UNIVERSIDADE FEDERAL DO MARANHÃO
Programa de Pós-Graduação em Engenharia Elétrica
Centro de Ciências Exatas e Tecnologia - CCET

Compressão sem Perdas de Dados de Telemetria Satelital usando o Algoritmo On+

José Flávio Gomes Barros

São Luís - MA
07/2026

José Flávio Gomes Barros

Compressão sem Perdas de Dados de Telemetria Satelital usando o Algoritmo On+

Tese submetida ao Programa de Pós-Graduação em Engenharia de Eletricidade da Universidade Federal do Maranhão como parte dos requisitos necessários à obtenção do grau de Doutor em Engenharia Elétrica.

Orientador: Prof. Dr. Allan Kardec Duailibe Barros Filho

São Luís - MA

07/2026

Ficha gerada por meio do SIGAA/Biblioteca com dados fornecidos pelo(a) autor(a).
Diretoria Integrada de Bibliotecas/UFMA

Barros, José Flávio Gomes.
Compressão sem Perdas de Dados de Telemetria Satelital usando o Algoritmo On+ / José Flávio Gomes Barros. - 2026.
59 f.

Orientador(a): Allan Kardec Duailibe Barros Filho.
Tese (Doutorado) - Programa de Pós-graduação em Engenharia Elétrica/ccet, Universidade Federal do Maranhão, Ccet/ufma, 2026.

1. Algoritmo de Compressão. 2. Compressão Sem Perdas. 3. Codificação Por Entropia. 4. Satélites Cubesat. 5. Dados de Telemetria. I. Barros Filho, Allan Kardec Duailibe. II. Título.

José Flávio Gomes Barros

Compressão sem Perdas de Dados de Telemetria Satelital usando o Algoritmo On+

Tese submetida ao Programa de Pós-Graduação em Engenharia de Eletricidade da Universidade Federal do Maranhão como parte dos requisitos necessários à obtenção do grau de Doutor em Engenharia Elétrica.

Trabalho aprovado, São Luís - MA, 11/07/2026.

Prof. Dr. Allan Kardec Duailibe Barros Filho
Orientador

Prof. Dr. Vicente Leonardo Paucar Casas
Avaliador Interno

Prof. Dr. Ewaldo Eder Carvalho Santana
Avaliador Interno

Prof. Dr. Jonathan Araújo Queiroz
Avaliador Externo

Profa. Dra. Marta de Oliveira Barreiros
Avaliador Externo

São Luís - MA
07/2026

Dedico este trabalho a Deus, fonte de força e sabedoria; às minhas filhas, Layna Talita e Larha Theresa; à minha esposa, Helanne Cristina; à minha mãe, Francisca Antônia; à minha irmã, Ithayara Gomes; aos meus sobrinhos, José Felipe e João Lucas; aos meus avós, in memoriam; e aos meus amigos, pelo apoio e incentivo ao longo desta jornada.

Agradecimentos

Agradeço ao professor Allan Kardec, exemplo de dedicação à pesquisa, entusiasmo pela ciência e compromisso com a formação acadêmica.

À minha família e amigos, pelo apoio e suporte que sempre me deram para concluir mais essa jornada.

Aos amigos de pesquisa e professores Caio Magno, Luis Claudio e Letícia Correia, pelo constante apoio, pela amizade, pelas valiosas contribuições científicas e pela generosidade em compartilhar conhecimentos e experiências durante o desenvolvimento deste trabalho.

Ao meu amigo e irmão Christian Diniz, pelo constante apoio, pela amizade, pela confiança e pelas valiosas palavras de incentivo que contribuíram para esta conquista.

Aos amigos(as) do Laboratório de Processamento da Informação Biológica (PIB), por proporcionarem, além de ideias e contribuições relevantes à pesquisa, momentos de sorrisos e compartilhamento de experiências.

"Tropeçamos, até caímos, nunca desista de suas conquistas."

Resumo

A telemetria de dados de satélite compreende o conjunto de informações necessárias para monitorar, controlar e garantir a operação segura e eficiente de missões espaciais. A redução do volume desses dados, transmitidos do satélite para estações terrestres, é denominada compressão do fluxo de telemetria. A compressão sem perdas aplicada a esses dados é essencial para garantir a integridade dos dados originais. Essa abordagem explora principalmente a redundância estatística presente nos dados para reduzir o volume de informação transmitida ou armazenada. Nesse contexto, a compressão sem perdas torna-se ainda mais relevante, dadas as severas restrições de largura de banda nos enlaces de comunicação e a limitada capacidade de armazenamento a bordo dos satélites. Este estudo propõe o On+, um novo algoritmo de compressão sem perdas para dados de telemetria de satélite. Utilizando dados reais de telemetria capturados pelo satélite CubeSat Aldebaran-1 e da plataforma Ground Station TinyGS, foi criado um conjunto de dados composto por 2500 arquivos binários. O desempenho do algoritmo proposto foi avaliado em comparação com métodos clássicos (codificação de Huffman e Aritmética) e diversos compressores comerciais (.rar, .zip, .7z, .xz e .gz). O algoritmo On+ alcançou uma taxa média de compressão de 29,19% nos dados do satélite Aldebaran-1, com desvio padrão de 1,26% e mediana de 29,09%, superando a performance dos métodos avaliados em relação a taxa de compressão. Para os satélites da plataforma TinyGS, o On+ também apresentou resultados expressivos, alcançando uma taxa de compressão mediana próxima de 50% em classes específicas de satélites, com destaque para o Starlink. Esse resultado demonstra a capacidade do método em explorar eficientemente as características estatísticas de determinados conjuntos de dados de telemetria, obtendo ganhos substanciais de compressão.

Palavras-chave: Algoritmo de compressão; Compressão sem perdas; Codificação por entropia; Satélites CubeSat; Dados de telemetria.

Abstract

Satellite data telemetry comprises the set of information required to monitor, control, and ensure the safe and efficient operation of space missions. The reduction in the volume of this data, transmitted from the satellite to ground stations, is called telemetry stream compression. Lossless compression applied to this data is essential to guarantee the integrity of the original data. This approach primarily exploits the statistical redundancy present in the data to reduce the volume of information transmitted or stored. In this context, lossless compression becomes even more relevant, given the severe bandwidth constraints on communication links and the limited on-board storage capacity of satellites. This study proposes On+, a new lossless compression algorithm for satellite telemetry data. Using real telemetry data captured by the Aldebaran-1 CubeSat satellite and the TinyGS Ground Station platform, a dataset consisting of 2500 binary files was created. The performance of the proposed algorithm was evaluated in comparison with classical methods (Huffman coding and Arithmetic coding) and various commercial compressors (.rar, .zip, .7z, .xz, and .gz). The On+ algorithm achieved an average compression ratio of 29.19% on Aldebaran-1 satellite data, with a standard deviation of 1.26% and a median of 29.09%, outperforming the evaluated methods in terms of compression ratio. For satellites on the TinyGS platform, On+ also delivered significant results, achieving a median compression ratio close to 50% in specific satellite classes, with Starlink standing out in particular. This result demonstrates the method's ability to efficiently exploit the statistical characteristics of certain telemetry datasets, obtaining substantial compression gains.

Keywords: Compression Algorithm; Lossless Compression; Entropy Coding; CubeSat Satellites; Telemetry Data.

Lista de ilustrações

Lista de tabelas

Sumário

1 Introdução

O processo de redução do tamanho dos dados transmitidos de um dispositivo, como um satélite, para uma estação em solo é conhecido como compressão de fluxo de telemetria (ou compressão de dados de housekeeping[1]). Para armazenamento ou transmissão de dados, a compressão de fluxos de telemetria é crucial, especialmente em situações nas quais a energia (Guojun et al., 2013; Ketshabetswe et al., 2024) e largura de banda são limitadas (Ciaparrone et al., 2022). Por meio da compressão, a redução do volume de dados leva a menores custos de transmissão, uma vez que tanto o tempo de transmissão quanto o uso da largura de banda são frequentemente onerosos (Rakhmanov and Wiseman, 2023).

Segundo Meß et al. (2017), o monitoramento da saúde de um satélite é fundamental para garantir seu funcionamento adequado. Esse processo envolve a coleta contínua de dados sobre seu status e desempenho, abrangendo parâmetros como níveis de carga da bateria, eficiência dos painéis solares, temperatura de componentes críticos, bem como o desempenho dos sistemas de propulsão e comunicação, incluindo sua eficiência e integridade (Evans et al., 2022). Adicionalmente, dados de navegação e controle de atitude, como a posição e orientação do satélite, também são monitorados. Nesse contexto, a transmissão de dados de telemetria por satélites, conforme discutido por (Anmireddy et al., 2010; Meß et al., 2016; Xu et al., 2023b,a), desempenha um papel crucial, pois possibilita a manutenção, a operação eficiente e o sucesso das missões espaciais (Wan et al., 2019; Shehab et al., 2020; Evans et al., 2022).

Os dados de telemetria são essenciais para a operação diária de um satélite, pois dão suporte às operações da missão (Meß et al., 2017) ao permitir o monitoramento e controle do satélite, bem como ajustes de órbita, mudanças de orientação e gerenciamento da carga útil (Anmireddy et al., 2010; Meß et al., 2016). Consequentemente, o monitoramento da telemetria é crucial para garantir que o satélite opere dentro de parâmetros seguros (Rakhmanov and Wiseman, 2023; Hernández-Cabronero et al., 2024), prevenindo falhas potenciais como superaquecimento ou até mesmo perda de controle (Hao et al., 2020).

Em diversos cenários, a compressão sem perdas pode não proporcionar benefícios substanciais, conforme evidenciado por certos métodos existentes. Em alguns casos, o processo pode até levar a um aumento no tamanho do arquivo devido à incorporação de informações de controle ou à sobrecarga adicional (denominada *overhead*[2]) associada ao processo de compressão (Jeong et al., 2023). Essa sobrecarga pode surgir de múltiplos

[1] No contexto de satélites e missões espaciais, "housekeeping" refere-se ao conjunto de atividades e dados necessários para manter a operação segura e eficiente da espaçonave. Isso inclui o monitoramento e o controle de vários parâmetros e sistemas embarcados.

[2] O termo "overhead" está relacionado a qualquer custo adicional associado à implementação de técnicas de compressão em um conjunto de dados.

fatores, incluindo a execução de etapas auxiliares de processamento, a inclusão de dados de controle e, em algumas instâncias, a necessidade de atender a requisitos temporários de armazenamento. Consequentemente, o tempo e os recursos computacionais necessários para realizar essas operações são considerados componentes integrais da sobrecarga geral de compressão (Wang et al., 2023). As informações de controle tipicamente incluem cabeçalhos, metadados e outros dados auxiliares necessários para reconstruir com precisão a estrutura original dos dados.

1.1 Motivação

A compressão de dados sem perdas tem se tornado cada vez mais relevante no setor aeroespacial, especialmente diante do aumento exponencial da quantidade de dados gerados por satélites e outras missões espaciais. A telemetria de satélites produz volumes expressivos de dados que precisam ser transmitidos para estações em solo e armazenados em ambientes com recursos computacionais limitados, como memória, capacidade de processamento e largura de banda.

Dessa forma, a compressão sem perdas constitui uma classe de algoritmos que possibilita a reconstrução exata dos dados originais a partir dos dados comprimidos. Essa característica é essencial em aplicações aeroespaciais, nas quais a perda de informações pode comprometer análises críticas, diagnósticos de falhas e os processos de tomada de decisão em solo. Portanto, a preservação da integridade dos dados de telemetria é um requisito fundamental para garantir a confiabilidade das operações e das análises realizadas.

As vantagens da compressão sem perdas aplicada ao setor aeroespacial incluem:

- Redução do tamanho dos dados de telemetria, resultando em economia de espaço de armazenamento a bordo do satélite, onde os recursos são extremamente limitados e o custo de memória é elevado;
- Transmissão mais rápida e eficiente dos dados entre o satélite e as estações em solo, otimizando o uso da largura de banda disponível, que é compartilhada com outras telemetrias e comandos;
- Possibilidade de aumentar a frequência e a quantidade de dados coletados sem sobrecarregar o link de comunicação, permitindo maior retorno científico e operacional por missão;
- Viabilização do armazenamento de longas séries históricas de telemetria em solo, essenciais para análises preditivas, detecção de anomalias e estudos de envelhecimento de componentes espaciais.

Diante desse cenário, o desenvolvimento de modelos matemáticos e computacionais para compressão sem perdas de dados de telemetria de satélites mostra-se fundamental para o avanço das missões espaciais, contribuindo para o uso mais eficiente dos recursos embarcados e para o aumento da capacidade de transmissão e armazenamento de informações.

Apesar dos avanços obtidos pelos métodos de compressão sem perdas aplicados a dados de telemetria espacial, ainda existem desafios relacionados à obtenção de maiores taxas de compressão, à redução da complexidade computacional e à adaptação a diferentes perfis de dados gerados por sistemas espaciais. Ademais, diversas abordagens apresentam limitações quanto à sua capacidade de serem aplicadas a diferentes plataformas e missões.

Com base nesse contexto, esta pesquisa busca responder à seguinte questão: como a aplicação de transformações geométricas pode melhorar o desempenho da compressão sem perdas em dados de telemetria de satélites, de modo que possam ser aplicados na indústria aeroespacial?

1.2 Objetivos

1.2.1 Objetivo Geral

Este estudo tem como objetivo criar um algoritmo de compressão sem perdas especificamente projetado para dados de telemetria de satélite, com o propósito de alcançar taxas de compressão superiores aos métodos de referência na literatura (como Huffman e Aritmético) e compressores comerciais padrão.

1.2.2 Objetivos Específicos

Para se atingir o objetivo geral, faz-se necessário o cumprimento dos itens relacionados a seguir:

- Realizar um estudo aprofundado e uma análise detalhada de algoritmos de compressão sem perdas, explorando suas características, limitações e potencial de otimização;
- Analisar modelos matemáticos como base para o desenvolvimento de algoritmos de compressão sem perdas aplicados a dados de telemetria de satélites;
- Desenvolver um algoritmo para compressão sem perdas de dados de telemetria de satélites, com atenção especial às características únicas e aos requisitos técnicos inerentes a esse tipo de dado;

- Realizar uma série de testes e validações para comparar o desempenho do algoritmo proposto com os métodos existentes na literatura, avaliando sua eficácia e superioridade em cenários reais.

Além da proposição do algoritmo de compressão, esta pesquisa contribui com a construção de um repositório experimental contendo dados reais de telemetria espacial, utilizado para a validação dos experimentos e para a análise comparativa com métodos de compressão existentes. O repositório é composto por dados extraídos do satélite Aldebaran-1 (vinculado à Missão Aldebaran da Universidade Federal do Maranhão) e da plataforma *Ground Station TinyGS*.

1.3 Principais Contribuições da Tese

Durante o desenvolvimento desta tese, até o momento da defesa, as seguintes contribuições científicas foram alcançadas:

- Desenvolvimento de um modelo matemático e computacional para compressão sem perdas de dados de telemetria de satélites, baseado na Transformação Geométrica;
- Análise comparativa do algoritmo On+ com métodos tradicionais (Huffman e Aritmético) e compressores comerciais (.rar, .zip, .7z, .xz e .gz), aplicados a dados reais de missões espaciais;
- Análise de performance do algoritmo On+ utilizando dados do satélite Aldebaran-1 e de diversos satélites da plataforma *TinyGS*;
- Publicação do artigo intitulado *"Lossless Compression of Aldebaran-I Telemetry Data Using the On+ Algorithm"* no periódico *Technologies (MDPI)* (Barros et al., 2026).

1.4 Trabalhos Relacionados

Vários estudos investigaram métodos de compressão de dados sem perdas para dados de telemetria de satélites, buscando acelerar o processo de compressão, visando particularmente a redução da quantidade de informação, de modo que, no destino, uma cópia exata dos dados originais seja recuperada, ou seja, a compressão é reversível. Esses estudos focam, entre outras técnicas, na otimização do processo de compressão, incluindo autoaprendizagem (Wan et al., 2019), predição baseada em redes neurais e codificação de entropia (Shehab et al., 2020), algoritmos baseados em árvores de decisão (Hao et al., 2020) e o uso de recursos de inteligência artificial (Ciaparrone et al., 2022).

O trabalho de Wan et al. (2019) conduziu um estudo sobre classificação de dados de telemetria de satélites baseada em aprendizado de máquina. Quatro classes básicas de dados de telemetria foram sugeridas e estudadas utilizando características de séries temporais e análise de entropia da informação. Foi proposto um algoritmo leve de autoaprendizagem para aplicação embarcada, denominado Cálculo de Probabilidade de Classificação com Otimização de Janela por Passos (CP-WS), com o objetivo de obter as características das classes e tomar a decisão de cada parâmetro individual a partir da série temporal discreta e contínua de telemetria. Os resultados das simulações mostram que o algoritmo classifica corretamente os dados simulados e os dados reais da missão na classe básica apropriada, apresentando como vantagem uma alta precisão de classificação de 100%.

Shehab et al. (2020) apresentou uma predição baseada em redes neurais recorrentes para melhorar a compressão de telemetria de satélites. O método proposto consiste em uma etapa de decorrelação e uma etapa de codificação de entropia. Os resultados experimentais mostram que o método proposto melhorou a eficiência de compressão baseada em um único estágio de codificador.

Hao et al. (2020) propôs um método de detecção e diagnóstico de falhas em satélites baseado em compressão de dados e em uma árvore de decisão melhorada. O método aumentou a precisão e a eficiência do diagnóstico e detecção de falhas em satélites, além de economizar recursos computacionais e melhorar significativamente a eficiência do treinamento do modelo. Ao mesmo tempo, também é capaz de resistir ao sobreajuste (overfitting) e aumentar a precisão da detecção. Entretanto, a baixa velocidade de processamento e o alto custo computacional são pontos críticos da proposta.

Um estudo preliminar foi realizado com o objetivo de verificar a eficácia de um dos algoritmos de compressão de telemetria sem perdas baseados em inteligência artificial mais representativos em três diferentes conjuntos de dados da NASA (Ciaparrone et al., 2022). No entanto, a proposta apresenta desempenho inferior ao de outros métodos.

Também em 2022, um novo padrão de compressão de dados de housekeeping, o CCSDS 124.0-B-1 (baseado no POCKET+), foi implementado no OPS-SAT-1 (Evans et al., 2022). O algoritmo foi implementado utilizando instruções de processador de nível muito baixo, como OR, XOR, AND, entre outras. Além disso, o algoritmo executa com baixo uso de CPU e, mais importante, com curto tempo de execução. Porém, as taxas de compressão dependem do tipo de dados a serem comprimidos.

Por fim, Hernández-Cabronero et al. (2024) investigaram a resiliência e a eficiência do padrão de compressão de telemetria CCSDS 124.0-B-1. O trabalho apresenta uma introdução acessível ao padrão, com ênfase em suas características de resiliência. Além disso, foram propostas duas estratégias complementares de ressincronização de decodificadores com o objetivo de maximizar a quantidade de dados corretamente recuperados após a ocorrência de erros. Os resultados demonstraram desempenho competitivo em termos de

compressão, sendo superados apenas pelo algoritmo bzip2. Contudo, esse método não oferece mecanismos de resiliência à perda de dados, nem suporte à compressão em tempo real ou baixo consumo de memória, características essenciais para o processamento de dados de telemetria de sensoriamento remoto e *housekeeping*.

1.5 Uso de IA Generativa

As plataformas DeepSeek V4 Pro, Grok 4.20 Beta e GPT-5.3 Instant foram empregadas apenas para revisão linguística e aperfeiçoamento editorial de trechos escritos pelo autor. Estes recursos foram utilizados apenas para avaliar a clareza textual, à sugestão de alternativas de redação, ao refinamento de terminologia técnica e à identificação de possíveis ambiguidades, preservando-se integralmente o significado e o conteúdo originais. Todo o texto e o conteúdo científico desta tese, que abrange desde a concepção da pesquisa, passando pela modelagem matemática, pelo desenvolvimento dos algoritmos e pela implementação computacional, até a análise dos resultados, suas interpretações e conclusões, foram elaborados pelo autor.

1.6 Estrutura do Trabalho

Diante do supracitado, esta tese foi projetada para oferecer uma compreensão abrangente do assunto. O Capítulo 1 apresenta a contextualização e a motivação da pesquisa, define o problema de pesquisa, delineia os objetivos do estudo, destaca as principais contribuições da tese e apresenta os principais trabalhos relacionados. O Capítulo 2 apresenta o referencial teórico, abordando os conceitos fundamentais da compressão de dados, incluindo informação, dado e código, o limite teórico da compressão (entropia), os tipos de compressão (com e sem perda), os principais algoritmos clássicos de compressão sem perda, como os códigos de Huffman e Aritmético, bem como os fundamentos da transformação baseada na distribuição geométrica. A Metodologia Proposta é apresentada no Capítulo 3, cuja execução foi dividida em três etapas: análise e desenvolvimento do algoritmo On+, investigação dos indicadores de qualidade de compressão, e realização das validações e experimentos. O Capítulo 4 apresenta os resultados obtidos na pesquisa. Por meio do *benchmarking* do algoritmo On+, foi possível validar a proposta e compará-la com métodos tradicionais e comerciais, além de realizar a análise de complexidade temporal e espacial. O Capítulo 5 traz discussões acerca do trabalho, baseadas nos resultados obtidos no capítulo anterior. Por fim, no Capítulo 6, são feitas as considerações finais da pesquisa, além de abordadas as perspectivas futuras do presente trabalho.

2 Referencial Teórico

Este capítulo aborda, inicialmente, os conceitos fundamentais da teoria da compressão de dados, tais como informação, dados e códigos. Em seguida, apresenta o limite teórico da compressão de dados, representado pela entropia, bem como a classificação das principais técnicas de compressão. Posteriormente, são apresentados os algoritmos clássicos de compressão sem perdas, foco principal deste trabalho, com destaque para os códigos de Huffman e Aritmético. Na sequência, são discutidas técnicas de transformação de dados aplicadas à compressão sem perdas, bem como os fundamentos da transformação baseada na distribuição geométrica, proposta como etapa de pré-processamento para a reorganização dos dados. Por fim, destaca-se que a compressão sem perdas é empregada em aplicações nas quais a integridade dos dados originais deve ser preservada, garantindo que nenhuma informação seja alterada ou perdida durante o processo de compressão.

2.1 Informação, Dados e Código

De acordo com (Pu, 2006), a informação é algo que agrega conhecimentos para as pessoas. É tudo o que contribui para a redução da incerteza da mente humana ou do estado de um sistema. Entretanto, as pessoas sentem a existência da informação, veem a mídia que carrega a informação e reagem de acordo com determinada informação o tempo todo.

Em 1948, Claude Shannon formalizou o conceito de informação em seu trabalho "*A Mathematical Theory of Communication*" (Shannon, 1948). Para Shannon, o processo de comunicação envolvia a transmissão de mensagens, provindas de uma fonte de informação, enviadas para um destinatário através de um canal, como mostra a Figura 1.

Neste contexto, a informação não é visível sem que algum meio seja um portador (Pu, 2006). Os dados são a mídia lógica geralmente transportada por alguma mídia física (por exemplo, um pendrive) ou por um canal de comunicação. Assim, os dados podem ser vistos como uma forma básica de alguma informação factual. Por exemplo, os dados " $-3.0°C$ " carregam a informação factual de "está frio". A mesma informação pode ser fornecida através do texto 'menos trinta graus centígrados' no papel, por uma imagem de um termômetro na tela do computador ou por advertência oral. Portanto, sem esses meios, nem os dados nem as informações seriam visíveis.

Além disso, os dados podem ser classificados como texto, áudio, imagem e vídeo, enquanto o formato de dados digitais reais consiste em $0s$ e $1s$ em um formato denominado de binário. Dados de texto são geralmente representados por código ASCII, que consiste

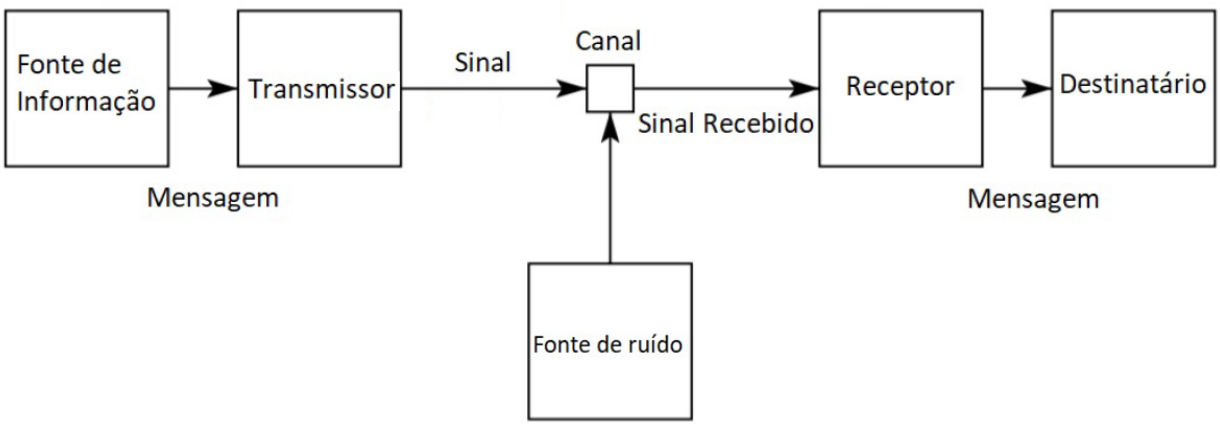

Figura 1 – Modelo de sistema de comunicação proposto por Shannon.
Fonte: adaptado de (Shannon, 1948).

em um conjunto de palavras-código de comprimento fixo (8 *bits*). Já dados de imagem são representados por uma matriz bidimensional de pixel que está associada ao seu código de cores. Outro tipo de dados são os gráficos, em formato de vetores ou equações matemáticas. E, os dados de som são representados por uma função de onda (periódica).

Em domínios de aplicativos, os dados de origem a serem compactados provavelmente são os chamados multimídia e podem ser uma mistura de formato de mídia estático, como texto, imagem e gráficos, e mídia dinâmica, como som e vídeo (Pu, 2006). A Figura 2 apresenta os estágios envolvidos para que os dados de origem em um arquivo de determinado tipo sejam codificados em um arquivo binário de origem antes da compactação.

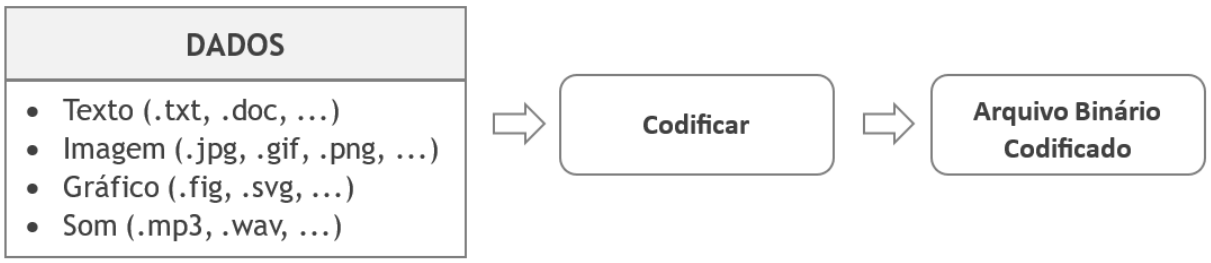

Figura 2 – Processo de codificação de dados.
Fonte: adaptado de (Pu, 2006).

Neste cenário, dado o alfabeto de uma fonte seja $S = \{s_1, s_2, ..., s_n\}$. A representação digital do conjunto de símbolos é chamada de código $C = (c_1, c_2, ..., c_n)$ e a representação c_j de cada símbolo é chamada de palavra-código para o símbolo s_j, onde $j = 1, 2, ..., n$. O processo de atribuir palavras-código a cada símbolo em uma fonte é chamado de codificação. O processo inverso, ou seja, reconstruir a sequência de símbolos na fonte, é chamado de decodificação.

Assim, é possível representar um alfabeto por um conjunto de palavras-código de comprimento fixo ou variável. Para exemplificar melhor, são dados dois códigos binários

diferentes, $C_1 = (000, 001, 010, 011, 100)$ e $C_2 = (0, 100, 11, 10, 111)$, usados para representar o alfabeto $A = \{A, B, C, D, E\}$ como pode ser observado na Figura 3, o alfabeto (símbolos) e o código (C_1). No entanto, C_1 é um código de comprimento fixo, pois todas as palavras de código consistem em 3 *bits*. Já C_2 é um código de comprimento variável, pois nem todas as palavras-código têm o mesmo comprimento.

Para Pu (2006) é importante distinguir o conceito de dados simbólicos, alfabeto e código no contexto de uma discussão. Assim, dados simbólicos é o termo usado para símbolos ou caracteres, significa a representação simbólica dos dados de entrada (origem), ou seja, são os símbolos de um alfabeto. Porém, um código contendo um conjunto de palavras-código é geralmente uma representação do conjunto do alfabeto.

A principal propriedade de um algoritmo de codificação é que os códigos produzidos sejam decodificáveis de forma única (Sayood, 2018), isto é, cada código só pode ser associado a um único símbolo do alfabeto. Um exemplo pode ser encontrado na Figura 3, que mostra a representação de dados simbólicos de um alfabeto (A, B, C, D, E). Cada símbolo representa um código binário de comprimento fixo (000, 001, 010, 011, 100).

Símbolos	Código
A	000
B	001
C	010
D	011
E	100

Figura 3 – Mapeamento dos símbolos do alfabeto para seus respectivos códigos binários de comprimento fixo.

Dado uma *string* CAABADAE, submetido a um processo de codificação baseado na Figura 3, terá como resultado o código binário 010 000 000 001 000 011 000 100, este sendo a representação binária dos dados simbólicos. Portanto, esses dados de origem em representação binária poderão ser inseridos em um algoritmo de compactação, ou seja, o arquivo de dados de origem binária deve ser reduzido, em caso de sucesso em um processo de compressão.

2.2 Compressão de Dados

A compressão de dados é um método de regras de codificação que permite a redução essencial do número total de *bits* para armazenar ou transmitir um arquivo (Patel et al., 2015). A compressão de dados é parte fundamental da Teoria da Informação, proposta

por Claude Shannon em 1948, abordada na Seção 2.1. De acordo com Shannon (1948), o objetivo da compressão de dados é reduzir informações redundantes. Em outras palavras, a redundância na informação coloca *bits* adicionais para a codificação e então remove o *bit* extra na informação e o tamanho da informação.

Falando em limites teóricos de compressão, de acordo com o teorema de codificação de fonte de Shannon, o limite teórico que se pode comprimir dados (sem perdas) é igual à sua entropia (Brian, 2020; Barros, 2025). Entretanto, o número médio de *bits* por símbolo da sua mensagem não pode ser menor que:

$$H(X) = -\sum_{i=1}^{n} \; p_i \;\; \log_2(p_i), \tag{2.1}$$

na qual se assume o conhecimento prévio de p_i, distribuição de cada um dos seus n símbolos antes do tempo. Observe que o logaritmo é de base 2, o que naturalmente nos permite falar em termos de *bits*.

O exemplo a seguir apresenta a entropia de uma distribuição de probabilidade discreta. Dado um alfabeto com 3 símbolos: $\{a, b, c\}$. Seja a variável aleatória X representada pela probabilidade do símbolo em uma mensagem, dada pela distribuição: $p_a = \frac{4}{7}$, $p_b = \frac{2}{7}$, e $p_c = \frac{1}{7}$.

A entropia e o número médio mínimo de *bits* por símbolo que podemos obter para mensagens com esta distribuição (segundo o teorema da codificação da fonte) é:

$$\begin{aligned} H(X) &= -\textstyle\sum_{i=1}^{n} \;\; p_i \;\; \log_2(p_i) \\ &= -p_a \;\; log_2(p_a) - p_b \;\; log_2(p_b) - p_c \;\; log_2(p_c) \\ &= -\tfrac{4}{7} \;\; log_2(\tfrac{4}{7}) - \tfrac{2}{7} \;\; log_2(\tfrac{2}{7}) - \tfrac{1}{7} \;\; log_2(\tfrac{1}{7}) \\ &\approx 1.3788 \;\; bits \end{aligned}$$

Ainda neste cenário, a compressão de dados pode ser com ou sem perdas. Desta forma, um método de compactação apresenta perdas se não for possível reconstruir o original exatamente a partir da versão compactada Pu (2006). Existem alguns detalhes insignificantes que podem se perder durante o processo de compactação. A Figura 4 mostra um exemplo em que um número decimal longo se torna uma aproximação mais curta após o processo de compressão e descompressão.

A compressão com perdas caracteriza-se pela irreversibilidade, uma vez que a reconstrução exata dos dados originais é impossível Pu (2006). Ainda assim, a reconstrução aproximada dos dados pode ser desejável em determinadas aplicações, pois permite

Figura 4 – Algoritmos de compressão com perdas.
Fonte: adaptado de (Pu, 2006).

alcançar maiores taxas de compressão. Por essa razão, dados como imagens, vídeos e áudios são frequentemente compactados por técnicas de compressão com perdas, devido às características dos sistemas visual e auditivo humanos.

Já as técnicas de compressão sem perdas são usadas quando os dados originais de uma fonte são tão importantes que nenhum detalhe pode ser perdido Patel et al. (2015). Exemplos de tais dados são imagens médicas, textos e imagens preservados por motivos legais, alguns arquivos executáveis de computador, entre outros. A Figura 5 apresenta um processo de compressão sem perdas.

Figura 5 – Algoritmos de compressão sem perdas.
Fonte: adaptado de Pu (2006).

A técnica sem perdas é usada quando os dados originais de uma fonte são tão importantes que não se pode perder nenhum detalhe (Pu, 2006; Patel et al., 2015). Além disso, objetiva a compressão de arquivo diminuindo a informação de tal forma que não haja perda ao descriptografar qualquer arquivo de volta para o arquivo original. Portanto, os métodos sem perdas geralmente são usados para empacotar o *software* antes de ser enviado pela *internet* ou baixado de um site para reduzir o tempo e a largura de banda necessários para transmitir os dados (Gilchrist, 2004).

Para Betetto (2005) e Khandwani and Ajmire (2018), esse tipo de compressão de dados explora a redundância estatística para representar as informações de forma mais eficiente. Segundo os autores, os métodos de compressão sem perdas podem ser agrupados em diferentes categorias, tais como codificação baseada em entropia, codificação estatística e codificação baseada em dicionário.

A codificação baseada em entropia é independente das características específicas do meio (Khandwani and Ajmire, 2018), pois esse método inicialmente determina a frequência de ocorrência de cada símbolo presente nos dados (Blelloch, 2001). Dessa forma, sua aplicação independe do tipo de informação que está sendo comprimida. Para símbolos declarados do arquivo original, esses novos símbolos são fixos e não dependem do conteúdo do arquivo. Assim, o comprimento dos novos símbolos é variável e depende da frequência de ocorrência dos símbolos do arquivo original (Patel et al., 2015). Portanto, os codificadores de entropia exploram as propriedades estatísticas estimadas de sua mensagem para chegar bem perto do limite teórico de compactação (Brian, 2020).

Os algoritmos estatísticos segundo Gilchrist (2004), utilizam modelos construídos a partir da entrada conhecida e são usados para facilitar a compressão eficiente no codificador. É sabido que os métodos estatísticos utilizam códigos de comprimentos variáveis, ou seja, os dados na informação original que aparecem com maior frequência são representados por palavras-código menores, enquanto os dados de menor incidência são representados por palavras-código maiores.

Já a codificação Aritmética Gilchrist (2004), Sayood (2018) e Khandwani and Ajmire (2018) é uma técnica de compressão Estatística que usa estimativas das probabilidades de eventos para atribuir palavras de código. Idealmente, palavras de código curtas são atribuídas a eventos mais prováveis e palavras de código mais longas são atribuídas a eventos menos prováveis. Portanto, o codificador Aritmético deve trabalhar junto com um modelador que estima as probabilidades dos eventos na codificação.

E, na codificação baseada em dicionário Gilchrist (2004), Patel et al. (2015), Khandwani and Ajmire (2018) e Lukin et al. (2021), os algoritmos não codificam símbolos únicos como *strings* de *bits* de comprimento variável e sim *strings* de símbolos de comprimento variável como *tokens* únicos. Sob essa ótica, o codificador busca encontrar o par correspondente de *string* no dicionário do texto original e se esta correspondência for encontrada, ele substitui o símbolo pelo código correspondente no dicionário.

Neste contexto, o Lempel–Ziv–Welch (LZW) é um método de compressão de dados sem perda baseado em dicionário criado por Abraham Lempel, Jacob Ziv e Terry Welch (Khandwani and Ajmire, 2018). Considerado um algoritmo simples de implementar, com potencial para um rendimento muito alto em implementações de *hardware*. Neste sentido, por ser uma técnica adaptativa, não há necessidade de transmitir o dicionário ao receptor, uma vez que ele é reconstruído durante o processo de decodificação.

2.3 Algoritmos de Codificação sem Perdas

Nesta seção serão apresentados os métodos mais conhecidos e utilizados na compressão sem perdas de dados. Tais algoritmos são utilizados na metodologia proposta.

Também será apresentado o funcionamento, características e aplicações de cada método.

2.3.1 Código de Huffman

O algoritmo de Huffman foi desenvolvido por David Huffman em 1950 (Patel et al., 2015; Yuan and Hu, 2019). Neste método, os caracteres de um arquivo de dados são convertidos em códigos binários. Segundo Karam (2009) e Khan et al. (2020), a técnica de Huffman atribui códigos de prefixo aos dados a serem codificados. Sua implementação é relativamente simples, seu esquema de codificação é baseado em uma árvore de busca. Ademais, esse algoritmo é um método de codificação baseado em entropia que segue duas premissas:

- Símbolos que ocorrem com mais frequência devem ter códigos mais curtos e símbolos menos frequentes, códigos maiores; e
- Os dois símbolos de menor frequência devem ter códigos de mesmo comprimento.

Para um alfabeto $\mathcal{A}$ com N símbolos $\{s_1\, s_2\, s_3\, \ldots\, s_N\}$, associamos a cada símbolo s_i um peso p_i que representa sua probabilidade de ocorrência de modo que $\sum_i p_i = 1$. O algoritmo de Huffman objetiva construir uma árvore binária com os símbolos s_i como suas folhas que serão interligadas para formar ramos até chegar a uma raiz comum Sayood (2018).

A figura 6 ilustra um exemplo adaptado de McAnlis and Haecky (2016), dado um alfabeto $\mathcal{A}$ formado pelos símbolos $\{s_1,\ s_2,\ s_3,\ s_4,\ s_5\}$, os quais estão associadas às seguintes probabilidades de ocorrência, respectivamente, $\{p_1,\ p_2,\ p_3,\ p_4,\ p_5\}$. Considere ainda que $p_4 = p_5$ e $p_4 < p_3 < p_1 < p_2$.

Na primeira etapa da construção da árvore de Huffman, os dois símbolos de menor probabilidade, s_4 e s_5, foram combinados em um supersímbolo s_{45}, com peso $p_{45} = p_4 + p_5$. Assim, obtém-se o conjunto de símbolos e supersímbolos $s_1, , s_2, , s_3, , s_{45}$. Supondo ainda que $p_{45} < p_3$, os símbolos s_3 e s_{45} são unidos em um novo supersímbolo s_{345}, cujo peso é dado por $p_{345} = p_3 + p_{45}$. O novo conjunto de símbolos passa a ser $s_1, , s_2, , s_{345}$. Considerando que $p_{345} < p_1$, obtém-se um novo supersímbolo s_{1345}, cuja probabilidade é $p_{1345} = p_1 + p_{345}$. Por fim, une-se o último símbolo ao novo supersímbolo para determinar o nó raiz da árvore.

Dentro dessa perspectiva, um aspecto importante do processo de construção da árvore é a necessidade de ordenar os símbolos, em cada etapa, de acordo com seus respectivos pesos. Na etapa inicial, são considerados apenas os dois menores pesos. Contudo, à medida que novos supersímbolos são formados, seus pesos são calculados a partir das probabilidades dos símbolos que os compõem. Dessa forma, passam a ser considerados tanto os pesos dos símbolos remanescentes quanto os dos supersímbolos, até que a árvore seja completamente construída.

Ainda na Figura 6, é apresentada a atribuição de códigos a cada símbolo. Convenciona-se o dígito 0 para representar o deslocamento à esquerda na árvore e o dígito 1 para representar o deslocamento à direita. Assim, o código de cada símbolo é determinado pela trajetória percorrida desde a raiz até o respectivo símbolo.

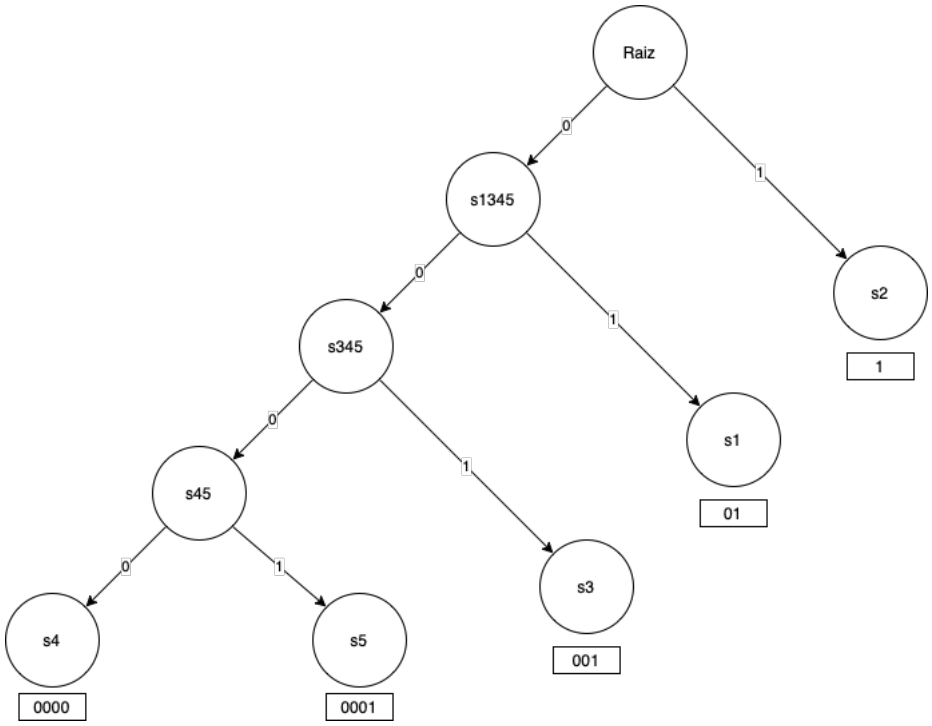

Figura 6 – Codificação de cada símbolo pela árvore de Huffman do exemplo dado.
Fonte: adaptado de (McAnlis and Haecky, 2016).

Já na decodificação, o dicionário símbolo x código gerado é utilizado para retornar à sequência original, pois, em seu código prefixo (Adel et al., 2018), cada símbolo possui um código único e não haverá redundâncias na decodificação.

Patel et al. (2015) apresenta dois tipos de categorias de Codificação de Huffman: Algoritmo de Huffman Estático e Algoritmo de Huffman Adaptativo. Os algoritmos de Huffman estáticos primeiro contam as frequências e, em seguida, geram uma árvore comum para os processos de compressão e descompressão. Já os algoritmos adaptativos de Huffman, por sua vez, constroem a árvore ao mesmo tempo em que contabilizam as frequências. Essa abordagem resulta na formação de duas árvores ao longo de ambos os processos.

2.3.2 Codificação Aritmética

A codificação aritmética foi proposta por (Langdon, 1984), técnica que gera códigos de comprimento variável (Karam, 2009; Mishra and Singh, 2016; Sneyers and Wuille, 2016). Em algumas situações, é superior a Huffman, especialmente em alfabetos reduzidos com distribuições de probabilidade altamente assimétricas (Jayasankar et al., 2021). Codificação e decodificação são baseadas na aplicação de operações aritméticas (Khan et al., 2020).

Como resultado, os dados serão representados como um único número. Tanto o código Huffman quanto o Aritmético precisam ler todos os dados para calcular as probabilidades antes da codificação. Porém, quando o conjunto de dados é pequeno, isso não é um problema.

Assim, quando o fluxo de dados aumenta, dois pontos precisam de atenção:

- A necessidade de mais tempo para ler e calcular as probabilidades; e
- A suposição de que a probabilidade calculada permanecerá constante ao longo da codificação.

Para uma melhor compreensão do funcionamento do algoritmo, um exemplo adaptado de Sayood (2018) é ilustrado na figura 7. Considere um alfabeto $A = \{a_1, a_2, a_3\}$ com $P(a_1) = 0.7$, $P(a_2) = 0.1$ e $P(a_3) = 0.2$. Usando o mapeamento da sequinte equação $X(a_i) = i, a_i \in \mathcal{A}$, temos $F_X(1) = 0.7$, $F_X(2) = 0.8$ e $F_X(3) = 1$.

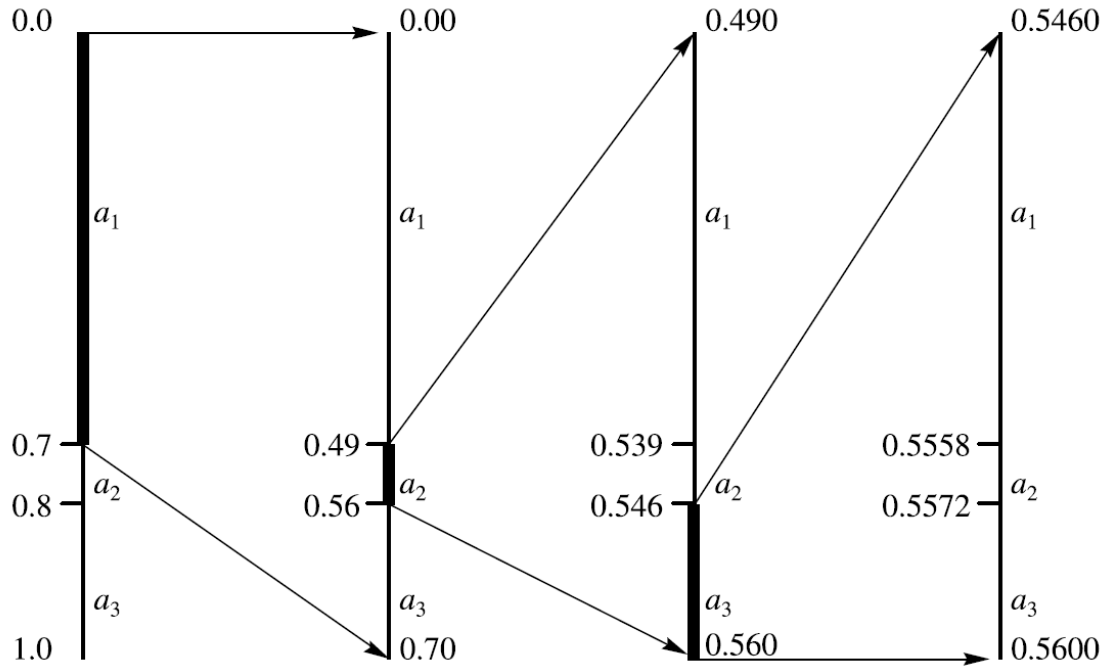

Figura 7 – Restringindo o intervalo contendo a *tag* para a sequência de entrada $\{a_1, a_2, a_3, ...\}$. Fonte: (Sayood, 2018).

A figura 7 ilustra que a partição na qual a *tag* reside depende do primeiro símbolo da sequência que está sendo codificada. Desta maneira, se o primeiro símbolo for a_1, a *tag* fica no intervalo $[0.0, 0.7)$; se o primeiro símbolo for a_2, a *tag* fica no intervalo $[0.7, 0.8)$; e se o primeiro símbolo for a_3, a *tag* fica no intervalo $[0.8, 1.0)$.

Uma vez identificado o segmento que contém a *tag*, toda a região restante do intervalo unitário é descartada, e o segmento selecionado passa a representar o novo espaço

de busca. Em seguida, ele é novamente particionado de acordo com as mesmas proporções utilizadas inicialmente. Assim, se o primeiro símbolo for a_1, a *tag* estará contida em $[0.0, 0.7)$. Esse segmento torna-se o novo intervalo de referência e é dividido nas mesmas proporções do intervalo original, produzindo as faixas $[0.0, 0.49)$, $[0.49, 0.56)$ e $[0.56, 0.7)$.

A primeira faixa corresponde ao símbolo a_1, a segunda ao símbolo a_2 e a terceira, $[0.56, 0.7)$, ao símbolo a_3. Supondo que o segundo símbolo da sequência seja a_2, a *tag* ficará restrita ao intervalo $[0.49, 0.56)$. Esse novo intervalo é então dividido segundo as mesmas proporções do intervalo original, resultando nas faixas $[0.49, 0.539)$, associada ao símbolo a_1, $[0.539, 0.546)$, associada ao símbolo a_2, e $[0.546, 0.56)$, associada ao símbolo a_3. Caso o terceiro símbolo seja a_3, a *tag* passará a estar contida em $[0.546, 0.56)$. Esse processo pode ser repetido sucessivamente, refinando a localização da *tag* a cada novo símbolo decodificado.

Observe que a aparência de cada novo símbolo restringe a *tag* a um subintervalo que é separado de qualquer outro subintervalo que possa ter sido gerado por meio desse processo. Para a sequência que começa com $a_1, a_2, a_3, ...$, no momento em que o terceiro símbolo a_3 é recebido, a *tag* foi restrita ao subintervalo $[0.546, 0.56)$. Se o terceiro símbolo fosse a_1 em vez de a_3, a *tag* residiria no subintervalo $[0.49, 0.539)$, que é separado do subintervalo $[0.546, 0.56)$. Mesmo que as duas sequências sejam idênticas a partir deste ponto (uma começando com a_1, a_2, a_3 e a outra começando com a_1, a_2, a_1), o intervalo de *tag* para as duas sequências sempre será disjunto.

Como podemos ver, o intervalo no qual reside a *tag* de uma determinada sequência é separado de todos os intervalos nos quais pode residir a *tag* de qualquer outra sequência. Assim, qualquer membro desse intervalo pode ser usado como *tag*. Uma escolha popular é o limite inferior do intervalo; outra possibilidade é o ponto médio do intervalo. Por enquanto, vamos usar o ponto médio do intervalo como *tag*. Para ver como o procedimento de geração de *tags* funciona matematicamente, começamos com sequências de comprimento um e supondo que temos uma fonte que emite símbolos de algum alfabeto $\mathcal{A} = \{a_1, a_2, ..., a_m\}$.

Podemos mapear os símbolos $\{a_i\}$ para números reais $\{i\}$. Defina $\bar{T}_X(a_i)$ como:

$$\bar{T}_X(a_i) = \sum_{k=1}^{i-1} P(X = k) + \frac{1}{2}P(X = i) \tag{2.2}$$

$$= F_X(i-1) + \frac{1}{2}P(X = i) \tag{2.3}$$

Para cada a_i, $\bar{T}_X(a_i)$ terá um valor único. Esse valor pode ser usado como uma tag exclusiva para a_i.

É importante destacar que, para aumentar a eficiência dos codificadores estatísticos, são empregados mecanismos adaptativos capazes de ajustar as probabilidades dos símbolos

à medida que estes são observados no fluxo de dados, especialmente quando a distribuição da fonte é desconhecida. Na codificação Aritmética adaptativa (Barannik et al., 2018), a tabela de probabilidades é atualizada após a ocorrência de cada símbolo. De forma análoga, durante a decodificação, o mesmo processo de atualização é realizado para garantir a sincronização entre codificador e decodificador. Consequentemente, as probabilidades são continuamente atualizadas ao longo do processamento, permitindo que o modelo represente com maior precisão as características estatísticas da sequência de dados e, assim, alcance melhor desempenho de compressão (Sayood, 2018).

2.4 Técnica de Transformação de Dados para Compressão sem Perdas

A transformação de dados consiste em uma etapa de pré-processamento amplamente empregada em algoritmos de compressão sem perdas, cujo objetivo é modificar a representação dos dados de entrada de forma a torná-los mais adequados às etapas subsequentes de codificação.

Diferentemente dos algoritmos de codificação, que exploram diretamente as redundâncias estatísticas presentes na fonte de informação, as transformações alteram a representação dos dados sem modificar seu conteúdo informacional, preservando integralmente a possibilidade de reconstrução da sequência original. Entre as técnicas clássicas destacam-se a Transformação de Burrows-Wheeler (*Burrows-Wheeler Transform* – BWT) (Adjeroh et al., 2008), a Move-to-Front (MTF) (Arnavut, 2000), Codificação Preditiva (Millidge et al., 2021), e a Run-Length Encoding (RLE) (Peng et al., 2023), amplamente utilizadas para reorganizar os dados de modo a evidenciar padrões estatísticos que possam ser explorados com maior eficiência pelos algoritmos de compressão.

Este trabalho utiliza uma técnica de pré-processamento denominada *Transformação Baseada na Distribuição Geométrica* (Artikis et al., 1998), aplicada como etapa intermediária entre a representação binária dos dados e a codificação aritmética adaptativa (Blelloch, 2001; Sayood, 2018; Barannik et al., 2018). Essa transformação converte uma sequência binária, originalmente modelada por um processo de Bernoulli (Chan, 2026), em uma sequência composta pelas distâncias entre ocorrências consecutivas do símbolo “1”. Ou seja, em vez de codificar diretamente cada bit da sequência original, a transformação registra o número de zeros consecutivos que antecedem cada ocorrência do símbolo “1”.

Sob a perspectiva probabilística, essa abordagem promove uma mudança no modelo estatístico da sequência de entrada. Enquanto a sequência binária original é descrita por uma distribuição de Bernoulli, caracterizada pela probabilidade p de ocorrência do símbolo “1”, a sequência transformada passa a ser modelada por uma distribuição geométrica (Durrett, 2019), cuja função de probabilidade é dada por

$$P(K = k) = (1 - p)^k p, \qquad k = 0, 1, 2, \ldots \tag{2.4}$$

em que k representa o número de símbolos "0" consecutivos observados antes da ocorrência do próximo símbolo "1".

Essa transformação mostra-se particularmente adequada para dados de telemetria satelital, nos quais determinados eventos ocorrem com baixa frequência, resultando em longas sequências de zeros intercaladas por poucas ocorrências do símbolo "1". Nessas condições, a distribuição geométrica concentra maior probabilidade em um conjunto reduzido de valores de k, produzindo uma sequência estatisticamente mais previsível e favorecendo a estimação adaptativa das probabilidades realizada pela codificação aritmética.

Como a transformação estabelece uma correspondência biunívoca entre a sequência binária original e a sequência de distâncias obtida, ela preserva integralmente a informação da fonte, caracterizando-se como uma transformação reversível. Durante a descompressão, a sequência original é reconstruída pela expansão das distâncias codificadas, reconstituindo cada sequência de zeros e inserindo os símbolos "1" em suas respectivas posições. Dessa forma, garante-se a recuperação exata dos dados originais, preservando a principal característica da compressão sem perdas.

Por fim, a transformação baseada na distribuição geométrica não realiza compressão diretamente, mas atua como um mecanismo de pré-processamento estatístico que modifica a representação da informação antes da etapa de codificação. Ao transformar uma sequência modelada por uma distribuição de Bernoulli em outra descrita por uma distribuição geométrica, a técnica produz um fluxo de dados mais compatível com a codificação aritmética adaptativa, permitindo uma exploração mais eficiente das redundâncias estatísticas presentes em sequências binárias não uniformes.

3 Metodologia Proposta

Neste capítulo, diante do problema abordado, são propostas as etapas para o desenvolvimento de um algoritmo de compressão sem perdas voltado para dados de satélite, com foco específico em dados de *housekeeping*. O objetivo do algoritmo é reduzir o volume de dados hexadecimais sem comprometer sua integridade. Esta proposta está estruturada em três etapas. A primeira etapa consiste na análise e no desenvolvimento de um algoritmo On+ para compressão sem perdas em dados de telemetria, incluindo: modelo matemático baseado na transformação geométrica, modelo de estimação de entropia, algoritmos de codificação e decodificação, e análise de desempenho e eficiência do algoritmo proposto. A segunda etapa consiste em investigar métricas de avaliação da qualidade da compressão. Por fim, na terceira etapa, são realizadas a validação experimental, bem como a apresentação do repositório de dados de telemetria, da plataforma experimental e dos procedimentos de *benchmark*.

3.1 Algoritmo proposto On+

A Figura 8 apresenta o fluxo de codificação e decodificação da técnica proposta neste trabalho, denominada On+.

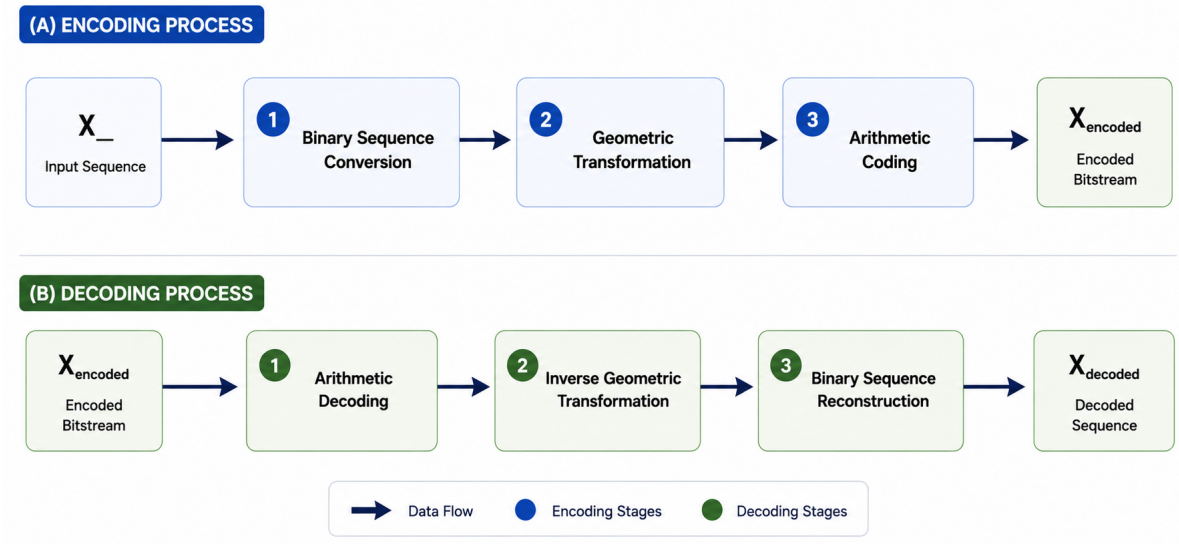

Figura 8 – Pipeline de codificação e decodificação proposto. (A) Processo de codificação. (B) Processo de decodificação.

O processo de codificação (A) do algoritmo On+ começa com a sequência de entrada X, que é submetida sequencialmente a três etapas principais: conversão para uma

sequência binária, transformação geométrica e codificação aritmética, resultando no fluxo de bits compactado X_{encoded} com tamanho reduzido.

O processo de decodificação (B), por sua vez, realiza as operações inversas. Partindo do fluxo de bits codificado X_{encoded}, executa-se a decodificação aritmética, seguida pela transformação geométrica inversa e, finalmente, pela reconstrução da sequência binária original, recuperando a sequência decodificada X_{decoded}.

3.1.1 Modelagem Matemática

Apresentamos uma notação formal para a descriçao do algoritmo On+. Sejam os vetores:

- $X = \{x_1, x_2, x_3, \ldots, x_n\}$, vetor binário de tamanho n, onde $x_i \in \{0, 1\}$, e $x_n = 1$.
- $Y = \{y_1, y_2, y_3, \ldots, y_m\}$, vetor de conversão de tamanho m, onde $y_j \in \mathbb{N}_{>0}$.

Seja i índice dos elementos do vetor X, com $i = 1, \ldots, n$ e j índice dos elementos do vetor Y, com $j = 1, \ldots, m$. Definimos k, inicialmente igual a 1, como a quantidade de *zeros* consecutivos entre $x_i = 0$ e $x_{i+k} = 1$.

Defina-se a seguinte regra de codificação da conversão de X para Y:

$$y_j = \begin{cases} 1, & se \;\; x_i = 1 \\ \\ k + 1, & se \;\; x_i = 0 \;\; e \;\; x_{i+k} = 1 \end{cases} \tag{3.1}$$

Para ilustrar a transformação proposta nesta seção, a Figura 9 apresenta um exemplo de aplicação da transformação geométrica a uma sequência hipotética. Consideramos o vetor binário $X = [0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1]$, com o objetivo de transformá-lo no vetor Y. Para isso, percorremos sequencialmente cada elemento do vetor X.

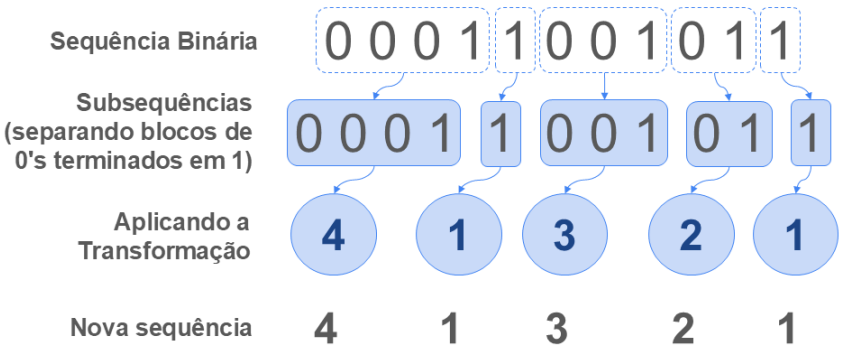

Figura 9 – Exemplo de aplicação da transformação geométrica em uma sequência.

O resultado da transformação do vetor binário $X = [0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1]$ é o vetor $Y = [4, 1, 3, 2, 1]$. Cada símbolo $Y \in \Sigma$ possui uma probabilidade associada $P(Y)$, estimada com base na frequência de ocorrência dos símbolos no vetor Y. Este é diretamente submetido ao método de codificação Aritmético Adaptativo. Portanto, após o processamento de todos os símbolos, teremos um único número n, que representa a sequência completa.

Para o processo de decodificação da regra de conversão de Y para X, temos por definição, a regra de concatenação de $A \bigoplus B = [a_1, a_2, \ldots, a_n] \bigoplus [b_1, b_2, \ldots, b_m] = [a_1, a_2, \ldots, a_n, b_1, b_2, \ldots, b_m]$, onde $\bigoplus$ é o operador de concatenação, e $\bigotimes$ é operação de repetição.

Considerando a regra de concatenação conforme descrita na regra de conversão para decodificação:

$$X = \begin{cases} X \bigoplus [1], \;\; se \;\; y_j = 1 \\ \\ X \bigoplus (([0] \bigotimes (y_j - 1)) \bigoplus [1]) \;, \;\; se \;\; y_j > 1 \end{cases} \tag{3.2}$$

onde, para cada elemento y_j no vetor Y, um bit 1 é incluído no vetor X quando $y_j = 1$. Caso contrário, $(y_j - 1)$ bits 0 são concatenados, seguidos por um bit 1.

Para ilustrar a transformação proposta foi gerada uma sequência binária X de 5 MB (*Mega Bytes*) com probabilidade $p = 0,5$. A Figura 10 apresenta o histograma da sequência em questão transformada de binário para hexadecimal, que se aproxima de uma distribuição uniforme. O valor hexadecimal observado mínimo foi de 0 e o máximo de 65534. O tamanho do alfabeto 65072 símbolos, o que indica que há valores ausentes no intervalo entre o mínimo e o máximo observados.

A Figura 11 apresenta o histograma dos valores da sequência X após a aplicação da transformação geométrica. Observa-se que a distribuição resultante aproxima-se de uma distribuição geométrica, com valor mínimo igual a 1 e valor máximo igual a 22. O tamanho do alfabeto é de 22 símbolos, indicando que todos os símbolos compreendidos entre os valores mínimo e máximo ocorrem pelo menos uma vez na sequência.

Observa-se que a transformação proposta reduz o tamanho efetivo do alfabeto e torna a distribuição dos símbolos mais assimétrica, concentrando uma maior probabilidade de ocorrência em um subconjunto reduzido de valores. Essas características são particularmente favoráveis para codificadores de entropia, uma vez que aumentam a previsibilidade estatística da fonte e permitem uma representação mais eficiente dos dados. Consequentemente, algoritmos como a codificação aritmética podem explorar melhor essa distribuição, potencializando os ganhos de compressão.

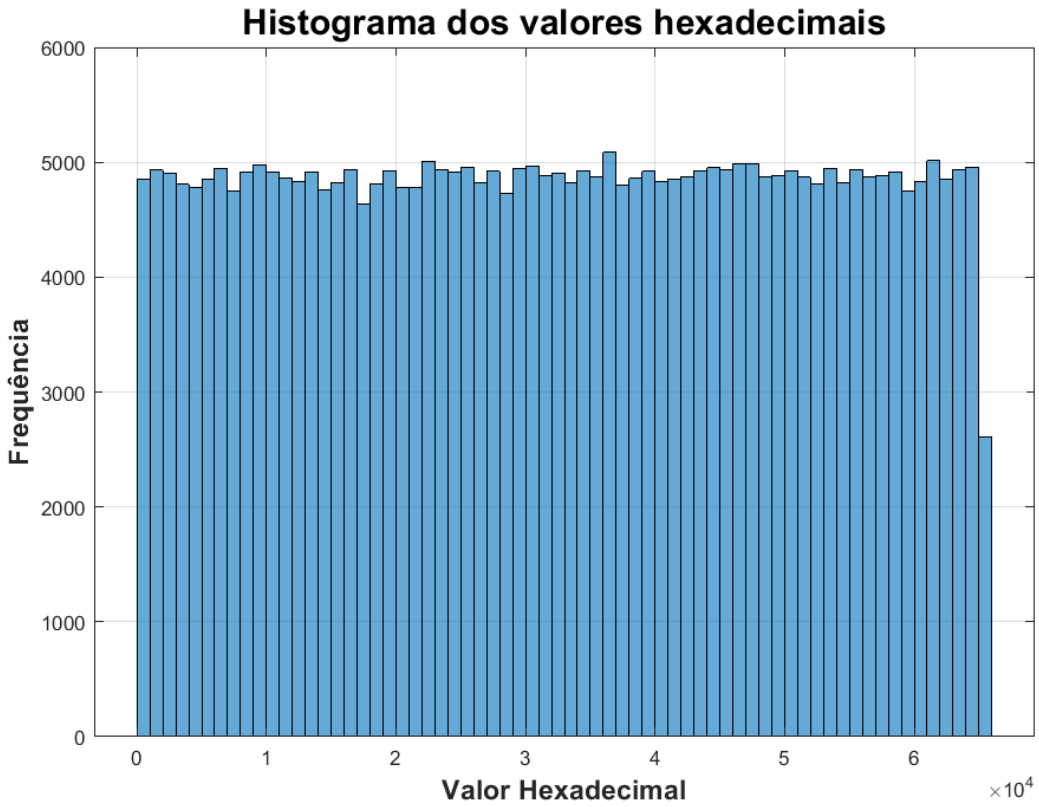

Figura 10 – Histograma dos valores hexadecimais.

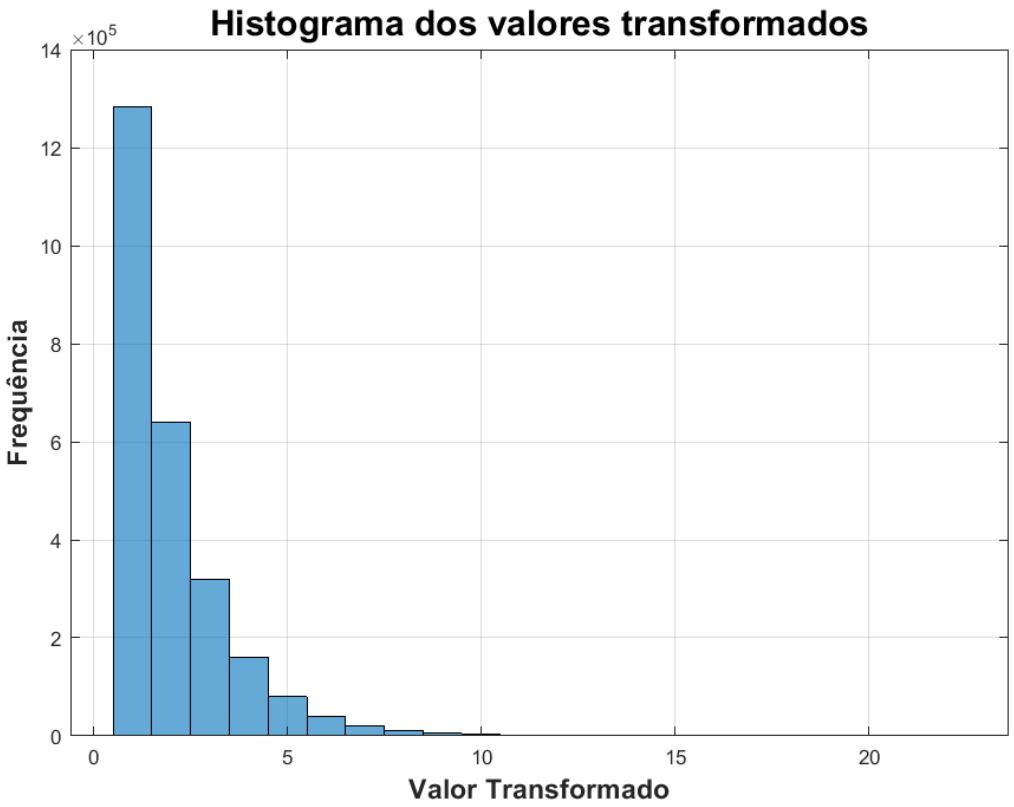

Figura 11 – Histograma dos valores transformados.

3.1.2 Modelo de Estimação de Entropia

Apresentamos um modelo formal para a descrição do algoritmo On+. Segue as etapas:

i) Dada uma sequência de Bernoulli $X = \{x_1, x_2, x_3, \ldots, x_{N_X}\}$, vetor binário de tamanho N_X, e $x_{N_X} = 1$, com cada $x_i \in \{0, 1\}$, e com $p(x_i = 1) = p$, a entropia H_X desta sequência é dada por:

$$H_X = -\left[(1-p)\log_2(1-p) + p\log_2 p\right] \tag{3.3}$$

O comprimento médio L_X da sequência X, em bits, é dado por:

$$L_X = N_X \cdot H_X \tag{3.4}$$

ii) Transformação da sequência X em Y: A sequência X é transformada em uma sequência $Y = \{y_1, y_2, y_3, \ldots, y_{N_Y}\}$, vetor de tamanho N_Y, onde $N_Y < N_X$ e $y_i \sim \text{Geom}(p)$ segue uma distribuição geométrica com mesmo parâmetro p da sequência X. A entropia H_Y é dada pela equação:

$$H_Y = -\left[\frac{(1-p)\ log_2\ (1-p) + p\ log_2\ p}{p}\right] = \frac{H_X}{p} \tag{3.5}$$

O comprimento médio L_Y da sequência Y, em bits, é dado por:

$$L_Y = N_Y \cdot H_Y = N_Y \cdot \frac{H_X}{p} \tag{3.6}$$

A partir da análise comparativa entre as sequências X e Y, conclui-se, primeiramente, que a entropia de Y é sempre maior ou igual à de X ($H_Y \geq H_X$), dado que $p \in [0, 1]$. Em segundo lugar, verifica-se que a distribuição de X difere da distribuição de Y, que assume a forma de uma distribuição geométrica (Geo(p)). Esta alteração na distribuição original de X ocorre em função do fato de que os símbolos de Y são formados a partir de partições de comprimento variável da sequência binária de X.

A relação entre o tamanho, em bits, da sequência transformada Y e a sequência original X é dado por:

$$R = \frac{L_Y}{L_X} = \frac{N_Y H_Y}{N_X H_X} = \frac{\frac{N_Y H_X}{p}}{N_X H_X} = \frac{N_Y}{N_X} \cdot \frac{1}{p} \tag{3.7}$$

Pelo fato que cada símbolo da sequência Y é formado por uma partição de X que possui apenas um único bit 1, a razão $\frac{N_Y}{N_X} = p$ e a relação R é reduzida a

$$R = p \cdot \frac{1}{p} = 1 \tag{3.8}$$

Constata-se que o processo não altera o tamanho do arquivo em bits. Verifica-se que não há compressão efetiva durante a transformação, pois a entropia de Y seja superior à de X ($H_Y > H_X$), porém o número de símbolos em Y é menor ($N_Y < N_X$), mantendo a razão $R = 1$. No entanto, a distribuição assimétrica assumida por Y favorece o desempenho de codificadores de entropia, como o algoritmo Aritmético, que podem explorar essa característica para alcançar compressão mais eficiente.

3.1.3 Algoritmo de Codificação

O algoritmo 1 apresenta o processo de codificação em detalhes. Nesta etapa, os dados originais são analisados e representados de forma mais compacta. O algoritmo On+ desempenha um papel crucial ao substituir padrões repetitivos por símbolos mais curtos, reduzindo assim a redundância nos dados e eliminando informações desnecessárias ou representando-as de maneira mais eficiente.

Algoritmo 1: Codificação

```
Input: X (vetor binário)
Output: encoded (vetor vector)
begin
    X ← [X 1]
    X_geometric ← [ ] (vetor vazio)
    count0 ← 0
    for i ← 1 to tamanho(X) do
        if X(i) = 0 then
            count0 ← count0 + 1
        else
            if count0 > 0 then
                concatenar (count0 + 1) ao final do vetor X_geometric
            else
                concatenar 1 ao final do vetor X_geometric
            count0 ← 0
    encoded ⟵ AdaptativeAritmethicEncoding (X_geometric)
```

Para garantir que o critério de parada do processo de codificação seja sempre satisfeito, o algoritmo adiciona inicialmente um *bit* igual a 1 ao final da sequência binária de entrada, conforme mostrado na linha de inicialização do vetor $X \leftarrow [X \;\; 1]$. Esse bit adicional atua como um marcador de fim de sequência, garantindo que a última sequência de zeros seja corretamente processada e codificada. Durante a etapa de decodificação, esse bit é removido ao final do processo para restaurar exatamente a sequência binária original.

3.1.4 Algoritmo de Decodificação

O processo de decodificação sem perdas do arquivo gerado pelo método de codificação é apresentado em detalhe no Algoritmo 2. A decodificação consiste em reconstruir os dados originais a partir da sequência comprimida, revertendo as operações realizadas durante a etapa de codificação. Nesta fase, o algoritmo On+ restaura o vetor binário original a partir do arquivo codificado, aplicando as operações inversas às executadas no Algoritmo 1.

Algoritmo 2: Decodificação

Input: *encoded* (*vetor decimal*)
Output: *decoded* (*vetor binário*)
begin
  $X_{geometric} \longleftarrow$ AdaptativeAritmethicDecoding (*encoded*)
  for $i \leftarrow 1$ ***to*** *tamanho*$(X_{geometric})$ **do**
    concatenar $(X_{geometric}(i) - 1)$ zeros ao final de *decoded*
    concatenar 1 ao final de *decoded*
  Remover o último bit de *decoded*

Inicialmente, o algoritmo realiza a decodificação aritmética adaptativa para recuperar o vetor $X_{geometric}$. Em seguida, cada elemento deste vetor é utilizado para reconstruir a sequência binária original, adicionando $(X_{geometric}(i) - 1)$ zeros seguidos por um *bit* igual a 1. Ao final do processo, o último bit da sequência reconstruída é removido. Esse *bit* corresponde exatamente ao *bit* adicional inserido durante a etapa de codificação no Algoritmo 1, utilizado para garantir o critério de parada do processo de codificação. Portanto, sua remoção não causa qualquer perda de informação, mas simplesmente restaura corretamente a sequência binária original.

3.1.5 Análise de Desempenho e Eficiência do Algoritmo On+

A avaliação do desempenho do algoritmo On+ foi realizada por meio de uma abordagem estruturada que compreendeu a análise de complexidade computacional, a definição de métricas específicas e a execução de testes comparativos (*benchmarks*). Inicialmente, foi conduzido um estudo de complexidade com o objetivo de determinar o consumo de recursos computacionais em função do tamanho da entrada, seguindo a abordagem proposta por Phalke et al. (2022). Foram examinadas duas dimensões principais: a complexidade temporal, expressa pela notação assintótica $O(n)$, que descreve a variação do tempo de execução à medida que o volume de dados aumenta, e a complexidade espacial, que quantifica a quantidade adicional de memória requerida durante o processamento (Vaz et al., 2017).

Essa etapa permitiu caracterizar o comportamento do On+ sob diferentes condições de execução, incluindo os cenários de melhor caso, caso médio e pior caso, fornecendo

subsídios para a comparação com métodos alternativos, a identificação de oportunidades de otimização e a avaliação de sua escalabilidade.

A análise de complexidade é fundamental para compreender a eficiência de um algoritmo e prever seu desempenho em diferentes condições de operação (Vaz et al., 2017). De acordo com Phalke et al. (2022), a complexidade temporal mede o crescimento do tempo de execução em função do tamanho da entrada, sendo normalmente representada por notações assintóticas, como $O(n)$. A notação O (ou *Grande-O*) expressa o limite superior do custo computacional e é amplamente utilizada para descrever o comportamento do algoritmo no pior caso. Em contrapartida, a complexidade espacial quantifica a memória adicional requerida durante a execução, permitindo estimar como a demanda por armazenamento evolui à medida que o volume de dados aumenta.

Para quantificar o desempenho do On+, adotou-se a taxa de compressão (CR – *Compression Ratio*) como um dos principais indicadores de qualidade da compressão, calculada conforme a Equação 3.2.2. Essa métrica expressa a variação percentual do tamanho do arquivo após o processo de compressão: valores positivos indicam redução no tamanho do arquivo, enquanto valores negativos representam expansão em relação ao arquivo original. A escolha da taxa de compressão deve-se à sua ampla utilização na avaliação de algoritmos de compressão e à sua relevância para dados de telemetria, nos quais a economia de armazenamento e de largura de banda constitui um requisito essencial.

Por fim, para validar a eficiência do On+ em comparação com métodos consolidados na literatura, foram realizados *benchmarks* utilizando técnicas amplamente reconhecidas, como a codificação de Huffman e a codificação Aritmética. Os experimentos avaliaram tanto a taxa de compressão quanto o tempo de execução, permitindo uma análise comparativa abrangente do desempenho do algoritmo proposto.

3.2 Indicadores de Qualidade de Compressão

Algoritmos de compressão podem ser avaliados por diversos critérios, como velocidade, complexidade computacional, uso de memória e taxa de compressão (Cayoglu et al., 2019; Jayasankar et al., 2021). As métricas utilizadas nesta proposta para avaliar os algoritmos desenvolvidos são apresentadas a seguir.

3.2.1 Fator de Compressão (FC)

$$FC = \frac{\text{Tamanho do arquivo original em bits}}{\text{Tamanho do arquivo codificado em bits}} \tag{3.9}$$

A Equação 3.9 fornece a razão entre o arquivo original e o arquivo comprimido. Quanto maior o Fator de Compressão (FC), melhor o algoritmo de codificação. O resultado

dessa medida será sempre maior que 1 para arquivos que foram comprimidos. Caso contrário, quando o arquivo não sofre compressão e sim expansão, essa medida estará no intervalo $[0, 1]$.

3.2.2 Taxa de Compressão (CR)

$$CR = 1 - FC \tag{3.10}$$

A Equação 3.10 mede a taxa de compressão em relação ao tamanho do arquivo original. Em outras palavras, CR representa a redução relativa obtida após o processo de compressão. Assim, valores positivos de CR indicam que a compressão foi bem-sucedida, enquanto valores negativos indicam que ocorreu expansão dos dados.

3.3 Validação Experimental

A terceira etapa da metodologia consistiu na realização dos experimentos destinados à validação da proposta desenvolvida. Nessa fase, são descritos os procedimentos adotados para avaliar o desempenho do algoritmo proposto, bem como a plataforma experimental utilizada e o conjunto de dados de telemetria empregado nos testes. Adicionalmente, são apresentados os critérios de avaliação e os métodos de comparação utilizados para verificar a eficácia da solução proposta.

3.3.1 Análise Experimental com Dados de Telemetria de Satélites

Utilizando o repositório de dados descrito na Subseção 3.3.2, foi realizada uma análise de desempenho do algoritmo On+ por meio de um teste de *benchmark*. O objetivo foi avaliar o desempenho do algoritmo On+ e compará-lo com métodos tradicionais de compressão (Huffman e Aritmético), bem como com compressores comerciais (.rar, .zip, .7z, .xz e .gz), analisando a eficiência de compressão. As métricas de desempenho consideradas na avaliação são baseadas nos indicadores apresentados na Seção 3.2, e incluem:

- **Taxa de compressão mediana (%):** representa a taxa de compressão mediana alcançada pelos algoritmos em termos percentuais. A mediana é o valor central do conjunto de dados, sendo menos sensível a valores extremos (*outliers*) do que a média. A taxa de compressão indica quanto do arquivo original foi reduzido após a compressão. Valores mais altos indicam melhor desempenho.
- **Taxa de compressão média (%):** representa a taxa de compressão média alcançada pelos algoritmos em termos percentuais. Valores mais altos indicam melhor desempenho na compactação dos dados.

- **Desvio padrão da taxa de compressão (%):** quantifica a dispersão das taxas de compressão obtidas entre as diferentes amostras. Esse parâmetro é utilizado para avaliar a consistência dos algoritmos, de modo que valores menores indicam resultados mais uniformes e previsíveis.

- **Taxa de compressão máxima (%):** o maior valor de compressão observado em uma amostra foi registrado para cada algoritmo. Este parâmetro fornece uma ideia do melhor caso de compressão, destacando o potencial máximo de redução.

- **Taxa de compressão mínima (%):** o menor valor de compressão observado entre todas as amostras. Este parâmetro é útil para identificar cenários em que os algoritmos apresentam desempenho inferior.

- **Quantidade de arquivos comprimidos:** indica a quantidade de arquivos que apresentaram redução de tamanho após a aplicação do algoritmo. Este parâmetro é importante para avaliar a eficácia da técnica de compressão, permitindo quantificar sua capacidade de reduzir os requisitos de armazenamento e transmissão de dados.

- **Quantidade de arquivos expandidos:** refere-se à quantidade de arquivos cujo tamanho aumentou após a aplicação do algoritmo. Este parâmetro é relevante para identificar cenários em que a compressão não é vantajosa, permitindo avaliar a robustez e a aplicabilidade do algoritmo em diferentes conjuntos de dados.

3.3.2 Plataforma Experimental e Repositório de Dados de Telemetria

Em termos de plataforma experimental, os algoritmos foram implementados utilizando o *software* Matlab R2020a, com licença fornecida pela Universidade Federal do Maranhão (UFMA). Os experimentos foram conduzidos em um notebook equipado com processador Intel(R) Core(TM) i7-10510U CPU (1.80GHz - 2.30GHz) e 8 GB de memória RAM. Processadores de vídeo (GPU) não foram empregados.

A avaliação dos algoritmos de compressão sem perdas foi conduzida sobre arquivos binários de telemetria de satélites, utilizando-se dados reais provenientes do satélite Aldebaran-1 (Marques et al., 2022) e da plataforma de *Ground Station TinyGS* (TinyGS Community, 2026).

O satélite Aldebaran-1 é um CubeSat de formato 1U. Esse CubeSat foi desenvolvido pelo Laboratório de Eletrônica e Sistemas Embarcados Espaciais (LABESEE/UFMA) em parceria com o SpaceLab/UFSC, com o objetivo principal de auxiliar no resgate de pescadores em situações de emergência ao longo da costa brasileira, por meio do

recebimento de sinais de alerta transmitidos por embarcações que enfrentam condições oceânicas adversas (Marques et al., 2022).

A Figura 12 apresenta o modelo de engenharia do satélite Aldebaran-1 e o ambiente experimental utilizado para a aquisição dos dados de telemetria.

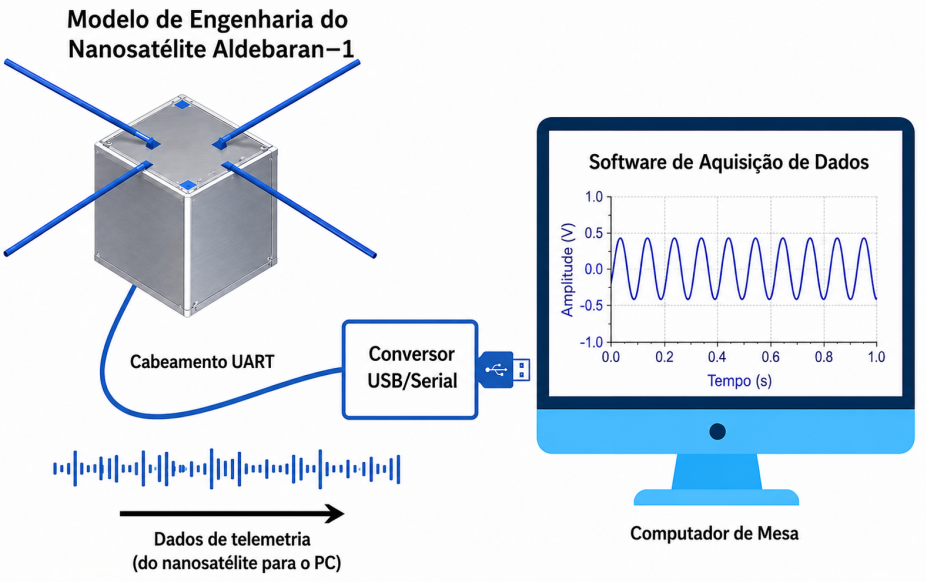

Figura 12 – A transmissão da telemetria é realizada por meio da interface UART.

O projeto Aldebaran-1 foi projetado para operar como receptor de dados de plataformas de coleta, garantindo a integração eficiente das informações adquiridas. O modelo de engenharia utilizado nos experimentos foi desenvolvido para coletar e transmitir dados de telemetria por meio de sua interface UART. Os dados transmitidos são capturados por um conversor Serial UART-para-USB e exibidos em um computador pessoal para análise. Os dados recebidos são então processados por um software de aquisição e subsequentemente utilizados em testes de compressão.

Para prevenir distorções na transmissão, a modulação LoRa inclui detecção/correção nativa de erros na camada física (Elzeiny et al., 2021). Ela utiliza Correção de Erros Avançada (FEC - *Forward Error Correction*) para aumentar a resiliência do sinal contra interferências, o que é crucial para comunicações de longo alcance com satélites em órbita baixa da Terra. O LoRa adiciona bits redundantes a cada pacote, permitindo a detecção e correção de erros de bit sem retransmissão, sendo esse recurso configurável para ambientes ruidosos. Para combater interferências de rajada, o LoRa emprega entrelaçamento (*interleaving*) para distribuir os erros entre as palavras de código, melhorando a eficácia da correção por código de Hamming. Adicionalmente, o LoRa utiliza um código de redundância cíclica para garantir a integridade dos dados na carga útil da mensagem.

Também foram utilizados dados reais de telemetria provenientes de diversos satélites disponíveis na plataforma *TinyGS* (TinyGS Community, 2026). Trata-se de uma rede

global de estações terrestres de código aberto, baseada na tecnologia LoRa, que possibilita a construção e operação de estações de rastreamento de satélites de baixo custo. Como iniciativa de ciência cidadã, a *TinyGS* conta com a participação de voluntários distribuídos mundialmente, responsáveis pela recepção e pelo compartilhamento de dados de missões espaciais.

Por meio dessa infraestrutura colaborativa, é possível obter, em tempo real, dados de telemetria e informações científicas provenientes de satélites, balões meteorológicos e outras plataformas aeroespaciais. Atualmente, a rede conta com mais de 2.250 estações ativas distribuídas globalmente, proporcionando ampla cobertura para a recepção de sinais e o monitoramento de objetos em órbita.

Além de disponibilizar dados para pesquisa e desenvolvimento, a plataforma contribui para o rastreamento de satélites e outros objetos espaciais, fornecendo informações sobre suas posições e trajetórias. Dessa forma, a *TinyGS* constitui uma importante fonte de dados reais para experimentos e avaliações de técnicas de processamento e compressão de telemetria, como as desenvolvidas neste trabalho.

O repositório de dados de telemetria contém 1900 arquivos binários (.bin) extraídos da rede global *TinyGS* e 600 arquivos de telemetria capturados do satélite Aldebaran-1, totalizando 2500 arquivos. Cada arquivo foi processado individualmente pelos algoritmos avaliados durante os experimentos de *benchmark*. A Tabela 1 apresenta a quantidade de arquivos, tamanho mínimo e máximo de cada satélite usado no repositório.

Tabela 1 – Repositório com quantidade de arquivos, tamanho mínimo e máximo em KBytes.

Descrição	**Quantidade**	**Tamanho Mínimo (kB)**	**Tamanho Máximo (kB)**
GaoFen	300	0,100	0,100
Polytech Universe 3	100	0,068	0,068
Swarm	100	0,015	0,248
SpaceBEE	500	0,015	0,248
Starlink	300	0,073	0,087
Norby	300	0,081	0,143
Tianqi	300	0,100	0,100
Aldebaran-1	600	0,164	0,165

4 Resultados

Neste capítulo, avaliamos o desempenho do algoritmo On+ e o comparamos com outros métodos de compressão sem perdas. Inicialmente, apresentamos um repositório com um conjunto de dados de telemetria de satélite utilizados nos experimentos e descrevemos a plataforma experimental, destacando suas principais características. Em seguida, realizamos o *benchmark* do algoritmo On+ utilizando um conjunto predefinido de parâmetros, comparando o método proposto com técnicas clássicas de compressão, como a codificação de Huffman e a codificação Aritmética, além de compressores comerciais amplamente utilizados, como .rar, .zip, .7z, .xz e .gz.

Desta forma, as métricas de avaliação usadas serão as mesmas apresentadas na Seção 3.2. Além disso, realizamos também uma análise da entropia dos arquivos do repositório, bem como da entropia associada à probabilidade de sucesso (p). Por fim, analisamos a complexidade temporal e espacial dos algoritmos de codificação e decodificação do On+, juntamente com os métodos de compressão Aritmética e de Huffman, incluindo uma tabela comparativa que resume os principais resultados obtidos.

4.1 Benchmarking do Algoritmo On+

Para avaliar o desempenho do algoritmo On+, foi realizado um benchmark utilizando dados de telemetria do satélite Aldebaran-1, comparando-o com o método de Huffman e a codificação Aritmética. O objetivo foi analisar a eficiência de compressão utilizando as seguintes métricas: taxa média de compressão (%) e desvio padrão amostral (%). Adicionalmente, foram obtidas as taxas máxima e mínima de compressão (%).

4.1.1 Comparação com Métodos Tradicionais e Comerciais

A Tabela 2 apresenta os resultados da avaliação de compressão sem perdas para os métodos clássicos Huffman e Aritmético, bem como para os compressores comerciais .rar, .zip, .7z, .xz e .gz. Observa-se que os métodos .rar, .zip e .7z resultaram em expansão do tamanho dos arquivos (taxas de compressão negativas). O algoritmo On+ alcançou uma taxa média de compressão de 29,19%, com desvio padrão de 1,26% e mediana de 29,09%.

A Figura 13 apresenta um gráfico boxplot comparando a taxa de compressão dos métodos tradicionais sem perdas avaliados. Observa-se que o valor mínimo do método On+ supera o máximo não-outlier dos métodos Huffman e Aritmético, ou seja, mesmo o pior caso do On+ é superior ao melhor caso típico dos concorrentes. Além disso, o On+ apresenta uma caixa mais curta (menor intervalo interquartil) e uma média (e mediana)

Tabela 2 – Resumo estatístico do desempenho de compressão (%).

Estatística	**On+**	**Huffman**	**Aritmético**	**RAR**	**ZIP**	**7Z**	**XZ**	**GZ**
Média	**29,19**	18,23	8,11	-6,28	-64,32	-69,72	20,77	26,30
Desvio Padrão	**1,26**	3,83	0,32	1,90	2,40	2,98	1,76	1,76
Mediana	**29,09**	17,45	7,93	-6,10	-64,63	-70,73	20,00	25,60

mais elevadas, indicando maior consistência e melhor desempenho médio de compressão. Por outro lado, o Huffman exibe *outliers* superiores que atingem taxas de compressão excepcionalmente altas, superando ocasionalmente o On+. Esses eventos, no entanto, são raros e constituem exceções à regra geral, não comprometendo a superioridade do On+ em cenários típicos.

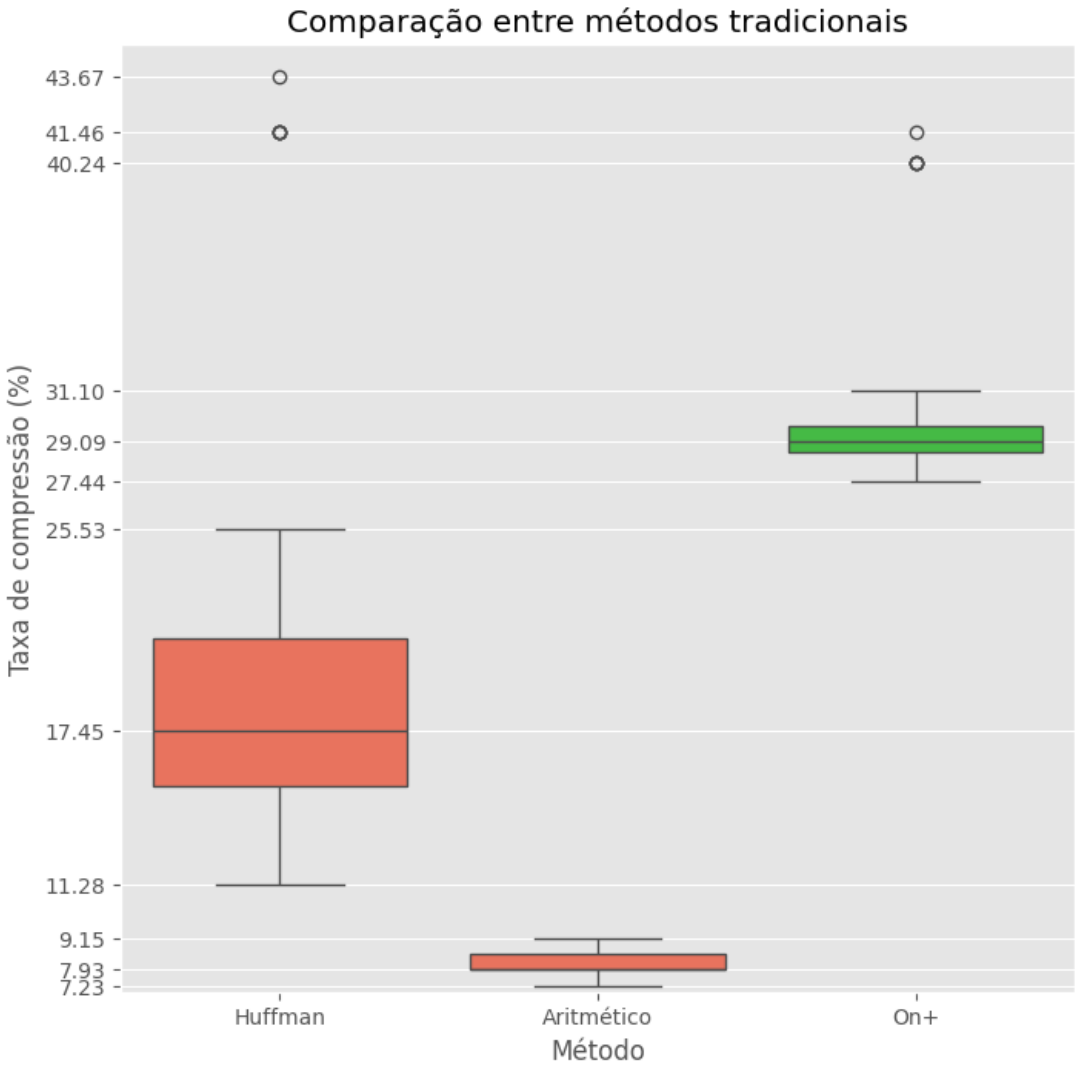

Figura 13 – Gráfico de desempenho dos métodos dos algoritmos Huffman, Aritmético e On+, utilizando dados do satélite Aldebaran-1.

A Figura 14 apresenta um boxplot com os resultados da taxa de compressão dos métodos On+, xz e gzip, que foram os únicos a apresentar compressão efetiva. Embora os

métodos xz e gzip tenham alcançado resultados superiores em alguns arquivos específicos, esses resultados não foram estatisticamente representativos.

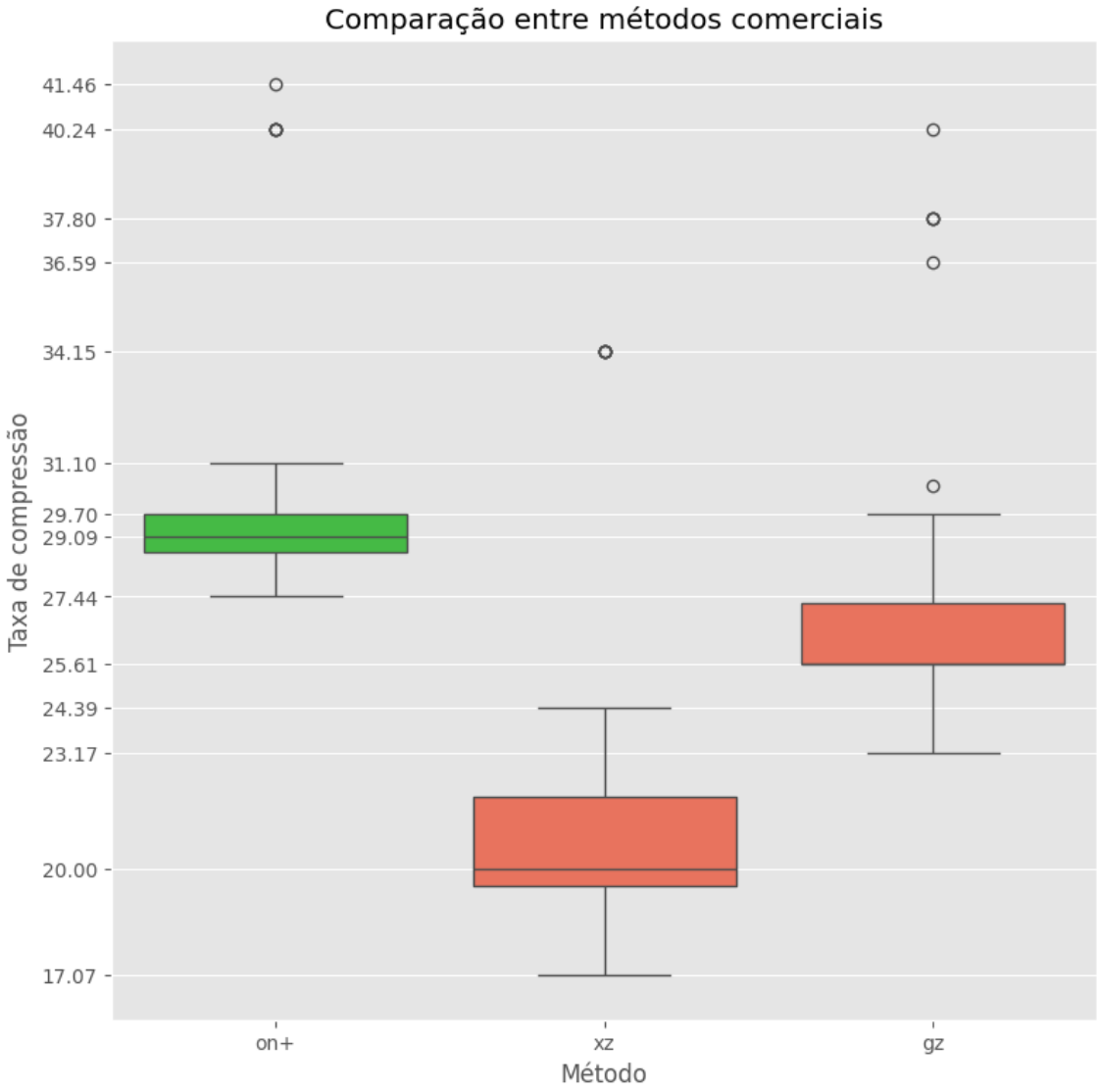

Figura 14 – Gráfico de desempenho do método On+ e dos compressores comerciais que não expandiram os dados.

4.1.2 Comparação Inter-Satélites

Além dos testes realizados com o satélite Aldebaran-1, também foram realizados testes utilizando um pacote de dados de telemetria da plataforma de satélite TinyGS. Os resultados deste teste estão apresentados em detalhes na Figuras 15 e 16.

A Figura 15 apresenta a performance do algoritmo On+ com os dados de diversos satélites da rede global *TinyGS*.

As métricas de desempenho utilizadas neste segundo teste foram as mesmas aplicadas no primeiro (Seção 4.1.1), garantindo a consistência na avaliação do algoritmo On+.

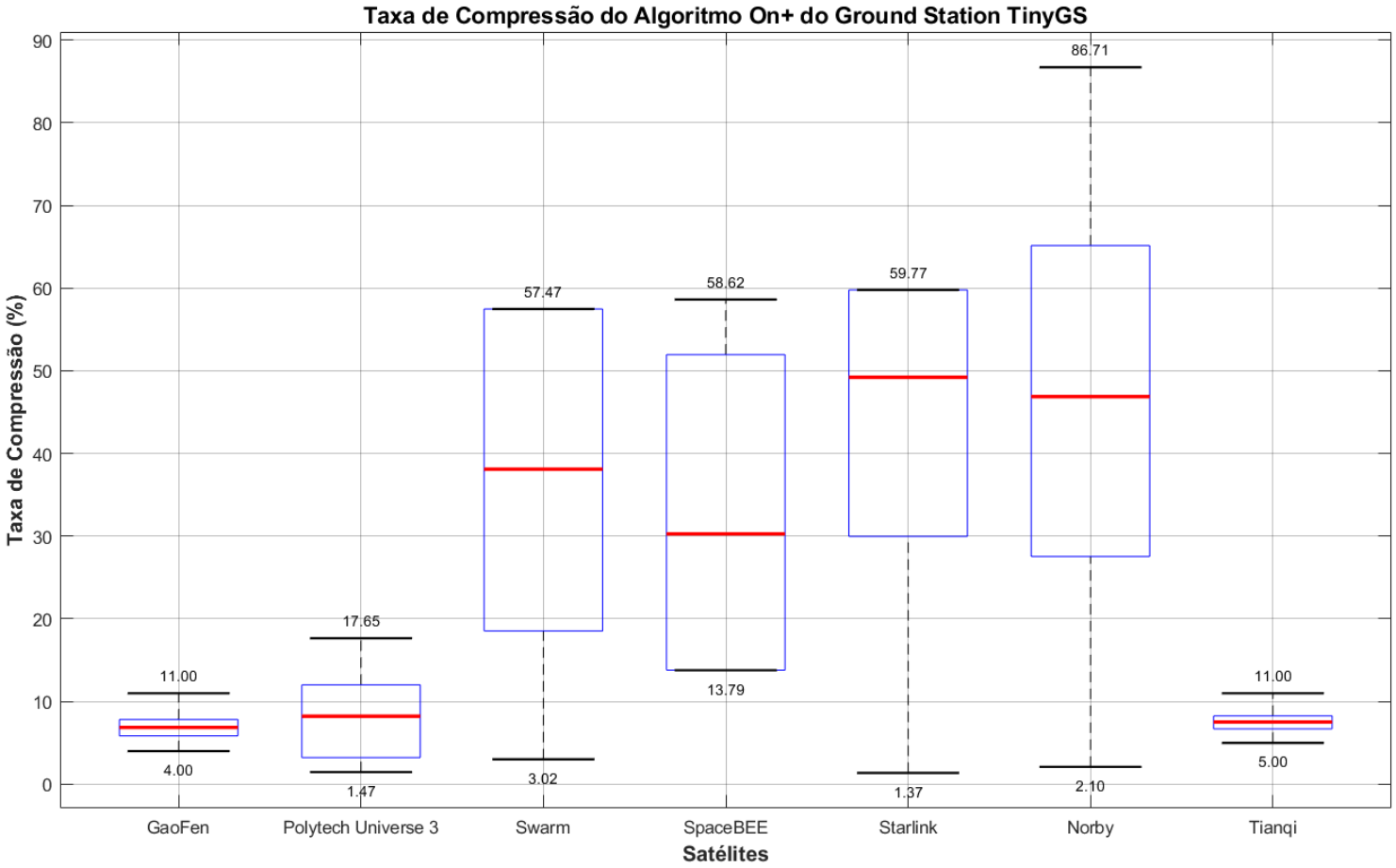

Figura 15 – Desempenho do algoritmo On+ com dados de diversos satélites da Ground Station TinyGS.

Adicionalmente, incluiu-se um parâmetro importante: o total de arquivos utilizados no experimento. Esse parâmetro é fundamental, pois reflete a diversidade dos dados analisados, que variaram conforme o satélite. Sua inclusão permite uma comparação mais abrangente e oferece uma visão mais clara do desempenho do algoritmo On+ em um conjunto heterogêneo de dados de telemetria.

A Figura 16 apresenta a quantidade de dados coletados de diferentes satélites da plataforma *TinyGS*. Observa-se que alguns satélites apresentaram expansão dos dados ao final do processo de compressão utilizando o algoritmo On+.

4.2 Análise de Entropia

Foram analisados os resultados obtidos pelo algoritmo On+, explorando suas relações com diversos fatores, como as taxas de compressão máxima e mínima, a taxa de compressão média e o cálculo da entropia. Além disso, apresentamos o comportamento do algoritmo On+ para diferentes valores de p (probabilidade de ocorrência do *bit* 1).

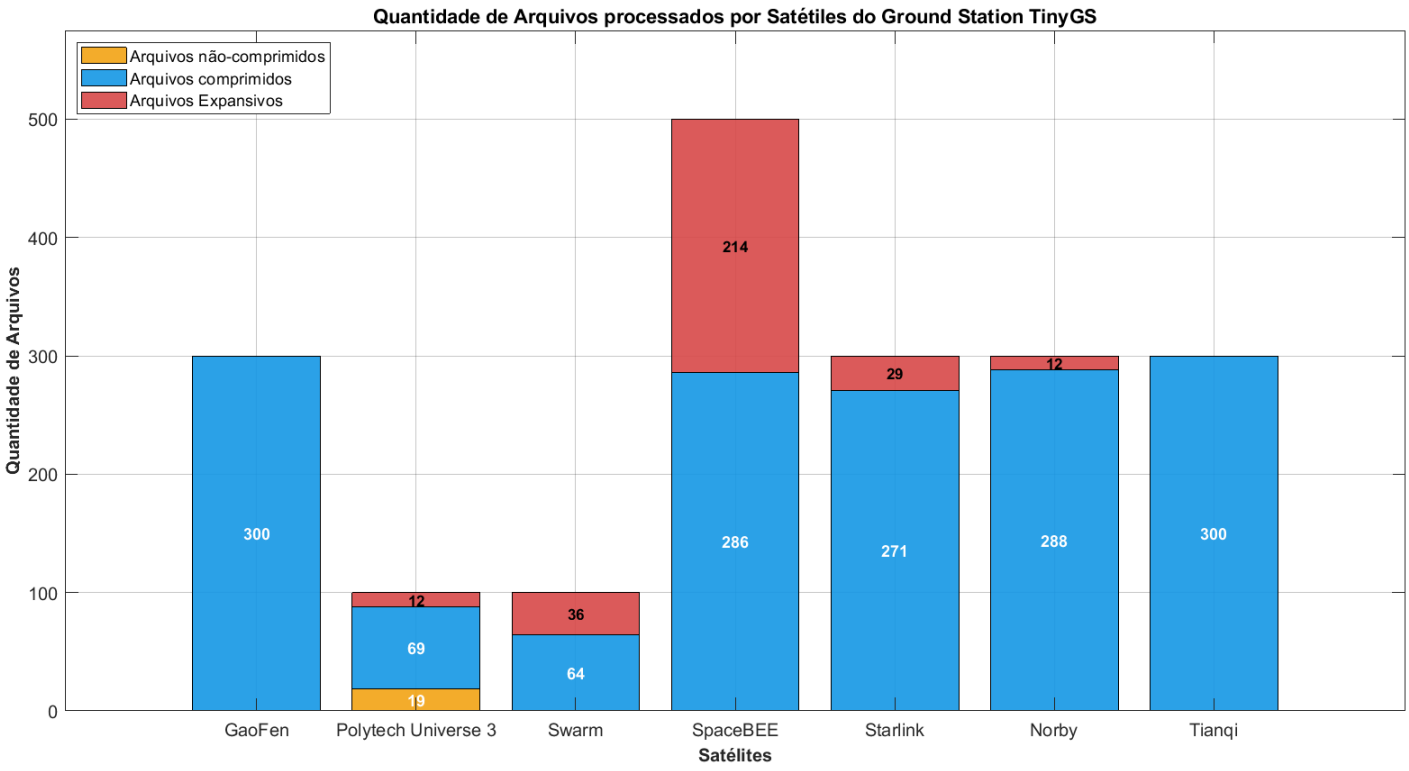

Figura 16 – Quantidade de arquivos processados de diversos satélites da *TinyGS*.

A Tabela 3 apresenta as taxas de compressão que quantificam a eficácia dos algoritmos na redução do tamanho dos dados, usando como exemplo 10 arquivos selecionados aleatoriamente. Os resultados revelam variações significativas entre os casos extremos, com valores variando de um mínimo de 27,44% a um máximo de 41,46% (incluindo valores *outliers*).

Tabela 3 – Comparação do tamanho, da taxa de compressão e da entropia entre arquivos originais e arquivos comprimidos pelo algoritmo On+, utilizando 10 arquivos selecionados aleatoriamente.

Arquivo	Tamanho Original	On+	On+ Cr.	Entropia Original	Entropia On+
File 1	0,164	**0,096**	41,46	0,875	**1,000**
File 2	0,164	**0,098**	40,24	0,876	**1,000**
File 3	0,164	**0,098**	40,24	0,875	0,999
File 4	0,164	**0,098**	40,24	0,876	0,999
File 5	0,164	**0,098**	40,24	0,877	**1,000**
File 6	0,165	**0,114**	30,91	0,887	0,998
File 7	0,164	**0,118**	28,05	0,891	**1,000**
File 8	0,164	**0,119**	27,44	0,900	**1,000**
File 9	0,164	**0,119**	27,44	0,894	0,997
File 10	0,164	**0,119**	27,44	0,892	0,999

Legenda: Tamanho original, em kB (kilobytes); On+, tamanho após a compressão pelo algoritmo On+, em kB (kilobytes); On+ Cr., taxa de compressão do On+, em %; Valores em azul indicam entropia máxima igual a 1,000.

A Figura 17 apresenta os resultados do comportamento do algoritmo On+ para diferentes valores de p (a probabilidade de ocorrência do *bit 1*). A entropia binária quantifica o grau de incerteza no vetor binário X, enquanto a probabilidade p influencia diretamente a eficiência do algoritmo On+.

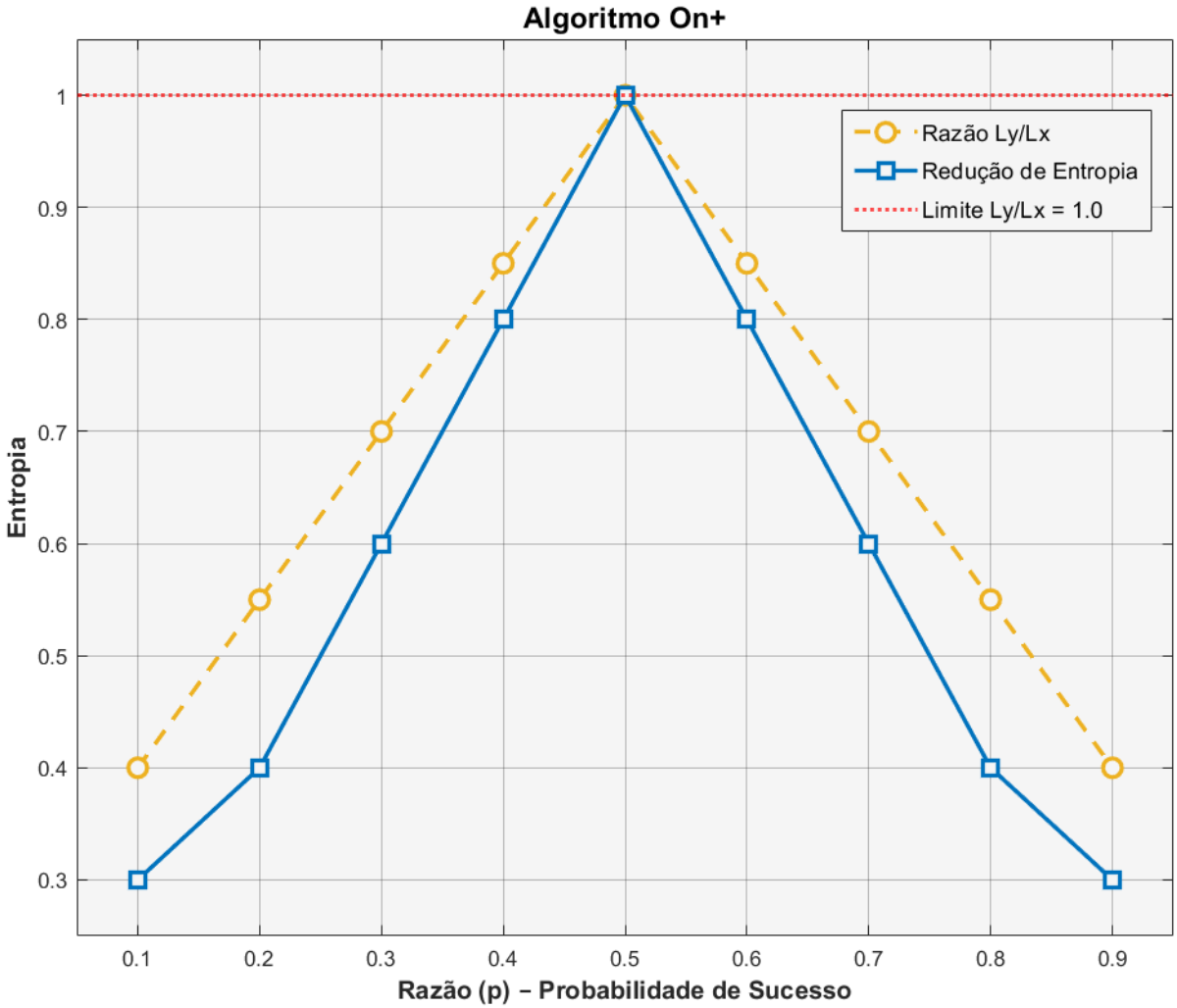

Figura 17 – Relação entre a entropia, a razão L_Y/L_X e a probabilidade de sucesso (p).

Observa-se que, na condição limite em que $\frac{L_Y}{L_X} = 1$, a compressão é ineficaz ou inexistente, o que significa que o comprimento médio da sequência comprimida (L_Y) não é menor do que o comprimento médio da sequência original (L_X).

4.3 Análise de Complexidade Temporal e Espacial

As complexidades temporal e espacial dos algoritmos de codificação e decodificação propostos foram analisadas e comparadas às dos métodos tradicionais de compressão Aritmética e Huffman, destacando suas principais vantagens e limitações. Esta análise

permite uma compreensão detalhada dos recursos computacionais necessários, contribuindo para a avaliação da eficiência do algoritmo em aplicações práticas de telemetria por satélite.

A Tabela 4 apresenta um resumo comparativo da análise de complexidade dos algoritmos de compressão sem perdas investigados para dados de telemetria por satélite. O algoritmo On+ oferece uma solução eficiente, com complexidade linear $O(n)$ tanto para codificação quanto para decodificação. Este algoritmo é particularmente vantajoso em cenários de telemetria, onde a simplicidade e a velocidade são cruciais. Em ambientes espaciais, caracterizados por largura de banda limitada e recursos de processamento restritos, a eficiência temporal e espacial do On+ representa uma vantagem significativa. A complexidade linear garante que os processos de compressão e descompressão não sobrecarreguem os recursos do satélite, consolidando o On+ como uma solução prática e eficaz.

Tabela 4 – Resumo Comparativo da Análise de Complexidade para os Algoritmos de Compressão On+, Aritmético e Huffman.

Método	**Tempo de Codificação**	**Espaço de Codificação**	**Tempo de Decodificação**	**Espaço de Decodificação**
Codificação On+	n	n	-	-
Decodificação On+	-	-	n	n
Aritmético	n	n	n	n
Huffman	$n\ log\ n$	n	n	n

Legenda: Para simplificar, estamos usando n em vez de $O(n)$ para representar a complexidade de tempo e espaço.

5 Discussões

Neste capítulo, serão apresentadas as discussões baseadas nos resultados da aplicação da técnica no Capítulo anterior. A escolha de um algoritmo de compressão para dados de telemetria de satélites é essencial devido aos requisitos rigorosos de eficiência de processamento e armazenamento, especialmente em ambientes com recursos limitados e com a necessidade de comunicação de dados em alta velocidade. Buscou-se discutir as implicações da utilização do algoritmo On+, da codificação Aritmética, da codificação de Huffman e de compressores comerciais, como rar, zip, 7z, xz e gzip, considerando as particularidades dos dados de telemetria de satélites.

A Tabela 3 mostra que a entropia dos arquivos comprimidos se aproxima de $1,000$, valor próximo do limite teórico de 1 bit/símbolo para fontes binárias. Esse resultado confirma a alta eficiência do algoritmo On+, que elimina redundâncias e produz uma distribuição uniforme de *bits*, alcançando uma compressão bem-sucedida e próxima do ideal. No entanto, o sucesso da compressão depende diretamente da estrutura dos dados originais, uma vez que padrões altamente aleatórios reduzem significativamente a eficácia da codificação.

As Figuras 13 e 14 mostram que o algoritmo On+ alcançou uma taxa mediana de 29,09%, superando o método Huffman (17,45%), a codificação Aritmética (7,93%) e os compressores comerciais xz e gzip. O método On+ apresentou uma taxa máxima de compressão de 31,10% (excluindo *outliers*) e uma taxa mínima de 27,44%. Mesmo essa taxa mínima é superior à taxa máxima típica observada para os métodos Huffman, de 25,48% (excluindo *outliers*), e para a codificação Aritmética, de 9,15%, confirmando a capacidade do On+ de explorar padrões em dados altamente estruturados. O ganho de desempenho do algoritmo On+ sobre os algoritmos aritmético e Huffman pode ser atribuído à transformação da distribuição de probabilidade de entrada em uma distribuição geométrica. Esta última apresenta uma assimetria que pode ser efetivamente explorada por codificadores entrópicos.

A Figura 15 apresenta o desempenho do método proposto sobre os dados da plataforma *TinyGS*. A maior taxa de compressão observada foi de 86,71%, obtida nos dados do satélite *Norby*, enquanto a menor foi de 1,37%, registrada nos dados do satélite *Starlink*. O satélite *Polytech Universe 3* foi o único a não apresentar compressão em todos os arquivos analisados, com 19 arquivos resultando em expansão ou manutenção do tamanho original. Por outro lado, os satélites *GaoFen* e *Tianqi* apresentaram compressão em 100% dos arquivos processados. Em termos gerais, os resultados evidenciam uma taxa de compressão mediana próxima de 50%, demonstrando a eficácia do método para essa

classe de dados de telemetria.

A Figura 16 apresenta que alguns satélites apresentaram expansão dos dados, após o processo de compressão. Esse comportamento indica a necessidade de uma análise mais aprofundada, a fim de investigar as características específicas desses conjuntos de dados e identificar os fatores que influenciaram a redução da eficiência da compressão.

Os resultados apresentados na Tabela 4 indicam que o algoritmo On+ combina eficiência computacional e simplicidade de implementação, mantendo complexidade temporal e espacial linear tanto na codificação quanto na decodificação. Essa característica é particularmente relevante para aplicações de telemetria por satélite, nas quais os recursos computacionais disponíveis são limitados e a transmissão de dados deve ocorrer de forma contínua e confiável. Diferentemente de abordagens que demandam estruturas de dados mais complexas ou maior capacidade de processamento, o comportamento linear do On+ favorece sua utilização em sistemas embarcados, contribuindo para a previsibilidade do consumo de recursos e para a escalabilidade do processamento à medida que o volume de dados aumenta.

A análise da complexidade temporal do processo de codificação, apresentada no Algoritmo 1, considerou a inicialização das variáveis, as instruções condicionais, o laço principal e a etapa de codificação Aritmética adaptativa. Inicialmente, o vetor binário de entrada X é percorrido uma única vez para contabilizar sequências consecutivas de zeros e construir o vetor $X_{geometric}$. Como cada elemento é processado apenas uma vez e todas as operações realizadas possuem custo constante, essa etapa apresenta complexidade linear.

Uma vez gerado o vetor $X_{geometric}$, seus símbolos são submetidos à função *Adaptive-Arithmetic*, a qual os processa sequencialmente. Consequentemente, o tempo de execução total cresce proporcionalmente ao tamanho da entrada n, resultando em complexidade temporal $O(n)$.

A análise da complexidade espacial do Algoritmo 1 considerou as variáveis locais, o vetor auxiliar $X_{geometric}$ e o vetor de saída *encoded*. O consumo de memória é dominado pelo armazenamento dessas estruturas, cujos tamanhos são proporcionais ao número de elementos processados. Como não são empregadas estruturas com crescimento superior ao linear, a complexidade espacial da codificação também é $O(n)$.

A análise da complexidade temporal do processo de decodificação, descrito no Algoritmo 2, considerou a leitura do vetor *encoded*, a execução da função *AdaptiveArithmetic(encoded)* e a reconstrução da sequência binária original. Inicialmente, a decodificação Aritmética gera o vetor $X_{geometric}$, contendo os comprimentos das sequências codificadas. Em seguida, cada elemento desse vetor é utilizado para reconstruir a sequência original.

Embora a quantidade de zeros inseridos varie a cada iteração, o número total de bits adicionados ao vetor *decoded* é proporcional ao tamanho final da sequência reconstruída.

Dessa forma, tanto a decodificação Aritmética quanto a reconstrução dos dados executam um número de operações proporcional à quantidade de elementos processados, resultando em complexidade temporal $O(n)$.

A análise da complexidade espacial do Algoritmo 2 considerou as variáveis locais, o vetor intermediário $X_{geometric}$ e o vetor *decoded*. O consumo de memória é dominado pelo armazenamento dessas estruturas, enquanto operações auxiliares, como leitura do fluxo codificado e remoção do último *bit*, demandam espaço desprezível. Como o tamanho do vetor reconstruído cresce proporcionalmente ao tamanho da entrada, a complexidade espacial da decodificação também é $O(n)$.

A codificação Aritmética adaptativa apresenta complexidade temporal linear para codificação e decodificação, ambas iguais a $O(n)$, uma vez que os dados são processados sequencialmente (Said, 2023). Da mesma forma, suas complexidades espaciais também são lineares, pois a memória requerida é proporcional ao tamanho da entrada e da saída comprimida.

Por sua vez, a compressão de Huffman possui complexidade temporal de codificação $O(n \log n)$ devido à construção da árvore de Huffman, que envolve operações de custo logarítmico (Tian et al., 2021). Em contrapartida, a decodificação apresenta complexidade temporal linear $O(n)$, utilizando a árvore previamente construída para reconstruir os dados. Tanto na codificação quanto na decodificação, a complexidade espacial permanece $O(n)$, em função do armazenamento da árvore, da tabela de códigos e dos dados processados.

6 Conclusão

A escolha de um algoritmo de compressão para dados de telemetria de satélite é essencial devido aos requisitos rigorosos de eficiência de processamento e armazenamento, especialmente em ambientes de *hardware* restrito e necessidade de comunicação de dados em alta velocidade. Os satélites coletam uma ampla variedade de dados, como temperatura, pressão, nível de bateria e posição. Devido às limitações de recursos a bordo, otimizar a comunicação de forma eficiente é fundamental.

Neste estudo, foi demonstrada a viabilidade da aplicação da compressão por transformação geométrica a dados de telemetria de satélites. A abordagem proposta foi avaliada por meio de experimentos com dados reais provenientes do satélite Aldebaran-1 e da plataforma *TinyGS*, sendo comparada com métodos tradicionais de compressão sem perdas, como a codificação de Huffman e a codificação Aritmética, além de compressores amplamente utilizados, como rar, zip, 7z, xz e gzip.

Os resultados obtidos evidenciaram o desempenho destacado do algoritmo On+ no conjunto de dados do satélite Aldebaran-1, baseado em modulação LoRa. O método On+ alcançou compressão em todos os 600 arquivos extraídos do satélite Aldebaran-1, com uma taxa de compressão máxima de 31,10% (excluindo *outliers*), uma taxa mínima de 27,44% e uma mediana de 29,09%. Os resultados mostram que o método On+ é superior aos métodos Huffman e Aritmético tanto em termos de taxa média de compressão quanto em consistência. Observou-se que, mesmo nos piores casos, o On+ apresentou taxas de compressão superiores aos melhores resultados obtidos pelos métodos alternativos analisados, excetuando-se apenas os casos de expansão de dados (*outliers*) detalhados na análise experimental. Além disso, o método On+ apresentou baixo desvio padrão em comparação com os demais métodos, com exceção do método Aritmético, evidenciando a robustez e a consistência da abordagem proposta.

Os experimentos realizados com dados provenientes da plataforma *TinyGS* também produziram resultados expressivos. De forma geral, observou-se uma taxa de compressão mediana próxima de 50%, reforçando a capacidade da abordagem proposta de explorar padrões estatísticos presentes em diferentes tipos de telemetria de satélites.

Além dos ganhos obtidos em compressão, a análise de complexidade demonstrou que os processos de codificação e decodificação do On+ apresentam complexidade temporal e espacial linear, $O(n)$, característica especialmente relevante para aplicações embarcadas e sistemas com recursos computacionais restritos. Essa propriedade favorece sua adoção em ambientes espaciais, nos quais a eficiência computacional é tão importante quanto a redução do volume de dados transmitidos.

Dessa forma, os resultados obtidos confirmam que a compressão por transformação geométrica consolida-se como uma solução robusta e viável para a compressão sem perdas de dados de telemetria de satélites, combinando elevadas taxas de compressão, baixo custo computacional e comportamento consistente em diferentes conjuntos de dados.

6.1 Trabalhos Futuros

Como perspectivas para trabalhos futuros, pretende-se implementar o algoritmo On+ a bordo do satélite Aldebaran-1 em futuras missões, como uma etapa de compressão que antecede a transmissão dos dados de telemetria.

Para viabilizar essa implementação embarcada, duas frentes de aprimoramento mostram-se essenciais: *i)* otimização do tempo de execução: pretende-se aprofundar a análise de desempenho temporal do On+, avaliando métricas como tempos mínimo, máximo e médio de compressão sob diferentes volumetrias e perfis de tráfego de telemetria. O objetivo é garantir que o algoritmo atenda às restrições de processamento e às janelas de transmissão de um satélite de pequeno porte; e *ii)* redução do uso de memória: considerando o ambiente de *hardware* restrito do Aldebaran-1, planeja-se investigar o consumo de memória (RAM e flash) do algoritmo On+, propondo versões mais leves que mantenham elevadas taxas de compressão sem comprometer a estabilidade das missões aeroespaciais.

Outra linha de investigação consiste no aprimoramento do On+ por meio da incorporação de técnicas de predição (Millidge et al., 2021) e modelagem de contexto, explorando a similaridade existente entre pacotes consecutivos de telemetria (Rakhmanov and Wiseman, 2023). Adicionalmente, pretende-se ampliar a avaliação experimental com pacotes de dados maiores e provenientes de diferentes satélites, a fim de verificar seu impacto no desempenho da compressão. Essa ampliação fortalecerá a análise experimental e aumentará a robustez e a capacidade de generalização dos resultados.

Por fim, os resultados apresentados neste trabalho indicam que o algoritmo On+ constitui uma alternativa promissora para a compressão sem perdas de dados de telemetria, com potencial de aplicação não apenas em satélites e veículos espaciais, mas também em sistemas de monitoramento remoto e dispositivos da Internet das Coisas (IoT), nos quais a eficiência na transmissão e no armazenamento de dados é um requisito fundamental.

Referências

Adel, M., El-Naggar, M., Darweesh, M. S., and Mostafa, H. (2018). Multiple hybrid compression techniques for electroencephalography data. In *2018 30th International Conference on Microelectronics (ICM)*, pages 124–127. Citado na página 26.

Adjeroh, D., Bell, T., and Mukherjee, A. (2008). *The burrows-wheeler transform: data compression, suffix arrays, and pattern matching*. Springer. Citado na página 29.

Anmireddy, V., Vasudevan, R., Anand, D., Rao, T. V., Kapardhi, B. V. N., Trivedi, D., and Manchanda, R. K. (2010). Telemetry, telecommand and safety sub-systems for scientific ballooning from hyderabad. *Advances in Space Research*, 46(7):960–967. Citado na página 13.

Arnavut, Z. (2000). Move-to-front and inversion coding. In *Proceedings DCC 2000. Data Compression Conference*, pages 193–202. IEEE. Citado na página 29.

Artikis, T., Loukas, S., and Jerwood, D. (1998). A transformed geometric distribution in stochastic modelling. *Mathematical and computer modelling*, 27(1):43–51. Citado na página 29.

Barannik, V., Havrylov, D., Barannik, V., Dodukh, A., Gancarczyk, T., and Gowin, K. (2018). Development of adaptive arithmetic coding method to the sequence of bits. In *International Conference of Students, PhD Students and Young Scientists"Engineer of the XXI Century*
, pages 203–208. Springer. Citado na página 29.

Barros, A. K. (2025). Entropy as a geometric consequence of higher dimensions. *Technologies*, 13(12):563. Citado na página 22.

Barros, F., Correia, L., Magno, C., Diniz, C., Sousa, G., Barros, A. K., and Claudio, L. C. (2026). Lossless compression of aldebaran-i telemetry data using the on+ algorithm. *Technologies*, 14(353). Citado na página 16.

Betetto, L. A. d. O. (2005). Método para compressão de imagens digitais fundamentado em procedimentos de huffman e wavelets. Citado na página 23.

Blelloch, G. E. (2001). Introduction to data compression. *Computer Science Department, Carnegie Mellon University*, 54. Citado 2 vezes nas páginas 24 e 29.

Brian, K. (2020). Lossless compression with asymmetric numeral systems. Blog post. Citado 2 vezes nas páginas 22 e 24.

Cayoglu, U., Tristram, F., Meyer, J., Schröter, J., Kerzenmacher, T., Braesicke, P., and Streit, A. (2019). Data encoding in lossless prediction-based compression algorithms. In *2019 15th International Conference on eScience (eScience)*, pages 226–234. Citado na página 38.

Chan, S. H. (2026). *Introduction to Probability for Data Science*. Michigan Publishing, Ann Arbor, MI, 2nd edition. Citado na página 29.

Ciaparrone, G., Benedetto, V., and Gissi, F. (2022). A preliminary study on ai for telemetry data compression. In *International Conference on Deep Learning, Artificial Intelligence and Robotics*, pages 134–143. Citado 3 vezes nas páginas 13, 16 e 17.

Durrett, R. (2019). *Probability: theory and examples*, volume 49. Cambridge university press. Citado na página 29.

Elzeiny, S., Edward, P., and Elshabrawy, T. (2021). Lora performance enhancement through list decoding technique. In *2021 IEEE International Conference on Communications Workshops (ICC Workshops)*, pages 1–6. IEEE. Citado na página 41.

Evans, D., Labrèche, G., Marszk, D., Bammens, S., Hernández-Cabronero, M., Zelenevskiy, V., Shiradhonkar, V., Starcik, M., and Henkel, M. (2022). Implementing the new ccsds 124.0-b-1 housekeeping data compression standard (based on pocket+) on ops-sat-1. Citado 2 vezes nas páginas 13 e 17.

Gilchrist, J. (2004). Parallel data compression with bzip2. In *Proceedings of the 16th IASTED international conference on parallel and distributed computing and systems*, volume 16, pages 559–564. Citeseer. Citado 2 vezes nas páginas 23 e 24.

Guojun, L., Jian, S., and Running, Z. (2013). Lossless data compression algorithm for satellite packet telemetry data. In *Proceedings 2013 International Conference on Mechatronic Sciences, Electric Engineering and Computer (MEC)*, pages 2756–2759. Citado na página 13.

Hao, Y., Zhang, C., Chai, S., Li, Z., and Liu, X. (2020). Satellite fault detection and diagnosis based on data compression and improved decision tree. In *2020 Chinese Automation Congress (CAC)*, pages 1686–1691. Citado 3 vezes nas páginas 13, 16 e 17.

Hernández-Cabronero, M., Evans, D., Bartrina-Rapesta, J., Aulí-Llinàs, F., Blanes, I., and Serra-Sagristà, J. (2024). Resiliency and efficiency of the ccsds 124.0-b-1 telemetry compression standard. *IEEE Access*. Citado 2 vezes nas páginas 13 e 17.

Jayasankar, U., Thirumal, V., and Ponnurangam, D. (2021). A survey on data compression techniques: From the perspective of data quality, coding schemes, data type and applications. *Journal of King Saud University-Computer and Information Sciences*, 33(2):119–140. Citado 2 vezes nas páginas 26 e 38.

Jeong, S., Jeong, S., Woo, S. S., and Ko, J. H. (2023). An overhead-free region-based jpeg framework for task-driven image compression. *Pattern Recognition Letters*, 165:1–8. Citado na página 13.

Karam, L. (2009). *Lossless Image Compression*, pages 385–419. Citado 2 vezes nas páginas 25 e 26.

Ketshabetswe, L. K., Zungeru, A. M., Lebekwe, C. K., and Mtengi, B. (2024). Energy-efficient algorithms for lossless data compression schemes in wireless sensor networks. *Scientific African*, 23:e02008. Citado na página 13.

Khan, N., Iqbal, K., and Martini, M. G. (2020). Lossless compression of data from static and mobile dynamic vision sensors-performance and trade-offs. *IEEE Access*, 8:103149–103163. Citado 2 vezes nas páginas 25 e 26.

Khandwani, F. I. and Ajmire, P. E. (2018). A survey of lossless image compression techniques. *International Journal of Electrical Electronics & Computer Science Engineering*, 5(1):39–42. Citado 2 vezes nas páginas 23 e 24.

Langdon, G. G. (1984). An introduction to arithmetic coding. *IBM Journal of Research and Development*, 28(2):135–149. Citado na página 26.

Lukin, V., Vozel, B., and Serra-Sagristà, J. (2021). *Remote Sensing Data Compression*. MDPI. Citado na página 24.

Marques, J., Santos, J., Azulay, L., Junior, C., Filho, E., Junior, J., and Silva, L. (2022). Thermal simulation of a high altitude balloon. Citado 2 vezes nas páginas 40 e 41.

McAnlis, C. and Haecky, A. (2016). *Understanding compression: Data compression for modern developers*. "O'Reilly Media, Inc.". Citado 3 vezes nas páginas 9, 25 e 26.

Meß, J.-G., Schmidt, R., and Fey, G. (2017). Adaptive compression schemes for housekeeping data. In *2017 IEEE Aerospace Conference*, pages 1–12. IEEE. Citado na página 13.

Meß, J.-G., Schmidt, R., Fey, G., and Dannemann, F. (2016). On the compression of spacecraft housekeeping data using discrete cosine transforms. In *2016 International Workshop on Tracking, Telemetry and Command Systems for Space Applications (TTC)*, pages 1–8. Citado na página 13.

Millidge, B., Seth, A., and Buckley, C. L. (2021). Predictive coding: a theoretical and experimental review. *arXiv preprint arXiv:2107.12979*. Citado 2 vezes nas páginas 29 e 54.

Mishra, S. and Singh, S. (2016). A survey paper on different data compression techniques. *Indian J. Res. Pap*, 6(5). Citado na página 26.

Patel, H., Itwala, U., Rana, R., and Dangarwala, K. (2015). Survey of lossless data compression algorithms. *International Journal of Engineering Research and Technology*, 4(4):926–929. Citado 5 vezes nas páginas 21, 23, 24, 25 e 26.

Peng, X., Zhang, Y., Peng, D., and Zhu, J. (2023). Selective run-length encoding. *arXiv preprint arXiv:2312.17024*. Citado na página 29.

Phalke, S., Vaidya, Y., and Metkar, S. (2022). Big-o time complexity analysis of algorithm. In *2022 International Conference on Signal and Information Processing (IConSIP)*, pages 1–5. Citado 2 vezes nas páginas 37 e 38.

Pu, I. M. (2006). *Fundamental Data Compression*. Elsevier Science Technology. Citado 6 vezes nas páginas 9, 19, 20, 21, 22 e 23.

Rakhmanov, A. and Wiseman, Y. (2023). Compression of gnss data with the aim of speeding up communication to autonomous vehicles. *Remote Sensing*, 15(8):2165. Citado 2 vezes nas páginas 13 e 54.

Said, A. (2023). Introduction to arithmetic coding–theory and practice. *arXiv preprint arXiv:2302.00819*. Citado na página 52.

Sayood, K. (2018). *Introduction to Data Compression*. Morgan Kaufmann. Citado 6 vezes nas páginas 9, 21, 24, 25, 27 e 29.

Shannon, C. E. (1948). A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423. Citado 4 vezes nas páginas 9, 19, 20 e 22.

Shehab, A. F., Elshafey, M. A., and Mahmoud, T. A. (2020). Recurrent neural network based prediction to enhance satellite telemetry compression. In *2020 IEEE Aerospace Conference*, pages 1–11. Citado 3 vezes nas páginas 13, 16 e 17.

Sneyers, J. and Wuille, P. (2016). Flif: Free lossless image format based on maniac compression. In *2016 IEEE international conference on image processing (ICIP)*, pages 66–70. IEEE. Citado na página 26.

Tian, J., Rivera, C., Di, S., Chen, J., Liang, X., Tao, D., and Cappello, F. (2021). Revisiting huffman coding: Toward extreme performance on modern gpu architectures. In *2021 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 881–891. Citado na página 52.

TinyGS Community (2026). Tinygs — open source global satellite network. Site oficial do projeto. Rede global de estações terrestres de código aberto para rastreamento de satélites usando hardware LoRa de baixo custo. Citado 2 vezes nas páginas 40 e 41.

Vaz, R., Shah, V., Sawhney, A., and Deolekar, R. (2017). Automated big-o analysis of algorithms. In *2017 International Conference on Nascent Technologies in Engineering (ICNTE)*, pages 1–6. Citado 2 vezes nas páginas 37 e 38.

Wan, P., Zhan, Y., and Jiang, W. (2019). Study on the satellite telemetry data classification based on self-learning. *IEEE Access*, 8:2656–2669. Citado 3 vezes nas páginas 13, 16 e 17.

Wang, Z., Wen, M., Xu, Y., Zhou, Y., Wang, J. H., and Zhang, L. (2023). Communication compression techniques in distributed deep learning: A survey. *Journal of Systems Architecture*, page 102927. Citado na página 14.

Xu, Z., Cheng, Z., and Guo, B. (2023a). A hybrid data-driven framework for satellite telemetry data anomaly detection. *Acta Astronautica*, 205:281–294. Citado na página 13.

Xu, Z., Cheng, Z., Tang, Q., and Guo, B. (2023b). An encoder-decoder generative adversarial network-based anomaly detection approach for satellite telemetry data. *Acta Astronautica*, 213:547–558. Citado na página 13.

Yuan, S. and Hu, J. (2019). Research on image compression technology based on huffman coding. *Journal of Visual Communication and Image Representation*, 59:33–38. Citado na página 25.