents the analytical foundations for evaluating how key parameters of the Licklider Transmission Protocol (LTP) influence performance metrics such as goodput, delivery delay, and retransmission overhead. These equations provide a theoretical basis that complements the simulation study in the next section. We focus on parameters that are subject to control or optimization in our simulation: segment size (MTU), number of concurrent sessions, transmission rate, and ARQ timers. Each expression highlights how protocol behavior changes under deep-space conditions, where link quality and latency dominate performance.

LTP is designed for delay- and disruption-tolerant networks (DTNs), and operates below the Bundle Protocol using red/green segments. Red segments require acknowledgment via checkpoint (CP) and report segments (RS), while green segments are sent without feedback. Given this hybrid ARQ model, performance is closely tied to the reliability of segment delivery, retransmission scheduling, and timer configuration. The following subsections

describe each relevant equation, along with the intuition and performance implications behind them.

A. Definitions of Variables

TABLE I
NOTATION USED IN ANALYTICAL MODELS

Symbol	Description
b	Block size (bits)
m	Payload per LTP segment (bits)
f	Fraction of data requiring reliable (red-part) delivery
L	Length of link-layer frame (bits), including headers
r	Link transmission rate (bits/s)
d	One-way propagation delay (seconds)
T_o	Timeout duration, typically $T_o = 2d + \xi$
ξ	Timer margin or processing slack (seconds)
p_e	Bit error rate (BER)
p_f	Frame loss probability
p	Segment loss probability
p^*	Timeout probability per round (loss of CP or RS)
κ	Average number of transmission rounds

B. Segment Loss Probability

$$p_f = 1 - (1 - p_e)^L \tag{1}$$

This equation expresses the probability that a frame of length L bits is lost due to independent bit errors, where p_e is the bit error rate. Since each LTP segment typically maps to a single frame, the segment loss probability p is directly approximated by p_f. As L increases, p_f increases exponentially, creating a trade-off: larger MTU values improve efficiency but also raise retransmission risk. [5]

C. Segment Transmission Time T_r

$$T_r = \frac{L}{r} \cdot \frac{b}{m} \left[(1 - f) + \frac{f}{1 - p} \right] \tag{2}$$

This formula estimates the total time to transmit a block of size b bits. The first term $\frac{L}{r}$ is the time to transmit one segment. The term $\frac{b}{m}$ is the number of segments. The term $(1 - f) + \frac{f}{1-p}$ reflects that red-part data (fraction f) must be retransmitted on average $1/(1 - p)$ times. As p increases, red-part overhead dominates. [6]

D. Retransmission Penalty Time T_p

$$T_p = (\kappa - 1) \left[(1 - p^*) \cdot 2d + p^* T_o \right] = (\kappa - 1)(2d + p^* \xi) \tag{3}$$

This equation quantifies the penalty caused by ARQ retransmission rounds beyond the first. A round completes with either a successful RS arrival (delay $2d$) or a timeout (delay $T_o = 2d + \xi$). The expected penalty grows with the number of rounds κ, and worsens with larger p^* or link delay d. [7]

E. Expected Number of Rounds κ

$$\kappa = \frac{S_n}{1 - p^*}, \quad \text{where } n = \frac{fb}{m} \tag{4}$$

This equation models the expected number of rounds to complete transmission of n red segments. S_n is the expected number of segment-level attempts before success, based on a Markov chain model. If p^* is large (e.g., high loss), the denominator shrinks and κ increases significantly. [8]

F. Delivery Time and Goodput

$$T_{\text{del}} = T_r + T_p + d \tag{5}$$

$$G = \frac{b}{T_{\text{del}}} \tag{6}$$

The delivery time combines the transmission time, retransmission delay, and final one-way latency. Goodput is defined as the delivered payload per unit time. Optimizing goodput requires not just increasing r, but also minimizing retransmissions and idle timeout periods.

G. Design Implications

- Larger MTU reduces header overhead but increases L and thus frame loss probability.
- High propagation delay d significantly amplifies retransmission costs.
- Timer tuning (e.g., ξ) is critical to avoid long T_p due to premature or late retransmissions.
- Transmission rate r helps reduce T_r but not T_p if p^* remains high.
- Session-level concurrency is not directly included here but improves pipeline efficiency.

III. Simulation Setup

To evaluate the impact of LTP configuration parameters on transmission performance, we employed a Python-based event-driven simulator originally developed by Lent. [6]. The simulator models the core behavior of the LTP protocol, including segment delivery, checkpoint/report signaling, and retransmission timeout logic, in alignment with the principal state transitions and mechanisms described in RFC 5326 [9]. The simulator has been widely used in previous analytical studies of LTP and is well suited for examining protocol behavior under varying network conditions.

A. Network Configuration

The simulation models a unidirectional deep-space link between two DTN nodes communicating over a full-duplex, loss-prone channel. Bundle-level data flow is generated via the `BPDATA` application, which transmits 300 bundles per trial, each of size 200 kB, from sender to receiver. The simulator operates on a fixed event-driven loop, capturing segment- and block-level timing, and uses

a constant bit error rate (BER) of 10^{-7} with a fixed one-way propagation delay of 10 seconds. Forward error correction (FEC) and bundle-layer retransmission are disabled to isolate pure LTP-layer behavior. Each simulation scenario is repeated 30 times with a fixed random seed to ensure reproducibility. The MTU, session count, and transmission rate were varied independently during parameter sweeps, while all other values were held constant for fair comparison.

LTP operates in standard red/green mode with selective acknowledgment. The timeout for checkpoint and report segments is defined as:

$$T_o = 2d + \xi,$$

where $d = 10$ seconds and $\xi = 1$ second is an additional guard margin. Bundle-layer retransmission is disabled to allow pure LTP-layer dynamics to be evaluated.

B. Performance Metrics

Each simulation run captures the following performance metrics:

- **Goodput** [B/s]: Ratio of successfully received payload to total simulation time.
- **95th percentile file reception time (p95 RT)**: Time by which 95% of files were fully received, excluding the initial 2% for warm-up.
- **Protocol Overhead** [%]: Ratio of total transmitted bytes (including headers and retransmissions) to received payload bytes.

Each scenario was executed 30 times, and the reported values are the mean with standard error bars to account for variability.

C. Parameter Sweep Strategy

To assess the individual impact of each configuration parameter on LTP performance, we adopted a one-at-a-time (OAAT) parameter sweep approach. In each experiment, only a single parameter was varied across a predefined range while all other parameters were held constant at their baseline values. This approach ensures that the observed performance variations can be attributed solely to the parameter under investigation, thereby enabling controlled and interpretable analysis of protocol sensitivity.

The three parameters evaluated were:

- **Number of LTP Sessions:** {2, 4, 6, 8, 10, 12, 16}
- **MTU Sizes [Bytes]:** {1024, 2048, 4096, 8192, 16384, 32000}
- **Transmission Rate Cap [bps]:** {64k, 128k, 256k, 512k, 1M}

In all experiments, unless otherwise specified, the remaining parameters were held at their default values: 8 sessions, 4096-byte MTU, 128 kbps transmission rate, BER of 10^{-7}, and file size of 200 kB. The simulator collected per-file and per-session logs to support post-analysis of retransmission behavior and latency characteristics.

IV. Simulation Results and Comparison

This section presents experimental results analyzing the impact of three key LTP parameters—session count, MTU size, and transmission rate—on goodput, end-to-end delay, and protocol overhead. Each result is interpreted in the context of the analytical model from Section 3.

A. Impact of LTP Sessions

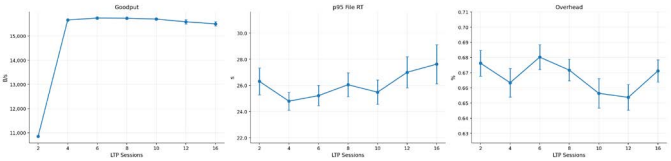

Fig. 1. Performance metrics as a function of LTP session count (MTU = 4096 B)

Figure 1 shows that increasing the number of concurrent LTP sessions from 2 to 4 significantly improves goodput, which rises from approximately 10.9 kB/s to 15.7 kB/s. Beyond 6 sessions, the goodput curve flattens, indicating diminishing returns due to saturated buffer or channel utilization.

Although session count is not explicitly included in Equation (2), its influence can be interpreted as reducing idle transmission intervals, effectively lowering T_r in the goodput expression $G = b/T_{\text{del}}$ (Equation (5)). The p95 reception time remains stable (24–27 seconds), implying that session-level parallelism does not induce latency penalties. Protocol overhead remains low between 0.66–0.68%, confirming that control segment cost remains minimal.

Overall, a configuration of 6–8 sessions provides the best trade-off between throughput gain and control overhead.

B. Impact of MTU Size

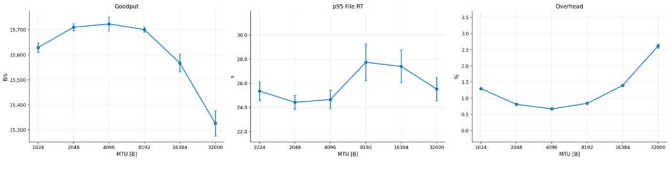

Fig. 2. Performance metrics as a function of MTU size

Figure 2 demonstrates that as MTU increases from 1024 B to 4096 B, goodput improves moderately while overhead decreases significantly—reflecting better per-segment efficiency due to reduced header-to-payload ratio. However, for MTUs beyond 4096 B, goodput declines and both p95 delay and overhead rise sharply, with overhead reaching 2.6% at 32 kB.

This reversal is well-explained by Equation (1), where the segment loss probability p_f increases exponentially with segment length L. Larger MTUs result in higher p^*, leading to more retransmissions, an increased number of

rounds κ (Equation (5)), and consequently longer retransmission delays T_p (Equation (4)).

Therefore, although larger MTUs initially improve efficiency, the retransmission penalty becomes dominant beyond 4096 B. In our scenario, 4096 B yields the optimal balance between efficiency and reliability.

C. Impact of Transmission Rate

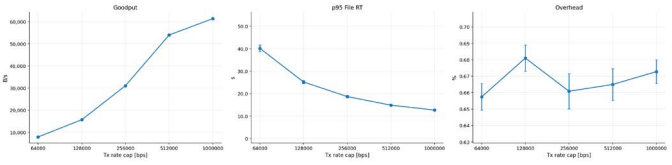

Fig. 3. Performance metrics as a function of physical-layer transmission rate (MTU = 4096 B)

Figure 3 presents the effect of increasing the channel transmission rate from 64 kbps to 1 Mbps. Goodput scales almost linearly, reaching 61 kB/s, while the p95 reception time drops significantly from 39.8 s to 13.4 s. This behavior is directly predicted by Equation (2), where T_r is inversely proportional to the transmission rate r.

Protocol overhead remains relatively constant (between 0.65% and 0.68%), indicating that signaling traffic does not scale with data rate. These results confirm that the system operates in a bandwidth-limited regime, and that the LTP implementation does not encounter CPU or memory bottlenecks in this range.

D. Summary and Interpretation

TABLE II
Summary of Parameter Effects on LTP Performance (MTU = 4096 B)

Metric	Sessions	MTU [B]	TxRate [bps])
Range	2–16	1k–32k	64k–1M
Optimal Value	6–8	4096	≥512k
ΔGoodput	+45%	+2%	+650%
Δp95 RT	±2 s	+5 s	–27 s
ΔOverhead	±0.01%	–0.6% → +2.6%	±0.03%

The best performance under our test settings (delay = 10 s, BER = 10^{-7}, 200 kB file size) was observed with:

- LTP session count of 6–8,
- MTU size of 4096 B,
- Transmission rate $\geq$ 512 kbps.

This configuration yields stable performance with approximately 15.7 kB/s goodput, ∼0.85% overhead, and a p95 reception time of ∼25 seconds.

V. Discussion and Conclusion

A. Discussion

Through a comprehensive simulation-based evaluation, this study confirmed that the performance of LTP is significantly influenced by its configuration parameters. Each parameter—session count, MTU size, and transmission rate—was shown to have a measurable and interpretable effect on key performance indicators such as goodput, delay, and overhead.

The results show that:

- **Session count** has a strong positive impact on goodput up to a saturation point (approximately 6–8 sessions), beyond which the benefit plateaus and control overhead slightly increases.
- **MTU size** affects both efficiency and reliability. While increasing the MTU to 4096 B improves header efficiency and goodput, further increases degrade performance due to higher retransmission likelihood—consistent with the exponential sensitivity to segment size predicted in Equation (1).
- **Transmission rate** exhibits nearly linear scaling of goodput and sharply reduces end-to-end delay. Overhead remains constant, confirming that LTP operates bandwidth-limited under these conditions.

These results not only empirically validate the analytical model presented earlier but also provide practical insights into how LTP parameter tuning impacts protocol performance. More importantly, they suggest that intelligent control of these parameters can lead to significant performance gains. In other words, if such parameters can be dynamically monitored and adaptively controlled, the overall system throughput and reliability can be substantially optimized—especially in deep-space and delay-tolerant scenarios.

B. Conclusion

This study conducted a detailed empirical analysis of how LTP protocol parameters affect communication performance. By independently varying session count, MTU size, and transmission rate under a fixed simulation environment, we verified their distinct and quantifiable effects on throughput, delay, and protocol overhead. The findings reinforce that precise and adaptive configuration of LTP can be crucial for ensuring high performance in challenging network environments.

Looking forward, future research can build upon this transmission model and parameter control framework by integrating recent advances in intelligent optimization techniques. In particular, multi-agent deep reinforcement learning (MADRL) approaches for autonomous cooperative control, such as those demonstrated in UAV coordination systems [10], could be adapted to enable environment-aware, dynamic tuning of LTP parameters.

Furthermore, the recent work by Moon et al. [11] demonstrates how deep reinforcement learning (DRL) can be used to optimize communication performance in complex and dynamic environments, such as RIS-assisted ISAC-UAV networks. By employing DRL to adaptively configure beamforming and reflection coefficients under varying channel conditions, the proposed framework achieves significant gains in sum-rate while maintaining

sensing quality. Inspired by this, similar DRL-based control strategies could be adapted for LTP parameter tuning, where environmental variability (e.g., delay, BER, session load) can be learned and acted upon in real time for adaptive optimization of protocol behavior in deep-space communications.

These approaches could help transform LTP configuration from a static, rule-based process into a dynamic, learning-based control problem—supporting long-term reliability and adaptability in future deep-space and interplanetary communication missions.

Acknowledgment

This work was supported in part by the IITP (Institute of Information & Communications Technology Planning & Evaluation) - ITRC (Information Technology Research Center) (IITP-2026-RS-2022-00156353, 50%) grant funded by the Korea government (Ministry of Science and ICT) and in part by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. RS-2023-00209125).

References

[1] S. Burleigh *et al.*, "Delay-tolerant networking: An approach to interplanetary internet," *IEEE Communications Magazine,* vol. 41, no. 6, pp. 128–136, Jun. 2003.

[2] V. Cerf *et al.*, "Delay-tolerant network architecture," Internet Engineering Task Force (IETF), RFC 4838, Apr. 2007, informational RFC. [Online]. Available: https://www.rfc-editor.org/info/rfc4838

[3] S. Burleigh, M. Ramadas, and S. Farrell, "Licklider transmission protocol – motivation," Internet Engineering Task Force (IETF), RFC 5325, Sep. 2008. [Online]. Available: https://www.rfc-editor.org/info/rfc5325

[4] R. Wang, S. C. Burleigh, P. Parikh, C.-J. Lin, and B. Sun, "Licklider transmission protocol (ltp)-based dtn for cislunar communications," *IEEE/ACM Transactions on Networking,* vol. 19, no. 2, pp. 359–368, Apr. 2011.

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

[6] R. Lent, "Analysis of the block delivery time of the licklider transmission protocol," *IEEE Transactions on Communications,* vol. 67, no. 1, pp. 518–526, Jan. 2019.

[7] G. Yang, R. Wang, S. C. Burleigh, K. Zhao *et al.*, "Analysis of licklider transmission protocol (ltp) for reliable file delivery in space vehicle communications with random link interruptions," *IEEE Transactions on Vehicular Technology,* vol. 68, no. 4, pp. 3919–3932, Apr. 2019.

[8] Y. Guo, Z. Dong, and Y. Zhu, "Performance improvement of licklider transmission protocol in complex deep space networks based on parameter optimization," *Chinese Journal of Aeronautics,* vol. 36, no. 5, pp. 406–420, May 2023.

[9] M. Ramadas, S. Burleigh, and S. Farrell, "Licklider transmission protocol – specification," Internet Engineering Task Force (IETF), RFC 5326, Sep. 2008. [Online]. Available: https://www.rfc-editor.org/info/rfc5326

[10] W. J. Yun *et al.*, "Cooperative multiagent deep reinforcement learning for reliable surveillance via autonomous multi-uav control," *IEEE Transactions on Industrial Informatics,* vol. 18, no. 10, pp. 7086–7096, Oct. 2022.

[11] S. Moon, C.-G. Lee, H. Liu, and I. Hwang, "Deep reinforcement learning-based sum rate maximization for ris-assisted isac-uav network," *ICT Express,* vol. 10, no. 5, pp. 1174–1178, Oct. 2024.