JULIANO VIEIRA MARTINS

Compressão de sequências genômicas baseada em formatos de arquivo de imagem

Dissertação apresentada ao Programa de Pós-Graduação em Informática da Pontifícia Universidade Católica do Paraná (PUCPR) como requisito parcial para obtenção do título de Mestre em Informática.

Curitiba
2018

JULIANO VIEIRA MARTINS

Compressão de sequências genômicas baseada em formatos de arquivo de imagem

Dissertação apresentada ao Programa de Pós-Graduação em Informática da Pontifícia Universidade Católica do Paraná (PUCPR) como requisito parcial para obtenção do título de Mestre em Informática.

Área de concentração: Ciência da Computação .

Orientador: Bráulio Coelho Ávila
Coorientador: Roberto Hirochi Herai

Curitiba
2018

Dados da Catalogação na Publicação
Pontifícia Universidade Católica do Paraná
Sistema Integrado de Bibliotecas – SIBI/PUCPR
Biblioteca Central
Edilene de Oliveira dos Santos CRB 9 / 1636

M386c
2018

Martins, Juliano Vieira
Compressão de sequências genômicas baseada em formatos de arquivo de imagem / Juliano Vieira Martins ; orientador, Braúlio Coelho Ávila ; cooorientador, Roberto Hirochi Herai. – 2018
72 f. : il. ; 30 cm

Dissertação (mestrado) – Pontifícia Universidade Católica do Paraná, Curitiba, 2018
Bibliografia: f. 67-72

1. Informática. 2. Compressão de dados (Computação). 3. Genoma humano. 4. DNA. 5. Biotecnologia. I. Ávila, Braúlio Coelho. II. Herai, Roberto Hirochi. III. Pontifícia Universidade Católica do Paraná. Programa de Pós-Graduação em Informática. III. Título.

CDD 20. ed. – 004.068

Pontifícia Universidade Católica do Paraná

PUCPR
GRUPO MARISTA

ATA DE SESSÃO PÙBLICA

DEFESA DE DISSERTAÇÃO DE MESTRADO Nº 05/2018

PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA – PPGIa
PONTIFÍCIA UNIVERSIDADE CATÓLICA DO PARANÁ - PUCPR

Em sessão pública realizada às 13h30 de **04 de Maio de 2018**, no Auditório Guglielmo Marconi – Bloco 8, ocorreu a defesa da dissertação de mestrado intitulada "Compreensão de Sequencias Genômicas baseada em Formatos de Imagem" apresentada pelo aluno **Juliano Vieira Martins**, como requisito parcial para a obtenção do título de **Mestre em Informática**, na área de concentração **Ciência da Computação**, perante a banca examinadora composta pelos seguintes membros:

Prof. Dr. Bráulio Coelho Avila (Orientador)- PUCPR

Prof. Dr. Roberto Hirochi Herai (co-orientador) – PUCPR/PPGCS

Prof. Dr. Edson Emílio Scalabrin – PUCPR

Prof. Dr. Mauri Ferrandin – UFSC

Prof. Dr. André Pinz Borges – UTFPR

Após a apresentação da dissertação pelo aluno e correspondente arguição, a banca examinadora emitiu o seguinte parecer sobre a tese:

Membro	Parecer	
Prof. Dr. Bráulio Coelho Avila	( ) Aprovado	( ) Reprovado
Prof. Dr. Roberto Hirochi Herai	(√) Aprovado	( ) Reprovado
Prof. Dr. Edson Emílio Scalabrin	(√) Aprovado	( ) Reprovado
Prof. Dr. Mauri Ferrandin	(√) Aprovado	( ) Reprovado
Prof. Dr. André Pinz Borges	(√) Aprovado	( ) Reprovado

Portanto, conforme as normas regimentais do PPGIa e da PUCPR, a tese foi considerada:

(√) **APROVADO**

(aprovação condicionada ao atendimento integral das correções e melhorias recomendadas pela banca examinadora, conforme anexo, dentro do prazo regimental)

( ) **REPROVADO**

E, para constar, lavrou-se a presente ata que vai assinada por todos os membros da banca examinadora. Curitiba, 04 de Maio de 2018.

Prof. Dr. Bráulio Coelho Avila

Prof. Dr. Roberto Hirochi Herai

Prof. Dr. Edson Emílio Scalabrin

Prof. Dr. Mauri Ferrandin

Prof. Dr. André Pinz Borges

Dedico essa pesquisa à Deus que tem me sustentado até aqui, que foi o autor e o consumador da minha fé nessa caminhada.

Agradecimentos

À Deus, por ter iluminado meu caminho em mais essa jornada e feito com que eu chegasse até aqui.

À minha esposa Silvia, pessoa com quem amo partilhar a vida. Com você tenho me sentido amparado. Obrigado pelo carinho, a paciência e por sua capacidade de acreditar em mim e de me trazer paz na correria de cada desafio.

À minha filha Aline e ao meu filho Juliano que, com muito carinho e apoio, não mediram esforços para que eu chegasse até esta etapa de minha vida.

À minha família, mãe e irmãos, por sempre terem uma palavra de incentivo. Mãe, seu cuidado e dedicação foi que deram, em alguns momentos, a esperança para seguir.

À Pontifícia Universidade Católica do Paraná (PUCPR) e ao Programa de Pós-Graduação em Informática aplicada (PPGIa).

Ao meu orientador Bráulio C. Ávila, que se portou como só fazem os mestres. Acreditando no meu trabalho, deu-me a liberdade necessária para pesquisar, dividindo comigo as expectativas, conduziu-me a maiores reflexões enriquecendo meu trabalho. Minha especial admiração e gratidão.

Ao professor Edson E. Scalabrin, que foi mais que um mestre, um verdadeiro amigo nessa caminhada. Meus sinceros agradecimentos.

Ao professor Roberto H. Herai, que emprestou-me seu precioso tempo para conduzir-me nesse estudo. Meu muito obrigado.

Aos professores Fabrício Enembreck, Sheila Reinehr, Andreia Malucelli, Jacques Facon e Carlos N. Silla Jr., que ministraram algumas das disciplinas que eu participei no decorrer do mestrado. Meus agradecimentos.

Aos amigos Kelvin V. Kredens, Osmar B. Dordal, Jean P. Barddal, Luis E. B. Ferreira e Heitor M. Gomes que sempre compartilharam suas experiências para enriquecer meus conhecimentos. Agradeço a todos e a cada um em particular.

Ao pessoal da Secretaria Acadêmica, suporte e recepção, pela eficiência, dedicação e simpatia. Muito obrigado.

Sumário

Lista de Algoritmos

Lista de Figuras

Lista de Tabelas

Lista de Acrônimos

ADN *Ácido desoxirribonucleico*

AdpISPO *Adaptive Intelligent Single Particle Optimizer*

ARV *Approximate Repeat Vector*

ASCII *American Standard Code for Information Interchange*

BPG *Better Portable Graphics*

BWT *Burrows–Wheeler Transform*

CD *Critical Difference*

CLPSO *Comprehensive Learning Particle Swarm Optimization*

CoGI *Compressing Genomes as a Image*

DNA *Deoxyribonucleic Acid*

FLIF *Free Lossless Image Format*

GIF *Graphics Interchange Format*

HGP *Human Genome Project*

HTS *High-Throughput Sequencing*

IUPAC *International Union of Pure and Applied Chemistry*

JPEG XR *Joint Photographic Experts Group Extended Range*

LZMA *Lempel–Ziv Markov chain algorithm*

MP3 *MPEG-1 Audio Layer III*

MTF *Move To Front transform*

NCBI *National Center for Biotechnology Information*

NGS *Next-generation sequencing*

NHGRI *National Human Genome Research Institute*

NML *Normalized Maximum Likelihood*

PNG *Portable Network Graphics*

POMA *Adaptative Particle Swarm Optimization-based Memetic Algorithm*

PPMd *Prediction by partial matching*

PUCPR Pontifícia Universidade Católica do Paraná

RLE *Run Length Encoding*

RSL *Revisão Sistemática da Literatura*

SRA *Sequence Read Archive*

WAVE *Waveform Audio File Format*

WBTC *Word-Based Tagged Code*

XFCMs *Extended Finite-Context Models*

Resumo

Uma quantidade sem precedentes de dados digitais decorrente de sequências genômicas, está sendo gerada atualmente com o surgimento das plataformas de "sequenciamento de próxima geração", comumente chamada (NGS). Assim, a demanda para o armazenamento e a transmissão de dados de sequências genômicas têm incentivado a realização de esforços para aumentar a economia de espaço de armazenamento e tempo de processamento das sequências de DNA. Para aumentar tal economia, uma tripla hipótese foi feita: a) a abordagem horizontal e sem perda é efetiva para a compressão de dados de sequências genômicas; b) o uso de formatos de arquivos de imagens como *FLIF* e *WebP* para a compressão de sequências genômicas é uma alternativa viável; e c) a transformação do alfabeto {A, T, C, G} com o objetivo de reduzir a entropia da informação, bem como o tamanho da sequência genômica, resulta em maior economia de espaço. O método desenvolvido nesta pesquisa incorpora o resultado desta tripla hipótese. Ele patenteia um método, de compressão de dados de sequências genômicas, viável nos seguintes termos: compressão com abordagem horizontal sem perda de dados e baseado em formato de arquivo de imagem. Os resultados são estatisticamente similares quando comparada a economia de espaço do método proposto com a economia de espaço obtida por ferramentas especializadas.

Palavras-chave: Compressão de genomas; Compressão sem perda; Compressão horizontal.

Abstract

An unprecedented amount of digital genomic data sequences is currently being generated with the advent of Next Generation Sequencing (NGS) platforms. Therefore, the demand for storage and transmission of genomic data sequence has encouraged researchers and practitioners on the development of light-weighted compression techniques for genomic sequences regarding processing time and memory consumption. In this work, we hypothesize the following: a) the lossless horizontal approach is effective for the compression of genomic sequence data; b) the use of image file formats such as *FLIF* and *WEBP* for the genomic compression is viable; and c) the transformation of the genomic alphabet {A, T, C, G} to reduce the information entropy, as well as the size of the genomic sequence, results in significant space savings. The proposed method incorporates the hypotheses mentioned above. The results of an empirical evaluation demonstrate that the proposed method is statistically similar to the state-of-the-art regarding data compression space savings.

Keywords: Genome compression; Lossless compression; Horizontal compression.

Capítulo 1

Introdução

Iniciado em 1990, o *Human Genome Project* (HGP) foi um dos grandes feitos de exploração de dados de sequência de *Deoxyribonucleic Acid* (DNA) na história da humanidade (LANDER et al., 2001). O HGP deu-se pelo esforço de uma pesquisa internacional para sequenciar e mapear todos os genes de alguns membros da espécie *Homo sapiens* que juntos são conhecidos como Genoma (National Human Genome Research Institute, 2015). Concluído em abril de 2003, o HGP permitiu ao homem, pela primeira vez, ler completamente o projeto genético da natureza para a construção de um ser humano. Tal avanço contribuiu para o desenvolvimento de novas tecnologias de sequenciamento genético e a utilização destas fez com que o custo do sequenciamento de genoma baixasse de 95 milhões de dólares (setembro de 2001) para 1.121 dólares (julho de 2017) (Wetterstrand KA, 2016). Em contrapartida, a queda do custo para sequenciar genoma tem proporcionado o acúmulo dos dados cuja tendência segue uma trajetória exponencial. O armazenamento e transferência desses dados é um problema substancial e, ao mesmo tempo, algo necessário ao desenvolvimento de ferramentas de compressão de dados para fins genéticos.

Os recentes esforços no desenvolvimento de novos algoritmos e ferramentas para armazenar e gerenciar dados de sequenciamento genômico, mostram crescentes demandas por métodos mais eficientes para a compressão destes dados. Essa demanda é alvo de um projeto de pesquisa, cujo presente trabalho faz parte, sobre compressão sem perda de sequências genômicas. Tal projeto está sendo realizado em um importante binômio de laboratórios de pesquisa da PUCPR, a saber: Agentes de Software (Escola Politécnica) e Bioinformática (Escola de Medicina), cujo tema principal de pesquisa é a compressão vertical e horizontal de dados genômicos.

O principal esforço desta pesquisa é propor um método de compressão de dados para sequências genômicas. Esse método explora, de um lado, os formatos de arquivo de imagem já conhecidos, tais como: *WebP* (GOOGLE, 2018) e *FLIF* (SNEYERS; WUILLE.

2016) e, de outro lado, a abordagem de compressão horizontal sem perda de dados para embasar o método. Apesar de existirem na literatura estudos que tratam do tema de compressão de sequências genômicas com abordagem horizontal e sem perda de dados, ainda há muito o que discutir e investigar, na tentativa de contribuir para alcançar uma solução melhorada para o problema; por exemplo, uma solução que supere a taxa de 75% de economia de espaço de qualquer sequência genômica. Esse percentual de 75% de taxa de economia de espaço é alcançado com a codificação de 2 *bits* por símbolo do alfabeto genômico {A, T, C, G} em que cada símbolo é substituído por dois *bits*, por exemplo: (A=00, T=11, C=01 e G=10), não sendo necessário a aplicação de outra técnica ou ferramenta de compressão especializada. Também, compressores de propósito geral tais como o *ZIP* (PKWARE, 1989) não são efetivos para a compressão de dados de sequenciamento genético, pois não são especializados para lidar com a distribuição dos símbolos do alfabeto genômico que ocorre na sequência genética. Com base nesse cenário e devido aos benefícios que a compressão de dados genômicos traz para a comunidade em termos de economia de espaço, aumentar a taxa de economia de espaço de dados é de grande relevância em termos científicos e tecnológicos. Assim, para validar o método de compressão proposto é apresentada uma análise comparativa entre o método proposto e algumas ferramentas especializadas de compressão de genomas, tais como: *DELIMINATE* (MOHAMMED et al., 2012) e o *MFCOMPRESS* (PINHO; PRATAS, 2014). Isso, deve auxiliar a comunidade na tomada de decisão, quanto ao método que deve ser utilizado para compressão dos dados genômicos.

1.1 Motivação e Hipóteses

Com o advento do *High-Throughput Sequencing* (HTS) (REUTER; SPACEK; SNYDER, 2016), e com as novas tecnologias aplicadas tais como *PacBio* e *Nanopore* e equipamentos de sequenciamento em contínuo desenvolvimento, o volume e a velocidade de geração de dados estão aumentando consideravelmente (Wetterstrand KA, 2016). Durante os próximos dez anos, espera-se que as capacidades de sequenciamento continuem a crescer muito rapidamente (STEPHENS et al., 2015). Se o crescimento continuar no ritmo atual, duplicando a cada 9 meses (KAHN, 2011), então o volume deve chegar a mais de 1 *Exabase* (10^{18} bases nitrogenadas de sequências genéticas) anuais para os próximos 5 anos e após esse período, 1 *Zetabase* (10^{21} bases nitrogenadas) por ano, até 2025 (cf. Figura 1.1). Curiosamente, mesmo as estimativas mais conservadoras fazem uma previsão de dobrar o volume de dados a cada 12 meses (REGALADO, 2014) ou a cada 18 meses, segundo a lei de Moore (STEPHENS et al., 2015; MOORE, 1965). Esses dados, forne-

cem subsídios suficientes para motivar a realização deste trabalho de pesquisa, cujo ponto central é: propor um método de compressão de dados genômicos baseado em formato de imagem, e também abordar técnicas de transformação do alfabeto das sequências de DNA com vistas a produzir melhores percentuais de economia de espaço após a codificação.

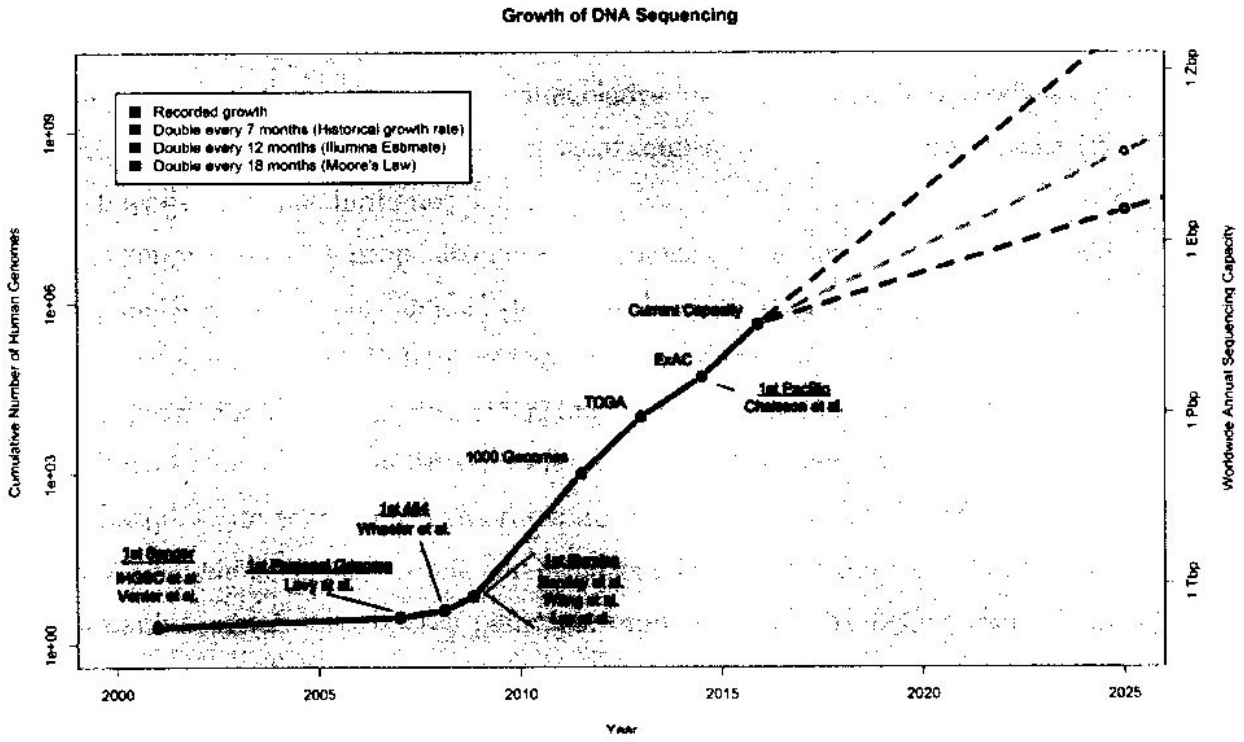

Figura 1.1: Crescimento do sequenciamento de genoma desde o ano 2000 até o ano de 2017, com projeção para 2025.
Fonte: (STEPHENS et al., 2015)

As hipóteses desta pesquisa são baseadas na trajetória de redução do custo do sequenciamento de DNA. Com novas tecnologias surgindo a cada ano (Wetterstrand KA. 2016), possibilitando sequenciamentos de genoma mais rápidos e com menor custo, o volume de dados gerados por tais tecnologias será sem precedentes. Sendo assim, essa pesquisa apresenta três hipóteses:

a) a compressão de dados é uma solução viável para reduzir o espaço de armazenamento das sequências genômicas;

b) a compressão de sequências genômicas baseado em formato de arquivo de imagem é um método viável para mitigar o problema de armazenamento e transmissão de dados genômicos;

c) a transformação do alfabeto de sequências genômicas, aplicada antes da compressão com formato de imagem, melhora a taxa de economia de espaço.

1.2 Justificativa

Dado o grande volume de dados de sequenciamento genético já existente e a trajetória crescente que o sequenciamento está seguindo, é necessário empreender esforços de pesquisa e desenvolvimento para se obter métodos de armazenamento mais eficientes e menos custosos. De acordo com as estimativas de (STEPHENS et al., 2015) sobre o volume total de genomas que serão sequenciados até o ano de 2025, é possível prever que a economia de espaço de apenas 1% desse total representa a quantia média de 3.125.000.000 em termos de genomas humanos.

Compressores de propósito geral, tais como os da família *ZIP* (PKWARE, 1989) não são efetivos para a compressão de dados de sequenciamento genético devido a forma com que os símbolos do alfabeto genômico estão distribuídos. Como resultado disso, as taxas de economia de espaço dessas ferramentas, quando aplicadas sobre sequências genômicas, ficam abaixo do resultado mínimo esperado. Uma sequência de DNA representada em um arquivo de texto com caracteres do formato *American Standard Code for Information Interchange* (ASCII) pode ser armazenada também em um arquivo com 25% do seu tamanho inicial. Em outras palavras, uma sequência de DNA codificada em ASCII é realmente comprimida se, e somente se, a taxa de economia de espaço for superior a 75%. Isso demonstra que para um alfabeto que considera apenas os símbolos do alfabeto genômico {A, T, C, G}, a simples transformação da representação de cada base nitrogenada, passando de 8 *bits* para 2 *bits*, codificação *Naïve-bit encoding*, já produz uma taxa de economia de espaço de 75%.

1.3 Objetivos

O objetivo principal é conceber um método computacional baseado em formato de arquivo de imagem para reduzir o espaço de armazenamento das sequências genômicas. A complexidade em termos de processamento não é o objeto central dessa pesquisa. Para que o objetivo principal seja alcançado, um conjunto de objetivos específicos foram estabelecidos, a fim de proporcionar uma melhor compreensão deste novo método de compressão. Esses objetivos consistem em:

a) avaliar a taxa de economia de espaço de sequências genômicas utilizando formatos de arquivo de imagem sem perda de dados como formatos viáveis para mitigar o problema de armazenamento e transmissão de dados genômicos:

b) selecionar, da literatura especializada, pelo menos dois métodos de compressão hori-

zontal sem perda para sequências genômicas para serem utilizados como ferramentas *baseline* para comparação dos resultados;

c) experimentar diferentes técnicas de transformações do alfabeto genético como pré-processamento para reduzir o espaço de representação e melhorar a taxa de economia de espaço;

d) utilizar um conjunto de sequências genômicas de teste proposto na literatura especializada e que represente um banco de dados genéticos, tal como o NCBI, para avaliação experimental da taxa de economia de espaço; e

e) relatar os resultados obtidos nos experimentos de avaliação da taxa de economia de espaço, envolvendo métodos da literatura especializada, formatos de arquivo de imagem digitais e formatos de arquivo de imagem digitais com pré-processamento, para tentar reduzir o espaço de representação.

1.4 Limitações e escopo

O escopo deste trabalho está limitado a realizar apenas a transformação dos dados das sequências genômicas bem como a compressão dos dados baseado em formato de arquivo de imagem. A descompressão não é alvo dessa pesquisa, pois os arquivos gerados pelo método de compressão proposto permitem que se façam buscas diretamente nas imagens com algoritmos existentes na literatura, sem a necessidade de descomprimir para o arquivo FASTA ou multi-FASTA original. Assim, não é avaliada a complexidade de tempo da compressão, pois espera-se que o custo seja apenas para comprimir os dados uma só vez. Também não são utilizados arquivos multi-FASTA nos testes de compressão com formato de arquivo de imagem, pois são apenas uma concatenação de FASTA "simples" em um mesmo arquivo. Por fim, é utilizada nessa pesquisa sequências genômicas que possuem 268.435.456 (2^{28}) ou menos bases nitrogenadas. Isso se deve a uma limitação do formato de arquivo de imagem *WebP* discutida nesse documento em seção apropriada.

1.5 Organização do documento

A organização deste trabalho será da seguinte forma: no Capítulo 2 é apresentada a fundamentação teórica sobre compressão de dados genéticos. No Capítulo 3 é abordada a revisão bibliográfica da literatura. No Capítulo 4 é apresentada o método de compressão com formato de arquivo de imagem. No Capítulo 5 é apresentado de que forma foi

realizado o desenvolvimento das análises e testes, e finalmente no Capítulo 6 é apresentado e discutido os resultados obtidos. Assim, está disposta a organização desse trabalho.

Capítulo 2

Fundamentação teórica

Antes de ser abordada, nesta pesquisa, a compressão de sequências genômicas em si, é necessário entender a origem destas sequências, desde a sua versão biológica existente na célula de todo organismo vivo (cf. Figura 2.1) bem como alguns vírus, e o caminho que ela percorre até chegar na sua versão digital. Tal versão é a informação da sequência genômica propriamente dita, que a partir da molécula de DNA, é fragmentada, sequenciada, lida e alinhada, resultando em vários formatos digitais, sendo um deles o formato *FASTA*, que permite interpretação humana e computacional. Essa interpretação é utilizada pelas diferentes áreas da medicina, biotecnologia e farmácia, sendo alguns dos principais objetivos o diagnóstico e tratamento de doenças a nível genético e molecular. A partir destas considerações, será apresentado o processo de compressão de dados com ênfase em compressão de sequências genômicas.

2.1 Obtenção dos dados genéticos

O DNA, em português *Ácido desoxirribonucleico* (ADN), é um polímero existente na célula de todo organismo vivo. A molécula é responsável por armazenar e transmitir de uma geração à outra as informações genéticas utilizadas no crescimento, desenvolvimento e reprodução da célula. Essa molécula é encontrada em todos os organismos vivos conhecidos e também em alguns vírus. Sua estrutura molecular é constituída de duas longas cadeias, não ramificadas, de nucleotídeos dispostos de forma complementar (cf. Figura 2.1). Tais nucleotídeos, singularmente chamados de base nitrogenada, são especificamente de quatro tipos, sendo eles: Adenina, Citosina, Guanina e Timina (B. Alberts, A. Johnson, J. Lewis, M. Raff, K. Roberts and P. Walter., 2002). Em 1869, o médico Friedrich Miescher descobriu uma substância na secreção de um processo infeccioso presente em uma bandagem cirúrgica descartada. Essa substância, encontrada no núcleo de

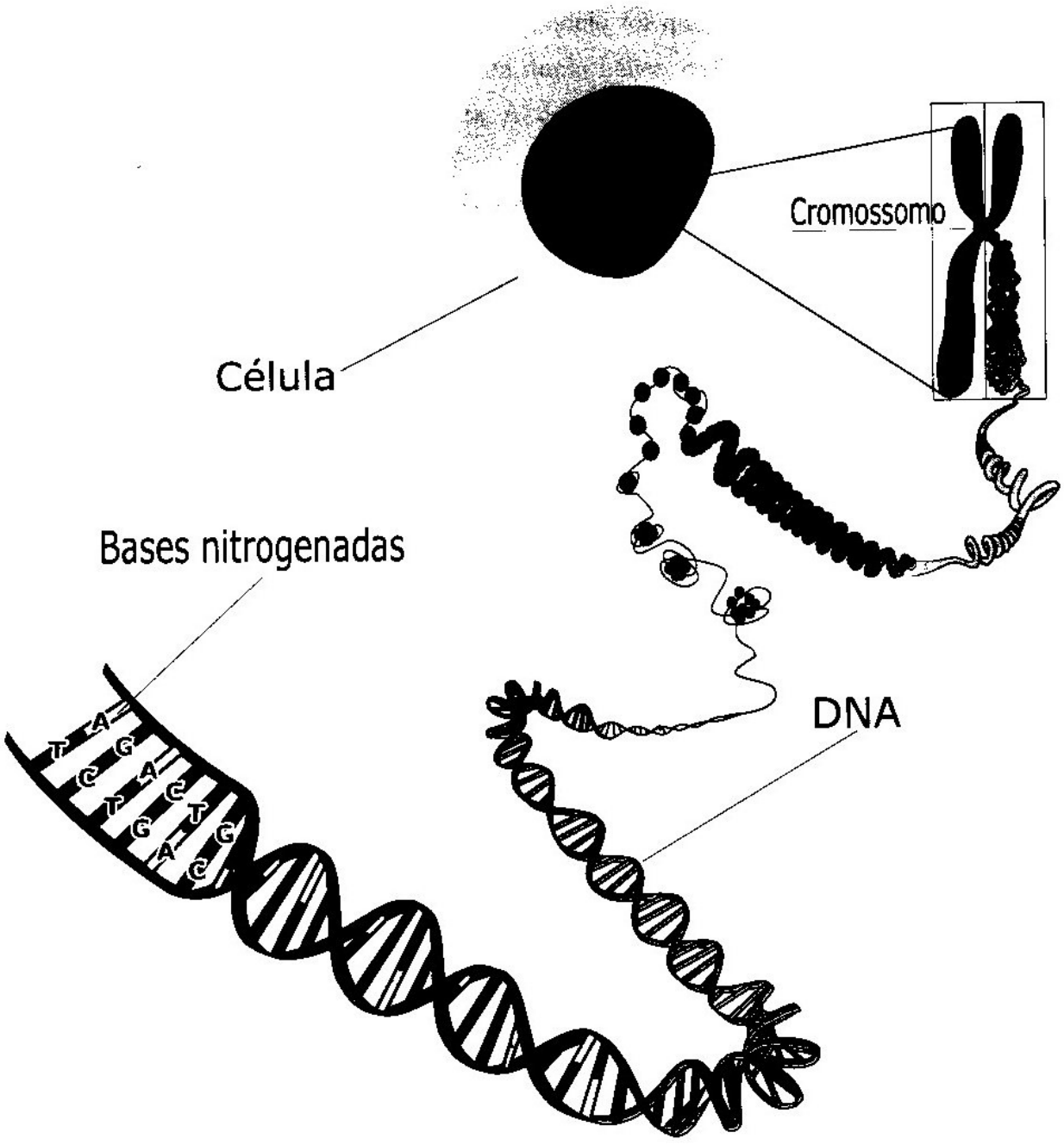

Figura 2.1: Ilustração de uma molécula de DNA encapsulada em um núcleo celular.
Fonte: Pixabay (2018)

uma célula, foi chamada de "nuclein" pelo médico. Com esse episódio, deu-se o primeiro isolamento na história de pesquisa do DNA. No início da década de 1950. por meio de difração de raios-x, análises de amostras de DNA sugeriram que a molécula em questão é um polímero helicoidal composto por duas vertentes. Essa constatação. de que o DNA tem formato de dupla hélice, foi crucial, pois forneceu a pista que. em 1953. levou à construção de um modelo estrutural. O modelo proposto se ajustava ao padrão observado na difração de raios-x, e assim, encontrou-se um caminho para tentar resolver o enigma da estrutura

do DNA . Uma característica essencial do modelo foi que todas as bases nitrogenadas da molécula de DNA residem no interior da dupla hélice, e do lado externo da estrutura, residem os fosfatos de açúcar (B. Alberts, A. Johnson, J. Lewis, M. Raff, K. Roberts and P. Walter., 2002).

O sequenciamento genético é o processo que define de forma precisa a ordem das bases nitrogenadas (Adenina, Citosina, Guanina e Timina) no interior de uma molécula de DNA. O conhecimento dessa ordem, tornou-se indispensável para pesquisa básica em biologia e seus diferentes campos de aplicações, como: diagnóstico médico, biotecnologia, biologia forense e virologia. O primeiro método de sequenciamento de molécula de DNA chamado "synthetic location-specific primer strategy", foi proposto em 1970 pelo biologista Ray Wu da Universidade de Cornell localizada na cidade de Ithaca, no estado de Nova York (PADMANABHAN; PADMANABHAN; WU, 1972). Em 1977 o método de Ray foi adotado pelo bioquímico Frederick Sanger do Centro de Pesquisas Médicas Council em Cambridge no Reino Unido, a fim de desenvolver métodos mais rápidos de sequenciamento (SANGER; NICKLEN; COULSON, 1977).

O primeiro sequenciamento completo de um genoma aconteceu em 1977 no qual o DNA do vírus "phi x 174" foi sequenciado (SANGER; NICKLEN; COULSON, 1977). Vários novos métodos de sequenciamento de DNA foram desenvolvidos no final dos anos 1990 e foram implementadas em sequenciadores de DNA profissionais até ao ano 2000. Estes são chamados de métodos de Próxima Geração *Next-generation sequencing* (NGS) ou HTS.

O sequenciamento do genoma humano foi concluído como "rascunho" em 2001 (LANDER et al., 2001). Pouco depois, as sequências do genoma de vários organismos modelos foram determinadas (MIKKELSEN et al., 2005). A primeira sequência do genoma humano custou entre 0,5 e 1 bilhão de dólares. Inicialmente, a limitação do custo reduziu o potencial de sequenciamento de DNA para outras aplicações, como o sequenciamento de genoma pessoal. Após a publicação do sequenciamento finalizado do genoma humano (COLLINS et al., 2004) o *National Human Genome Research Institute* (NHGRI) criou uma tecnologia de sequenciamento de DNA com investimento de 70 milhões de dólares para que o custo do sequenciamento do genoma humano fosse reduzido para 1.000 dólares em 10 anos (SCHLOSS, 2008), assim, surgiu uma nova geração de tecnologias de sequenciamento de alto rendimento HTS.

O sequenciamento de alto rendimento HTS tornou prático os projetos de sequenciamento de genoma em larga escala. Alguns projetos de HTS fazem sequenciamento de genomas de espécies para as quais uma sequência ainda não existe, enquanto outros são projetos que fazem sequenciamento de muitos genomas individuais da mesma espécie para

melhor compreender as variações presentes nos genomas. Vários projetos de sequenciamento, que surgiram com o advento do HTS, hospedam milhares de sequências genômicas em suas bases de dados. Somente o *1000 Genomes Project* (AUTON et al., 2015), iniciado em 2007, produziu mais de 50 *terabytes* de dados com o sequenciamento do genoma de 1092 indivíduos de 14 populações distintas, com o objetivo final de sequenciar 2500 indivíduos de 27 populações ao redor do mundo (ALTSHULER et al., 2012). Cada genoma humano sequenciado pelo *1000 Genomes Project* tem duas cópias complementares de 3,2 *gigabases*.

Desde que o projeto do genoma humano foi iniciado em 2001, com o custo de sequenciamento estimado em uma média de 100 milhões de dólares, a cada ano, foi possível observar um declive significativo do custo (cf. Figura 2.2), totalizando mais de 98% de queda ao final do ano de 2017 (National Human Genome Research Institute, 2018b). Naquele ano, mais precisamente no mês de julho, o custo do sequenciamento genético estava em torno de US 1.121,00 (National Human Genome Research Institute, 2018a).

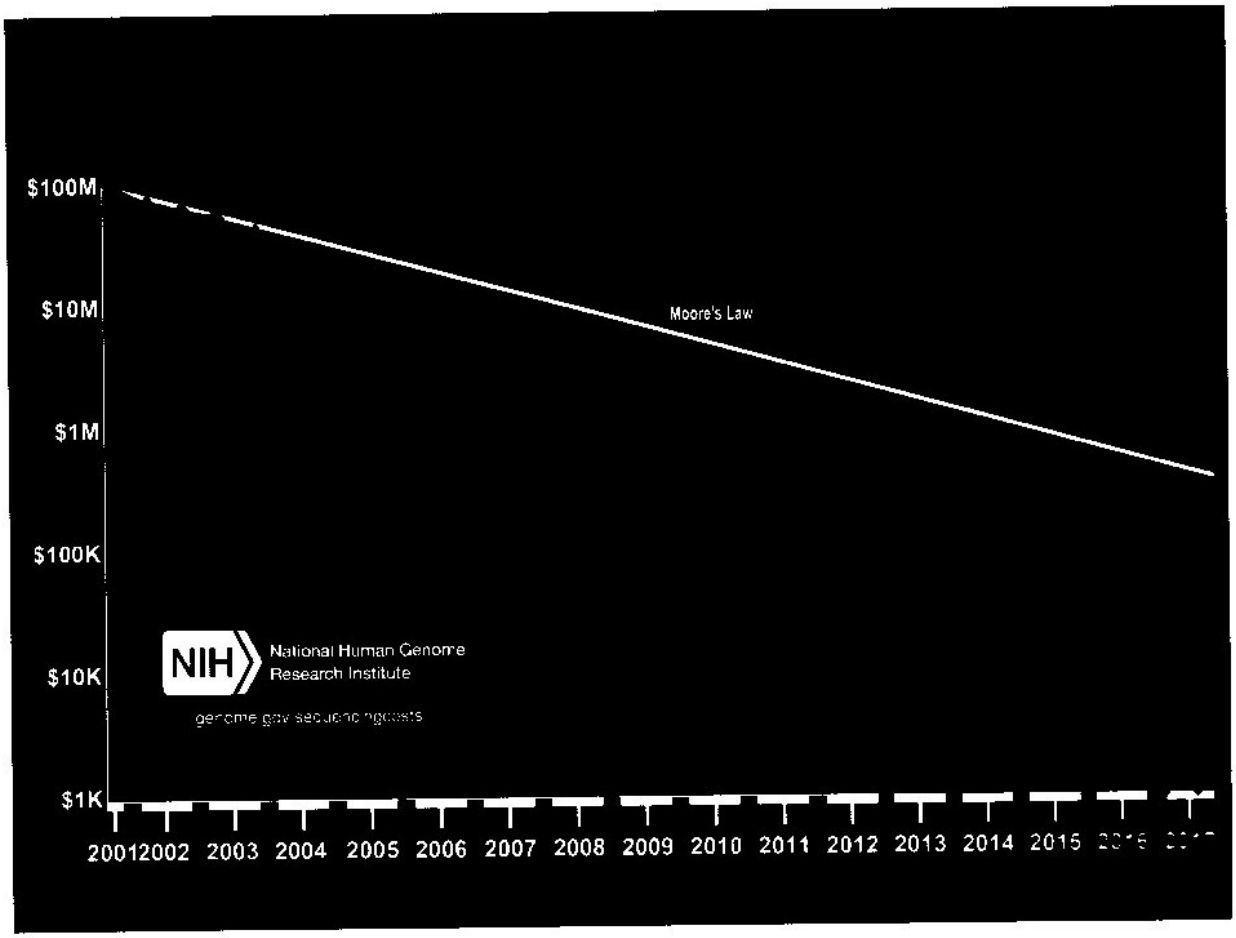

Figura 2.2: Custo do sequenciamento por genoma.
Fonte: NIH (2017)

Com o processo de sequenciamento de genomas cada vez mais rápido e com custo menor, testes de diagnóstico baseados em genoma foram desenvolvidos (OLSON et al., 2013). Tais testes genéticos têm o potencial de prever o risco e conduzir intervenções terapêuticas antecipadas, com o objetivo de detectar o início de uma doença, ou detectar uma doença residual. Isso tornou possível o tratamento personalizado, contribuindo para a descoberta de ligações entre variantes genéticas específicas e doenças. Embora ainda existam muitos esforços de pesquisa para descobrir e colocar em prática todo potencial genético, a melhoria da saúde é um dos principais objetivos da pesquisa genômica, segundo OLSON et al..

O conhecimento completo das funções de todos os genes do ser humano pode mudar drasticamente os processos de desenvolvimento de descobertas de fármacos e também a pesquisa de drogas como um todo (SIEST; MARTEAU; VISVIKIS-SIEST, 2009). A aplicação de tecnologias genômicas no desenvolvimento clínico de drogas, novas e existentes, é conhecida como farmacogenômica. Graças ao desenvolvimento recente de pesquisa em medicina genômica clínica e farmacogenômica, as doenças poderão ser tratadas, em um futuro próximo, a nível genético por marcadores genéticos individuais específicos, de forma que os medicamentos e suas posologias possam ser otimizados de acordo com o perfil genético dos pacientes (MANOLIO et al., 2013).

O *National Center for Biotechnology Information* (NCBI), que hospeda dados provenientes do sequenciamento de genomas, até 16 de fevereiro de 2018 detinha 34847 sequências hospedadas, sendo estas: Eucariontes (5.245); Procariontes (131.198); Vírus (14.026); Plasmídeos (11.535); Organelas (11.370) (NCBI, 2018). A *Sequence Read Archive* (SRA), uma divisão do NCBI, que armazena dados de sequências brutas de tecnologias de sequenciamento de NGS tais como *Illumina*, *Roche 454*, *IonTorrent*, *Complete Genomics*, *PacBio* e *Nanopore* espera exceder 10.000 terabytes até o final de 2018 (NIH, 2018). Isso reforça a informação de que o volume de dados de sequenciamento deve dobrar a cada nove meses, suplantando as melhorias de desempenho de computação e armazenamento (KAHN, 2011). Assim, métodos de compressão de dados para reduzir o espaço de armazenamento e economizar a largura de banda de transferência de dados tornaram-se cruciais para o gerenciamento eficiente de dados genômicos.

2.2 Compressão de dados

> *"Data compression is the process of converting an input data stream (the source stream or the original raw data) into another data stream (the output, the bitstream, or the compressed stream) that has a*

smaller size. A stream is either a file or a buffer in memory."

— D. Salomon (SALOMON, 2007)

Compressão de dados, no seu sentido mais amplo, é a aplicação de métodos de transformação do alfabeto alvo e de codificação para minimizar a quantidade de dados a serem armazenados, obtidos ou mesmo transmitidos. A compressão de dados pode ser aplicada a vários formatos de dados, como texto, imagens e sinais. De forma geral, faz-se uso da compressão de dados para reduzir custos e aumentar a eficiência na manutenção de grandes volumes de dados (SALOMON, 2007).

A compressão de dados é essencial para a computação moderna, por duas razões: i) primeiramente porque as pessoas acumulam dados e não importa o quão um dispositivo de armazenamento tem de capacidade, mais cedo ou mais tarde essa capacidade vai esgotar. É com essa ótica que a compressão de dados genéticos é percebida como alternativa útil, a fim de retardar tal inevitabilidade; ii) dado que tempo e espaço são custos importantes, logo, quanto maior forem os arquivos para transferir ou armazenar, mais elevados serão os custos.

Há vários métodos conhecidos para a compressão de dados (SALOMON, 2007). Tais métodos são baseados em diferentes abordagens e adequados a diferentes tipos de dados, bem como produzem resultados específicos. Independente disso, eles são baseados no mesmo princípio, comprimir os dados removendo as redundâncias do arquivo de origem. Quaisquer dados não randômicos possuem alguma estrutura e essa estrutura pode ser explorada para alcançar uma representação menor dos dados, uma representação em que nenhuma estrutura é discernível.

Alguns métodos de compressão que implementam *lossy compression* permitem a perda controlada de dados. Estes métodos alcançam melhores taxas de economia de espaço com tal perda de dados. Quando os dados são comprimidos e descomprimidos, o resultado final não é idêntico aos dados originais, pois houve perda, permitida, de parte dos dados (SALOMON, 2007). A utilização desses métodos faz sentido, especialmente, para comprimir dados armazenados de forma analógica, como: imagens, filmes ou sons. Assim, Quando a perda de dados não causa impacto percebido no resultado final da compressão, a diferença pode não ser percebida pelo ser humano. Um exemplo desse tipo de compressão com perda de dados é a conversão do formato de áudio *Waveform Audio File Format* (WAVE) para *MPEG-1 Audio Layer III* (MP3) (FLEISCHMAN, 1998; NILSSON, 2000).

Em contraste com os métodos de compressão com perda, estão os métodos de compressão que implementam *lossless compression*. Estes métodos geralmente exploram a

redundância dos dados a serem comprimidos, mantendo somente uma única representação dos mesmos. Assim, eles preservam os dados de modo que, após descompressão, o arquivo é restaurado para o seu estado antes da compressão (SALOMON, 2007).

Todo algoritmo de compressão de dados explora uma dentre duas ações, a saber, transformar ou codificar os dados. Essas ações podem ser chamadas de fases da compressão de dados. Diferentes tipos de dados requerem distintos tipos de tratamento para a compressão de dados. Um exemplo prático é realização da compressão de um conjunto de palavras de um texto qualquer, escrito em português e também a compressão de uma lista de valores numéricos com ponto flutuante, utilizando o mesmo algoritmo para ambos. Para cada uma das situações, o algoritmo responderá com maior ou menor eficiência. Isso mostra que, alguns tipos de dados, podem ser transformados antes de serem codificados, com o objetivo de obterem melhores taxas de economia de espaço (Colt Mcanlis, 2016).

O processo de comprimir dados passa por, pelo menos, uma de duas fases distintas (SALOMON, 2007) da compressão. Os dados podem ser, primeiramente, modificados na fase de transformação e finalmente codificados na fase de codificação.

Fase de transformação dos dados: Essa fase é também chamada de fase de processamento inicial ou de mapeamento de dados. De acordo com o matemático *Claude Elwood Shannon*, autor do artigo científico *A Mathematical Theory of Communication*, a entropia da informação, dada pela Equação 2.1, coloca um limite em quão pequeno se pode tornar um conjunto de dados (SHANNON, 1948). A chave para romper o limite da entropia é explorar a organização estrutural do conjunto de dados para transformar os dados em uma nova representação, e que tal representação gere uma entropia mais baixa do que a informação de origem. A compressão de dados trabalha para tentar vencer este limite, explorando duas propriedades sobre os dados: ordenação e relações entre símbolos. Sendo assim, a fase de transformação consiste basicamente em reduzir o número de símbolos únicos contidos no conjunto dos dados a ser comprimido. Em outras palavras, isso significa poder representar o conjunto dos dados com um alfabeto que utiliza a menor quantidade possível de caracteres únicos. É na fase de transformação que são eliminados dados ou palavras que podem ser omitidos sem a perda de informação (quando se trata de compressão sem perda). Esses dados são geralmente repetições ou informações supérfluas (Colt Mcanlis, 2016).

$$H = -s \sum_{1}^{n} p_i \log_2 p_i \tag{2.1}$$

Fase de codificação dos dados: Nesta fase, cada símbolo do novo alfabeto, proveniente da fase de transformação, é codificado usando um determinado número de

bits. Em geral, os algoritmos aplicados nessa fase operam com codificação estatística ou aritmética. A codificação de *Huffman* é provavelmente a maneira mais direta e mais conhecida de codificação estatística. Tal codificação usa uma árvore binária para explorar as probabilidades de cada símbolo ocorrer no conjunto de dados e assim, atribuir uma quantidade menor de bits para os símbolos mais frequentes. (COLT MCANLIS, 2016). Na codificação aritmética, ao invés de atribuir uma quantidade menor de bits para os símbolos mais frequentes, transforma todo o fluxo de entrada de um conjunto de símbolos para um valor numérico (excessivamente longo), cuja representação $\log_2$ está mais próxima do valor real da entropia para o fluxo (COLT MCANLIS, 2016).

Há estudos realizados sobre o tema de compressão de sequências genômicas que consideram algumas características peculiares dessas sequências. Tais características são: alfabeto reduzido (próprio do domínio), frequência de trechos repetidos e frequência de palíndromos. Outra característica que pode ser considerada é que a sequência genômica possui dupla fita, uma complementar a outra (cf. Figura 2.1). Uma distinção importante que vale a pena salientar aqui é a diferença entre compressão de genoma completo e compressão de dados de sequência genética. A compressão de sequências genéticas centra-se esforços na codificação da saída das máquinas de sequenciamento de nova geração NGS, que são capazes de ler grandes quantidades de bases nitrogenadas de uma amostra biológica (de 35 até mais de 1000 fragmentos). Esses fragmentos lidos são comumente chamados de *short reads*.

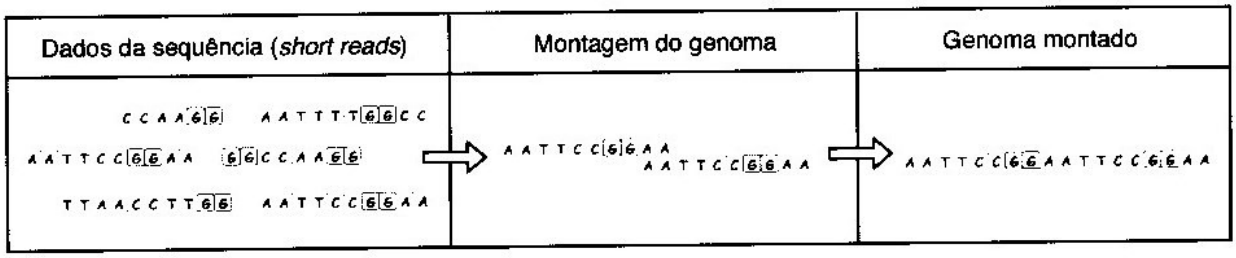

Figura 2.3: Montagem de genoma. De *short reads* até genoma completo.
Fonte: Autor.

Ferramentas de compressão de genoma visam a codificação da informação genética de um organismo vivo, expressa como uma sequência de símbolos do alfabeto genômico {A, T, C, G}, que representam as bases nitrogenadas. Essa sequência tem aproximadamente no caso do genoma humano de 2 a 3 bilhões de bases nitrogenadas e estão organizadas em 23 cromossomos. Para alguns organismos (e.g., *Plant*), essa quantidade pode chegar até 100 bilhões de bases nitrogenadas. A codificação de um genoma "completo" é o resultado de um longo processo de análise que atualmente, por um problema de limitação

de hardware, só pode fornecer uma aproximação da sequência genética real.

A seguir são apresentadas duas estratégias de compressão de dados. A primeira estratégia baseia-se nas características mencionadas que utilizam a própria sequência como fonte de informações. Essa estratégia é chamada de compressão horizontal. A segunda estratégia explora a similaridade compartilhada entre uma sequência alvo e uma sequência de referência. Esse compartilhamento de informações é utilizado na estratégia de compressão vertical e é significativo dado que as sequências genômicas de organismos da mesma espécie são 99,5% similares (LEVY et al., 2007). Nessa última estratégia, a sequência alvo é alinhada com a sequência de referência e as diferenças entre elas é que são codificadas e armazenadas. O primeiro trabalho a utilizar os nomes de "compressão horizontal" e "compressão vertical" como estratégias de compressão de sequências genômicas foi o *BioCompress* (GRUMBACH; TAHI, 1993).

O conceito de compressão vertical possui várias abordagens que podem ser exploradas em compressão de sequências genômicas. Uma dessas abordagens é a "compressão relativa" (cf. Figura 2.4), que faz uso de um conjunto de sequências genômicas de organismos da mesma espécie para realizar a compressão, selecionando cada uma dessas sequências e comparando-a com todas as outras do mesmo conjunto (GRUMBACH; TAHI, 1993). Para os pares de sequências que estão intimamente relacionados de forma filogenética, a taxa de economia de espaço é melhor do que para os pares de sequência distantemente relacionados. Outra abordagem, explora o fato de cada sequência da coleção ser quase idêntica a sequência de referência, com exceção de algumas variações e armazenar apenas as diferenças entre elas.

A abordagem vertical é significativamente vantajosa na compressão de organismos da mesma espécie. Essa vantagem é baseada na alta similaridade entre os genomas desses organismos. Por exemplo, a similaridade entre os genomas dos seres humanos é de 99,5% (LEVY et al., 2007), e, em teoria, requer-se armazenar apenas os 0,5% das diferenças existentes entre tais genomas (GIANCARLO; SCATURRO; UTRO, 2009). No entanto, para essa abordagem, as sequências usadas como espaço de busca para a compressão devem ser acessíveis para o processo de descompressão.

A estratégia de compressão horizontal, também chamada *"Reference-free Methods"* implementa algoritmos para explorar propriedades estruturais da sequência genômica, por exemplo, palíndromos, repetições, bem como propriedades estatísticas. Assim, na fase transformação dos dados, tal estratégia explora as características da própria sequência alvo. Dessa forma, a compressão horizontal não depende de uma referência de informação de outra sequência para ser comprimida ou descomprimida, i.e., cada sequência genômica é comprimida usando a informação contida na própria sequência (GRUMBACH; TAHI,

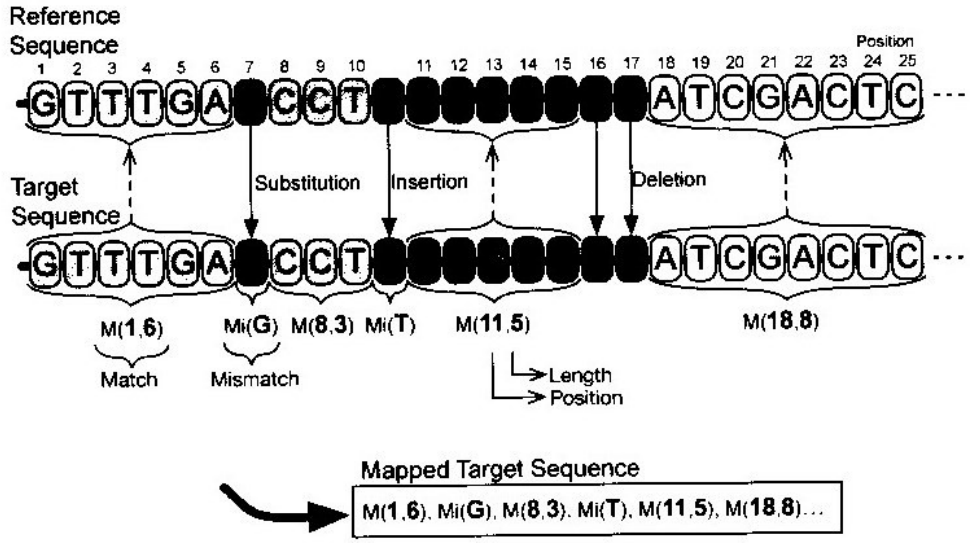

Figura 2.4: Exemplo de transformação dos dados utilizando a estratégia de compressão vertical. A sequência alvo é comparada com a sequência de referência e reescrita apenas com o mapeamento das diferenças.
Fonte: Kredens et al., 2018 (não publicado)

1993). Portanto, isso facilita o seu uso diário. dado que o processo de descompressão não depende de uma referência externa.

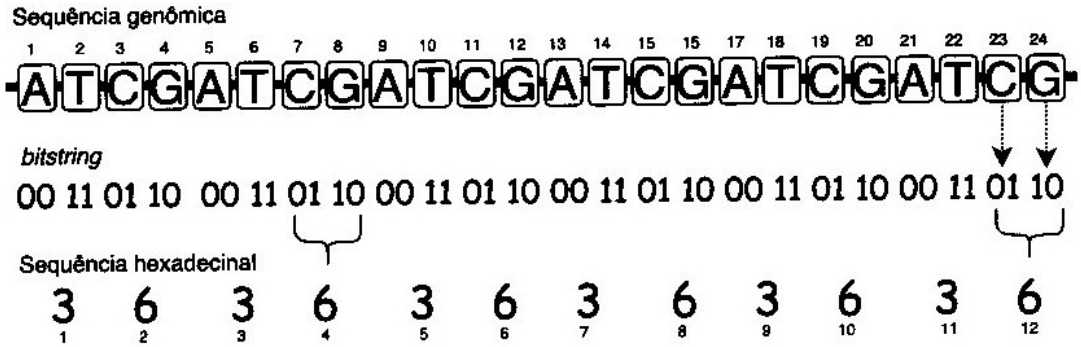

Figura 2.5: Fase de transformação dos dados que utiliza a estratégia de compressão horizontal. No exemplo, são utilizadas informações contidas na própria sequência para gerar uma nova representação do alfabeto genômico {A, T, C, G}.
Fonte: Autor.

Vale destacar que as vantagens e desvantagens em utilizar a abordagem vertical ou horizontal, no momento de compressão, estão ligadas ao o uso diário dos arquivos. Quando se tem uma coleção de genomas comprimidos, com a abordagem vertical. o custo para avaliar apenas uma sequência da coleção envolve manipular dados de todas elas. No caso da compressão com abordagem horizontal, se fizer necessário descomprimir. o custo será para manipular dados de apenas uma sequência.

2.3 Considerações finais

Neste capítulo foi apresentado a fundamentação teórica sobre a origem dos dados genômicos desde a descoberta da primeira molécula de DNA e o caminho percorrido do formato biológico ao formato digital. Foi abordado também alguns projetos de sequenciamento genético e as várias tecnologias envolvidas, bem como, mencionado o custo e volume do sequenciamento. A medicina personalizada, viável, está diretamente ligada ao custo e é uma realidade cada vez mais presente nos estudos e diagnósticos, além de fomentar a descoberta de drogas e novos tratamentos a nível genético e molecular. Por isso, o assunto compressão de dados genômicos precisa de atenção especial, principalmente em como armazenar os dados de forma mais eficiente. É a partir daqui que a compressão de dados genômicos e sem perda, baseado em formato de imagem. entra em cena como solução alternativa para ajudar a mitigar o problema.

Capítulo 3

Revisão bibliográfica

Este capítulo apresenta o estado da arte sobre compressão de sequências genômicas representadas em arquivo do formato *FASTA* (que permite interpretação humana e computacional) com abordagem horizontal e sem perda de dados. Neste contexto, são apresentadas algumas técnicas existentes na literatura para a transformação de alfabeto de sequências genômicas. Tais técnicas podem ser aplicadas nas sequências genômicas armazenadas em arquivos no formato *FASTA*. Cada técnica pode atender um ou mais objetivos, a saber: diminuir a entropia da informação, diminuir o tamanho da sequência a ser comprimida, rearranjar, agrupar ou ordenar os dados de uma dada sequência. Apresenta-se também algumas ferramentas especializadas em compressão de sequências genômicas bem como ferramentas de compressão de propósito geral. Por fim, examina-se os formatos de imagens utilizados nos testes de desempenho para a validação da proposta. Para o entendimento dos fundamentos de compressão de dados, aqui são sugeridos dois livros ao leitor: (SALOMON, 2007) e (Colt Mcanlis, 2016).

3.1 Formato de arquivo FASTA e multi-FASTA

A estrutura do arquivo no formato *FASTA* é composta de duas partes. A primeira parte, posicionada na primeira linha do arquivo (cf. Figura 3.1), é o cabeçalho do arquivo. O início do cabeçalho é marcado pelo caractere ">" (maior que) ou pelo caractere ";" (ponto e vírgula), sendo que este último caractere é menos utilizado. É no cabeçalho que se encontra a descrição da sequência genômica. A segunda parte deste formato de arquivo contém a sequência de DNA ou proteína propriamente dita, e é iniciada na primeira coluna da segunda linha, não possuindo marcação alguma. No caso de arquivo multi-FASTA essa estrutura com o par de "cabeçalho e sequência" repete-se ao longo do arquivo, pois são apenas uma concatenação de dois ou mais arquivos FASTA "simples". A sequência é

representada por uma serie de linhas que podem conter 70 ou 80 colunas, dependendo do equipamento de sequenciamento genético que escreveu o arquivo. É esperado que o alfabeto utilizado nas sequências genômicas contidas no formato *FASTA* (cf. Tabela 3.1), seja representado com o código padrão de aminoácidos e ácidos nucleicos conforme indica a autoridade mundial em nomenclaturas e terminologias químicas *International Union of Pure and Applied Chemistry* (IUPAC), com as seguintes exceções (IUBMB-IUPAC, 1984):

a) Letras minúsculas são aceitas, mas são convertidas em maiúsculas;

b) Um único hífen ou traço pode ser usado para representar uma lacuna de comprimento indeterminado;

c) Em sequências de aminoácidos, os caracteres "U" e "*" são aceitáveis; e

d) Quaisquer dígitos numéricos na sequência devem ser removidos ou substituídos por códigos de letra apropriada (e.g., "N" para ácido nucleico desconhecido e "X" para aminoácido desconhecido)

```
>gi|411145200|gb|JY157487.1| REI001N02.F REI Chlamydomonas reinhardtii
GGATCTTGAGCGGCAAGCGGGGCGCAAAGGGCGGGGACGGAGCGCGCAAGGATGCGGAGTTCGGCGTGAC
GCGCGGCATTGACTTCAAGGGCGTGCGCACCGTGATCAACTACGACCCGCCCTCCAGCCTGCAGGGCTAC
GTGCACCGAGTCGGCCGCACGGGTGCGGGCGGGCTGGTGTGGGGCGGACTGCGGAGTGCGGACGGCAAGG
GACGGGGCGGGGTGCACAGTAGGGCGGGCGGAGTGTGTGGTGAGACGGGACGGATACCGACAAGTTGT
CAAACGGCAGTAGGAGGCCTCACAGCATCCTTTTGACCTTTCGAACCCCTCGCACCCCTCTTCCACAGGC
CGCGTCTGTGAGTCGGGTACTGCCATCTCGCTGTTCACGCCGCAGCTCTCGGAGCTCAAGTTCCGCATGC
ATCTGGAGGAGGCGTTTGAGGCCGTGTCGGATGCTCGTATTCGCCTAGTCTTCACTGTCGTGGGCGTGCT
CGTGCTCTGCTTCCTAGTCGAGGTCGATGACTATTACTACTACGTCGTTTTGGTGGCCGCCGTTAGTTTT
TATCTTGTTGATTCTTGTCGTTGGCTTCTTTGTTTGTTGGGTTGTTTATTGTTCTTTCTGTTTTGTTTCT
CTCGGTCTGCTTTATTCAGTTTAGGTCTGAGTCTTCTGTCGTTCCTCTTGGGGTAGTATTTTTTCCCCTT
GCCTTTTATTTATAGTGGTTCATCCTGTGGATGTCCTCGATCCTATGTTATATTCTATATCTCTTTCGTT
TTACTTGTACTTTTTGGTGTGTTTTCCATTGATTCTCTTGTGACTCGCTTCCCTTGTTTTTTCTTCTCGC
CTCTCTTTGCTTTTTTCTTTTGCCTTTCACCCTCTTCTTTCTTTATTGTTTCTCTTTGTGTTAATTCGTC
TTTCTGCTCCGTTCCTATGTTTCGGTCGCTCTTGTCTTTTTGTTTTTTCTGATCTTTTTCACTTTTATCA
CCTTTTCTTGCTTTTAATTG
```

Figura 3.1: Recorte de imagem de um arquivo FASTA "simples".
Fonte: NCBI (2018).

A Figura 3.1 apresenta uma porção do DNA do organismo da espécie *Chlamydomonas reinhardtii* (Alga Verde) com identificador alfanumérico no *GenBank*: JY157487. Ainda na mesma figura as bases nitrogenadas, cujo comprimento total da sequência é de 1000 bases, estão dispostos em colunas de 70 símbolos.

Conforme Tabela 3.1, a coluna "Símbolo" apresenta os símbolos que representam cada conjunto de bases nitrogenadas descritos na coluna "Bases nitrogenadas". A coluna

Tabela 3.1: Caracteres que representam os ácidos nucleicos. Notação IUPAC.
Fonte: Wikipedia (2018).

Símbolo	Descrição	Bases nitrogenadas				Símbolo	Descrição	Bases nitrogenadas			
A	**A**denina	**A**				**M**	A**M**ina	**A**	**C**		
C	**C**itosina		**C**			**S**	Forte (**S**trong)		**C**	**G**	
G	**G**uanina			**G**		**W**	Fraco (**W**eak)	**A**			**T**
T	**T**imina				**T**	**B**	Não A (**B** após A)		**C**	**G**	**T**
U	**U**racila				**U**	**D**	Não C (**D** após C	**A**		**G**	**T**
R	Pu**R**ina	**A**		**G**		**H**	Não G (**H** após G)	**A**	**C**		**T**
Y	Pirimidina (P**Y**rimidine)		**C**		**T**	**V**	Não T (**V** após T)	**A**	**C**	**G**	
K	Cetona (**K**eto)			**G**	**T**	**N**	Qualquer Nucleotídeo	**A**	**C**	**G**	**T**

"Descrição" informa os nomes das bases nitrogenadas, e em alguns casos, ao lado de cada nome, foi adicionado o nome no idioma inglês, destacando em negrito e maiúsculo a letra que deu origem ao seu respectivo símbolo.

3.2 Ferramentas especializadas para compressão de genomas

A ideia básica da compressão de sequência genômica com abordagem horizontal, também chamada de compressão livre de referência, é explorar propriedades estruturais, como por exemplo, palíndromos, bem como propriedades estatísticas das sequências (GIANCARLO; SCATURRO; UTRO, 2012). O primeiro algoritmo que foi proposto especificamente para compressão de sequências genômicas é o *Biocompress* (GRUMBACH; TAHI, 1993). Ele é baseado no método de compressão *LZ77* (ZIV; LEMPEL, 1977), onde as repetições e palíndromos são detectados na sequência alvo e, em seguida, são codificados usando o comprimento e a posição inicial de cada ocorrência. O algoritmo *Biocompress-2*, uma extensão do *Biocompress*, explora a mesma metodologia, bem como o uso da codificação aritmética de ordem-2 (agrupamento de 2 símbolos) quando não consegue encontrar uma repetição significativa.

O algoritmo *Cfact* (RIVALS; DELAHAYE, 1996) consiste em duas fases, a saber, a fase de análise e a fase de codificação. Na fase de análise, são selecionadas as repetições mais significativas, isto é, as repetições que podem ser codificadas para obter o melhor ganho de economia de espaço de memória. Para selecionar tais repetições, cria-se uma estrutura de dados em árvore de sufixo (TREES, 1995) para encontrar as repetições mais longas. Na fase de codificação, todas as regiões da sequência que não são repetições e também as primeiras ocorrências das repetições são codificadas usando dois bits para cada símbolo. Além disso, as próximas ocorrências de repetições são codificadas usando

ponteiros para suas primeiras ocorrências, no formato de dupla (posição, comprimento).

O uso de "repetições aproximadas" nas sequências alvo começou com o algoritmo *GenCompress* (Xin Chen; KWONG; Ming Li, 2001) e foi seguido pelo algoritmo *DNA-Compress* (CHEN et al., 2002). Na técnica do *GenCompress*, a posição e o comprimento das ocorrências não-iniciais das repetições são usados para codificar. Na compressão com *DNACompress*, são consideradas duas fases: a) encontrar todas as repetições aproximadas contendo palíndromos, para as quais a ferramenta de busca *PatternHunter* (MA; TROMP; LI, 2002) é usada; e b) codificação de repetições e não repetições aproximadas, para as quais o método de compressão *LZ77* é explorado.

NMLComp (TABUS; KORODI; RISSANEN, 2003) e *GeNML* (KORODI; TABUS, 2005) são dois métodos que utilizam o modelo *Normalized Maximum Likelihood* (NML). O algoritmo *NMLComp* propõe uma versão do modelo NML para regressão discreta, com o objetivo de codificar as repetições aproximadas e, em seguida, combina-a com um modelo de *Markov* de primeira ordem. O algoritmo *GeNML* apresenta as seguintes melhorias para a metodologia utilizada no método *NMLComp*: a) restringir as correspondências de repetições aproximadas para reduzir o custo de busca das combinações anteriores, bem como obter um modelo NML mais eficiente; b) escolher os tamanhos de bloco que são usados na análise da sequência alvo; e c) introdução de fator de esquecimento, escalável, para o modelo de memória.

A codificação por *Word-Based Tagged Code* (WBTC) é usada no algoritmo *DNA-Compact* (GUPTA; AGARWAL, 2011). Na primeira fase deste método, a sequência alvo é convertida em palavras de forma que as bases A, T, C e G sejam substituídas por A, C, <espaço> A e <espaço> C, respectivamente, ou seja, o alfabeto de 4 símbolos é transformado em um alfabeto de três símbolos. Na segunda fase, a sequência obtida é codificada pelo WBTC. A vantagem do WBTC é não requer armazenar as frequências ou palavras-chave juntamente com o fluxo comprimido, já que o código de palavras depende apenas das classificações.

O método de compressão *DNAEnc3* (PINHO et al., 2011) considera as características estatísticas da sequência genômica e emprega vários modelos de *Markov* concorrentes de diferentes ordens para obter a distribuição de probabilidade de símbolos nas sequências. Segundo o autor, as vantagens deste método incluem: a) explorar uma técnica de programação flexível que fornece a capacidade de lidar com os modelos de ordem até dezesseis; b) a capacidade de manipulação das repetições invertidas; e c) fornecer estimativas de probabilidade que cobrem a ampla gama de profundidades de contexto utilizadas.

Alguns algoritmos de compressão de genomas com abordagem horizontal exploram a optimização por enxame de partículas. O algoritmo *Adaptative Particle Swarm*

Optimization-based Memetic Algorithm (POMA) proposto por (ZHU et al., 2011) baseia no algoritmo *Comprehensive Learning Particle Swarm Optimization* (CLPSO) (PENG, 2011) e também no algoritmo *Adaptive Intelligent Single Particle Optimizer* (AdpISPO). Nesse algoritmo, um *codebook* de *Approximate Repeat Vector* (ARV) é construído e, em seguida, otimizado usando CLPSO bem como AdpISPO para comprimir a sequência alvo. As repetições aproximadas que tem o menor número de variações de símbolos exploram os códigos de ARV candidatos codificados como partículas para alcançar a solução ideal em POMA. Posteriormente, os valores de aptidão ponderada são usados para selecionar as partículas de liderança no enxame. Finalmente, uma pesquisa local baseada em AdpISPO é explorada para afinar as partículas líderes.

O algoritmo *DNA-COMPACT* proposto por (GUPTA; AGARWAL, 2011) explora modelos contextuais complementares e consiste em duas fases. Na primeira fase, as repetições e palíndromos exatos são pesquisados e, em seguida, representados por uma quadrupla comprimida. Na segunda fase, os modelos contextuais e não sequenciais são introduzidos para explorar as características das sequências de DNA; então, as previsões desses modelos são sintetizadas usando o modelo de regressão logística. Neste método, a regressão logística mostra resultados menos tendenciosos, em vez de uma media Bayesiana. *DNA-COMPACT* é capaz de lidar com a compressão de genoma sem referência (compressão horizontal) e baseada em referência (compressão vertical).

O algoritmo *GeCo* proposto por (PRATAS; PINHO; FERREIRA, 2016), derivado de outros dois algoritmos (PINHO; PRATAS; FERREIRA, 2011; PRATAS; PINHO, 2014), explora uma combinação de modelos de contexto de várias ordens para referência livre, bem como para a compressão de sequência genômica baseada em referência. Neste método são introduzidos os *Extended Finite-Context Models* (XFCMs), que são tolerantes a erros de substituição. Além disso, *cache-hashes* são empregados em modelos de alta ordem para tornar a implementação do *GeCo* mais flexível. O *cache-hash* usa uma função *hash* fixa para simular uma estrutura específica, que é um ponto intermediário entre um dicionário e um modelo probabilístico. Para tornar o *GeCo* mais flexível, em termos de otimização de memória, o *cache-hash* considera apenas as últimas entradas *hash* na Memória. Desta forma, a quantificação de memória necessária para executar em qualquer sequência será flexível e previsível.

O algoritmo *MFCOMPRESS* se baseia em um modelo probabilístico (modelo de contexto finito) em conformidade com a propriedade de *Markov*. Tal propriedade estima a probabilidade do próximo símbolo ocorrer imediatamente após k-símbolos (contexto de ordem k), com o objetivo de selecionar a distribuição da probabilidade (PINHO; PRATAS, 2014). *MFCOMPRESS* divide a fonte de dados em duas partes: uma parte contendo o

cabeçalho do registro FASTA e a outra parte contendo as bases nitrogenadas. A parte que lida com as bases nitrogenadas pode ser dividida em duas ou três sub-sequências, ou seja, a sequência principal e outras duas sub-sequências auxiliares. A sequência principal é uma fonte de informação de quatro símbolos (A,T,C,G). Os caracteres maiúsculos da sequência original são convertidos para minúsculos. Se outros símbolos, além daqueles que representam as quatro bases nitrogenadas, também estiverem presentes, serão mapeados para o símbolo '0' na sequência principal. Assim, a primeira sub-sequência auxiliar só será criada quando sequência de DNA original contêm outros símbolos além de (A,T,C,G). Essa sub-sequência extra é responsável por representar todos os caracteres não-ATCG que foram encontrados e substituídos pelo símbolo "0". A terceira sub-sequência só será criada para fazer o mapeamento, caso exista mais de um símbolo não-ATCG diferente. Por fim a sequência contendo símbolos somente (A,T,C,G) é comprimida utilizando o modelo probabilístico.

A ferramenta *Deliminate* poposta por (MOHAMMED et al., 2012) trabalha com duas fases de compressão. Na primeira fase são registradas informações de todos os caracteres não-ATGC e regiões representadas por caracteres minúsculos. Um arquivo é então criado com o mapeamento dos caracteres não-ATGC. Todos os caracteres restantes na sequência são convertidos para símbolos em maiúsculos é processados na segunda fase. Nesta fase, as posições de dois símbolos com as maiores frequências de ocorrência são codificadas em *Delta Encoding* e estes símbolos são subsequentemente eliminadas da sequência restando somente os pares que são menos repetitivos. Os dois símbolos restantes, com as menores frequências, são então representados com um código binário. Os vários arquivos gerados nesse processo são comprimidos com 7Zip para gerar o arquivo final.

3.3 Algoritmos e ferramentas baseados em imagem

Transformar sequências genômicas em imagens, onde o espaço unidimensional é substituído por um espaço bidimensional, é um assunto abordado por (GUO et al., 2015; XIE; ZHOU; GUAN, 2015). (GUO et al., 2015) apresenta duas fases em seu método: i) A *Hilbert Space Filling Curve* Curve é explorada para mapear a sequência alvo em uma imagem; ii) Um modelo de ponderação de contexto é usado para codificar a imagem. O algoritmo *Compressing Genomes as a Image* (CoGI), proposto por (XIE; ZHOU; GUAN, 2015), inicialmente transforma a sequência genômica em uma imagem binária ou *bitmap*, então, usa um método de codificação de partição retangular (seleciona um retângulo dentro da imagem) para comprimir essa imagem (MOHAMED; FAHMY, 1995). Finalmente,

o método explora a codificação da entropia para maior compressão da imagem codificada, bem como os erros de emparelhamento comumente chamados de *mismatches*.

3.4 Ferramentas de compressão para propósito geral

Nesta seção é apresentada apenas uma ferramenta de compressão de propósito geral como exemplo. Esse tipo de ferramenta implementa algoritmos baseados no algoritmo *LZ77* e suas variações (ZIV; LEMPEL, 1977). Esse algoritmo, bem como suas variações, não consegue atingir um percentual de compressão acima de 75% quando se trata de comprimir sequências genômicas. Isso porque o alfabeto genômico é muito pequeno, com apenas 4 símbolos, e também a frequência em que os símbolos aparecem na sequência é muito próxima uma das outras. A ferramenta *7-Zip* é uma ferramenta *Open Source* para compressão sem perda de dados. *7-Zip* é comumente utilizada para compressão de propósito geral tal como textos. Além de operar com o seu próprio formato de arquivo "7z", suporta também vários outros formatos, e.g., ZIP. A compressão "7z" em seu núcleo usa uma variedade de algoritmos, sendo os mais comuns; BZIP2, *Prediction by partial matching* (PPMd), *Lempel–Ziv–Markov chain algorithm* (LZMA) e LZMA2 (PAVLOV, 2018).

3.5 Formatos de arquivo de imagem

O *Portable Network Graphics* (PNG) é um formato de arquivo de imagem que suporta compressão sem perda (DUCE, 2003). Inicialmente, foi desenvolvido para substituir o formato *Graphics Interchange Format* (GIF) (CompuServe Incorporated, 1990) e tornou-se o formato para compressão de imagem mais utilizado na Internet. Possui duas fases de compressão (transformação e codificação) e, na fase de transformação dos dados, faz uso de um método de predição de pixel. Esse método, aplica para cada linha da imagem o melhor filtro possível, tornando cada linha mais otimizada para a compressão do que a linha no estado original. Por fim, o método *DEFLATE* é aplicado para a finalização da compressão.

WebP é um formato de imagem *Open Source*, inicialmente desenvolvido pela empresa *On2 Technologies* e mais tarde adquirido pela empresa *Google* (GOOGLE, 2018). Esse formato comporta compressão com e sem perda de dados. *WebP* comprime sem perda uma imagem com tamanho final de arquivo até 26% menor que o formato PNG. Além disso, se for aplicada a conversão de PNG para *WebP*, a redução do tamanho do arquivo pode chegar a 45%. *WebP* usa fragmentos de imagem já visto para reconstruir

os novos pixeis. O formato também pode usar uma paleta de cores local, caso não seja encontrada uma correspondência interessante. Para a codificação final, *WebP* usa uma variante dos algoritmos *LZ77* e Codificação de Huffman.

Free-Lossless Image Format (FLIF) é um novo formato de imagem (sem perda) (SNEYERS; WUILLE, 2016). Ele é mais eficiente que os formatos sem perda: PNG, *WebP*, *Better Portable Graphics* (BPG) (BELLARD, 2015), JPEG 2000 e *Joint Photographic Experts Group Extended Range* (JPEG XR) (JPEG, 2018). FLIF suporta níveis de cinza, RGB e RGBA, com uma profundidade de cor de 1 a 16 *bits* por canal. Ambos tipos de imagem, assim como imagens e animações são suportadas. O formato FLIF tem suporte para manipular imagens de cores escassas (por exemplo, paletas de 256 cores) efetivamente. Também, FLIF tem um modo entrelaçado e não entrelaçado; ao contrário do PNG, o entrelaçamento de FLIF geralmente produz melhor compressão. O objetivo do entrelaçamento é ser capaz de reconstruir progressivamente uma imagem comprimida. Assim, é possível não ter que carregar todo o fluxo de dados comprimidos se for uma imagem grande e apenas uma pequena pré-visualização for necessária. FLIF usa uma generalização do entrelaçamento do formato de imagem PNG. Em cada etapa do entrelaçamento, o número de pixeis dobra. No primeiro passo é simplesmente um pixel: o pixel no canto superior esquerdo. Em seguida, em cada etapa de entrelaçamento, o número de linhas é duplicado (um passo horizontal) ou o número de colunas é duplicado (um passo vertical). O passo final é sempre um passo horizontal, percorrendo todas as linhas ímpares da imagem.

3.6 Técnicas de transformação

Naïve-bit encoding ou "codificação binária ingênua" é uma técnica que tem como abordagem a exploração da codificação de comprimento fixo de dois ou mais símbolos em um único *byte* (GRUMBACH; TAHI, 1993). Embora trate-se de uma técnica de codificação propriamente dita, nessa pesquisa, a técnica de *Naïve-bit encoding* é utilizada como uma das etapas de transformação dos dados. Essa transformação consiste em aplicar uma simples substituição de cada um dos quatro caracteres do alfabeto genômico {A, T, C, G} por dois *bits* cada (e.g., $A = 00, C = 01, G = 10$ *e* $T = 11$). O simples fato de aplicar a codificação *Naïve-bit encoding* em uma sequência genômica, livre de símbolos não-ATCG, já produz um percentual de economia de espaço de 75%.

Transformações de bases é uma notação para codificar dados de *bytes* arbitrários usando um conjunto restrito de símbolos que podem ser usados convenientemente pelo homem e processados por computadores. Com foco nessa abordagem, é possível aplicar o

agrupamento de *bits* nos processos de transformação dos símbolos do alfabeto genômico {A, T, C, G} de cada sequência. Os agrupamentos binários são combinações de 3, 4, 5 e 6 *bits*. O processo de agrupamento binário é constituído de dois passos. No primeiro passo, é aplicada a transformação *Naïve-bit encoding* a fim de se obter uma *string* de *bits*, precisamente dois *bits* por base. No segundo passo, para cada agrupamento de *bits* é atribuído um símbolo que, em seguida, é substituído por uma cor específica no processo de codificação com imagem.

O que difere entre os agrupamentos com 3, 4, 5 e 6 *bits* é a quantidade de *bits* que são substituídos por cada símbolo, sendo agrupamento com 8, 16, 32 e 64 combinações usando respectivamente 3, 4, 5 e 6 *bits*. Outro ponto que difere entre as combinações é o tamanho do conjunto do novo alfabeto pelos quais os *bits* são substituídos. No agrupamento para 8 combinações é utilizado um alfabeto de 8 símbolos, dado pelo seguinte conjunto {0, 1, 2, 3, 4, 5, 6, 7} (cf. Tabela 3.2).

Tabela 3.2: Exemplo de agrupamento binário com 8 combinações e seus respectivos símbolos.
Fonte: Autor.

bits	Símbolo	*bits*	Símbolo	*bits*	Símbolo	*bits*	Símbolo
000	0	010	2	100	4	110	6
001	1	011	3	101	5	111	7

Para agrupamento de 16 combinações é utilizado um alfabeto de 16 símbolos, dado pelo seguinte conjunto {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f} (cf. Tabela 3.3).

Tabela 3.3: Exemplo de agrupamento binário com 16 combinações e seus respectivos símbolos.
Fonte: Autor.

bits	Símbolo	*bits*	Símbolo	*bits*	Símbolo	*bits*	Símbolo
0000	0	0100	4	1000	8	1100	c
0001	1	0101	5	1001	9	1101	d
0010	2	0110	6	1010	a	1110	e
0011	3	0111	7	1011	b	1111	f

Para agrupamento 32 combinações é utilizado um alfabeto de 32 símbolos, dado pelo seguinte conjunto {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v} (cf. Tabela 3.4).

Por fim, para 64 combinações é utilizado um alfabeto de 64 símbolos, dado pelo seguinte conjunto {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r,

Tabela 3.4: Exemplo de agrupamento binário com 32 combinações e seus respectivos símbolos.
Fonte: Autor.

bits	Símbolo	*bits*	Símbolo	*bits*	Símbolo	*bits*	Símbolo
00000	0	01000	8	10000	g	11000	o
00001	1	01001	9	10001	h	11001	p
00010	2	01010	a	10010	i	11010	q
00011	3	01011	b	10011	j	11011	r
00100	4	01100	c	10100	k	11100	s
00101	5	01101	d	10101	l	11101	t
00110	6	01110	e	10110	m	11110	u
00111	7	01111	f	10111	n	11111	v

s, t, u, v, w, x, y, z, A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z, !, @} (cf. Tabela 3.5).

Tabela 3.5: Exemplo de agrupamento binário com 64 combinações e seus respectivos símbolos.
Fonte: Autor.

bits	Símbolo	*bits*	Símbolo	*bits*	Símbolo	*bits*	Símbolo
000000	0	010000	g	100000	w	110000	M
000001	1	010001	h	100001	x	110001	N
000010	2	010010	i	100010	y	110010	O
000011	3	010011	j	100011	z	110011	P
000100	4	010100	k	100100	A	110100	Q
000101	5	010101	l	100101	B	110101	R
000110	6	010110	m	100110	C	110110	S
000111	7	010111	n	100111	D	110111	T
001000	8	011000	o	101000	E	111000	U
001001	9	011001	p	101001	F	111001	V
001010	a	011010	q	101010	G	111010	W
001011	b	011011	r	101011	H	111011	X
001100	c	011100	s	101100	I	111100	Y
001101	d	011101	t	101101	J	111101	Z
001110	e	011110	u	101110	K	111110	!
001111	f	011111	v	101111	L	111111	@

Vale destacar aqui que os alfabetos mencionados são usados apenas para melhorar a compreensão do processo de transformação do alfabeto original de uma sequência genômica. De outra forma, é possível aplicar diretamente uma cor (pixel) para cada agrupamento de *bits*. Não está sendo abordado a transformação para 4 combinações devido ao fato de que a codificação com *Naïve-bit encoding* já a representa. Quantidades de

combinações acima de 64 fazem com que entropia da informação fique acima de 2. Do mesmo modo, a grande quantidade de símbolos do novo alfabeto acima de 64 combinações demanda uma quantidade significativa de *bits* para representar cada cor, na compressão baseado em-imagem.

As diferentes estruturas de repetições que ocorrem em toda a extensão do DNA são fenômenos importantes que podem ser exploradas com o intuito de minimizar ou até mesmo eliminar as redundâncias de informação. A transformação *Burrows Wheeler Transform* (BWT) (SALOMON, 2007) é uma ideia de transformação popular na bioinformática. O BWT aplicado puramente não é um método de compressão, mas um permutador de símbolos, contribuindo para um rearranjo eficiente. A ideia chave do BWT é permutar os símbolos de uma sequência de tal forma que os símbolos sejam agrupados por sua vizinhança, ou seja, os símbolos similares ficam próximos após a permutação, mesmo se eles estivessem longe na sequência original. Sendo assim, o método de transformação BWT pode ser utilizado com o objetivo melhorar a economia de espaço após a compressão com imagem, à medida que alguns formatos de imagens se aproveitam de cores vizinhas iguais.

BWT executa uma permutação dos caracteres na sequência, tal que, os caracteres semelhantes ficam agrupados, levando em conta o contexto lexicográfico. Dada uma sequência de entrada $T = t_1, t_2, \ldots, t_n$ a transformação BWT é constituída de 3 passos:

a) Execução de n permutações em T realizando rotações cíclicas dos caracteres em T. As permutações formam uma matriz $M' = n \times n$ em que cada linha em M' representa uma permutação de T.

b) Ordenação lexicográfica das linhas de M' formando outra matriz M. Essa nova matriz possui T em uma das suas linhas.

c) Guardar a última coluna L da matriz M que foi permutada e ordenada bem como o número do *id* da linha que corresponde a sequência original T. A saída do método BWT é composta pelo par L e *id*. Este, deve ser armazenada para que o processo reverso possa ser executado.

BWT pode ser aplicado como em uma sequência genômica como método de transformação da seguinte forma: seja $S = ACTAGA$ a sequência de entrada para a transformação BWT com tamanho $n = 6$. Ao final da sequência é adicionado um caractere de marcação para sinalizar o final da sequência. Para cada caractere da sequência, realiza-se uma permutação, resultando em uma matriz $n \times n$.

A saída resultante do exemplo mostrado (cf. Tabela 3.6) é o par $(L, id) = (GATAAC, 2)$, sendo que L é a sequência formada pela penúltima coluna de M e o

Tabela 3.6: Transformação BWT. M' é a matriz das rotações cíclicas antes da ordenação. M é a matriz depois da ordenação. F e L são as strings da primeira e última coluna respectivamente.
Fonte: Autor.

	M'		M			
id			**F**		**L**	**id**
0	A C T A G A !	← Sequencia original	**A**	! A C T A	G	5
1	C T A G A ! A	Ordenada →	**A**	**C T A G**	A !	0
2	T A G A ! A C		**A**	G A ! A C	T	3
3	A G A ! A C T		**C**	T A G A !	A	1
4	G A ! A C T A		**G**	A ! A C T	A	1
5	A ! A C T A G		**T**	A G A ! A	C	2

id é o número da linha onde a sequência L começa. Para o processo reverso é necessário ordenar lexicograficamente L para obter F e então criar um novo vetor de transformação V para mapear um por um para todos os elementos entre L e F. O vetor resultante do exemplo da Tabela 3.6 seria $V = [516234]$. Com o vetor V, o *id* e também a última coluna L, é possível reconstruir a sequência original pela Equação 3.1.

$$S[n-1-i] \leftarrow L[Vi[id]],\ para\ i = 0, 1, \ldots, n-1 \\ onde\ V0[j] = j,\ e\ Vi+1[j] = V[Vi[j]] \tag{3.1}$$

O algoritmo *Move To Front transform* (MTF) (BENTLEY et al., 1986) converte os dados em uma sequência de inteiros, com a expectativa que valores dos inteiros sejam pequenos e possam ser efetivamente transformados usando um algoritmo de codificação estatística. O codificador MTF mantém uma lista de símbolos, chamada de lista MTF, que é inicializada com todos os símbolos do alfabeto da sequência alvo (cf. Tabela 3.7). Em seguida, para cada símbolo nos dados de entrada, o codificador fornece sua posição na lista MTF na forma de um inteiro e atualiza a lista MTF. Um símbolo atualmente codificado é movido da posição atual na lista MTF para o início da lista. A propriedade mais importante dessa técnica é que os símbolos usados recentemente estão próximos do início da lista. Símbolos iguais aparecerão frequentemente próximos uns dos outros nos dados de saída, portanto, esses símbolos serão convertidos em inteiros pequenos. Em geral, os inteiros pequenos são mais frequentes, de modo que são codificados em menos *bits* que os inteiros maiores, usando uma codificação estatística como o *Huffman* ou a codificação aritmética. O algoritmo de codificação MTF transforma uma sequência de DNA em uma sequência de números, desde que o alfabeto seja conhecido antes. O algoritmo para o

método MTF opera da seguinte forma:

a) Inicialize a lista E com cada letra do alfabeto.

b) Leia às letras uma de cada vez. Para um caractere A que acabou de ser lido, anote o índice de A em E e mova A em E para a frente de E. Assim, E virá a ter diferentes permutações das letras do alfabeto à medida que S é processado e obtém uma sequência de índices.

Sejam as variáveis aplicadas no método MTF: a sequência $S = GGGTTTAATTCCC$, o alfabeto $A = \{A, C, G, T\}$ e a lista $E = [\emptyset]$.

Tabela 3.7: Exemplo do método de transformação *Move To Front*.
Fonte: Author.

Símbolos de S	E (antes)	Índice	E (depois)
G	TACG	3	GTAC
G	GTAC	0	GTAC
G	GTAC	0	GTAC
T	GTAC	1	TGAC
T	TGAC	0	TGAC
T	TGAC	0	TGAC
A	TGAC	3	ATGC
A	TGAC	0	ATGC
T	ATGC	1	TAGC
T	ATGC	0	TAGC
C	TAGC	3	CTAG
C	CTAG	0	CTAG
C	CTAG	0	CTAG

De acordo com a Tabela 3.7 a transformação final é 30010030103000. Uma maneira de melhorar a compressão é primeiro aplicar MTF na sequência genômica e em seguida. fazer a codificação aritmética ou *Huffman Encoding* em combinação com a *Run Length Encoding* (RLE), descrito a seguir.

A técnica de compactação RLE (ROBINSON; CHERRY, 1967) é usada quando um determinado arquivo contém dados redundantes ou uma longa execução de caracteres semelhantes. A cadeia repetida ou os caracteres presentes no arquivo de entrada ou na mensagem são chamados de (*Run*), que é codificada em 2 ou mais *bytes*. O primeiros *bytes* contém o número de vezes que determinado símbolo aparece na execução e o último *byte* representa o valor do símbolo na execução. Esse algoritmo, que codifica baseado no comprimento das repetições, consiste em substituir grandes sequências de dados repetidos por apenas um item desses dados, antecedido por um contador que mostra quantas vezes

esse item é repetido. Exemplo de codificação com RLE para a sequência de DNA: $S =$ *AAAAAAACCCCCCCTTTTTTTTTGGGGGGGAAAAAAAGGGGGGGG*
A sequência S, de comprimento 45, tem muitas repetições. Usando o algoritmo RLE as execuções repetitivas podem ser substituídas por um símbolo apenas, antecedido por um contador.

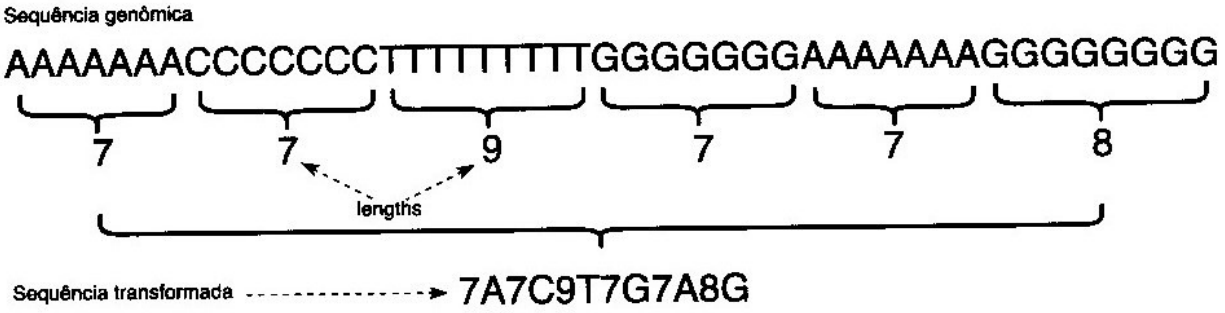

Figura 3.2: Codificacão com Run Length Encoding.
Fonte: Autor.

De acordo com a Figura 3.2 o resultado da codificação usando RLE deixou a sequência 50% menor que a original. Ainda assim, deve ser observado a partir de qual comprimento das repetições o algoritmo é eficiente, pois para armazenar a nova sequência representada com o novo alfabeto de 13 símbolos $(0, 1, ...9, A, T, C, G)$ são necessários no mínimo 4 *bits* por símbolo.

3.7 Dataset de testes para compressão de genoma

Com o objetivo de encontrar material relevante, foram realizadas buscas na literatura que retornassem como resultado, uma *Revisão Sistemática da Literatura* (RSL) sobre *dataset* de testes para compressão de sequência genômica, com abordagem horizontal e sem perda de dados. No entanto, nenhuma RSL do gênero foi encontrada nas bases de artigos selecionadas (*Scopus* e *Pubmed*). Em contrapartida, foi encontrado um trabalho, cujos autores propuseram um *dataset* para a realização de *benchmark* de ferramentas de compressão de sequências genômicas completas (BIJI; NAIR, 2016). O trabalho menciona 12 ferramentas especializadas de compressão de sequências de DNA além de propor um conjunto de dados e de métricas para avaliação de performance. Dentre as 12 ferramentas apresentadas no *benchmark*, foram escolhidas as 3 melhores, levando em conta critérios (cf. Tabela 3.8) para essa escolha. As ferramentas selecionadas foram: COMRAD (KURUPPU et al., 2012), *DLIMINATE* (MOHAMMED et al., 2012) e *MFCOMPRESS* (PINHO; PRATAS, 2014). Assim, essas 3 ferramentas fizeram parte dos testes de *ben-*

chmark da pesquisa de BIJI; NAIR. Ainda, os critérios para a seleção das ferramentas foram o tempo para executar a compressão e a taxa de economia de espaço obtida. Uma observação que deve ser feita aqui é o fato de que as taxas de economia de espaço informada no *benchmark* foram declaradas nos artigos dos próprios autores das ferramentas especializadas. Por fim, o último critério para a seleção das ferramentas de teste, segundo BIJI; NAIR, foi o tamanho do *dataset* que a cada ferramenta suporta.

A Tabela 3.8 mostra o critério de seleção das três melhores ferramentas descritas no trabalho de (BIJI; NAIR, 2016) baseadas nas colunas "Tempo" e "Percentual" de compressão.

Tabela 3.8: Lista de ferramentas mencionadas no artigo do *benchmark*.

Ferramenta	Tamanho dataset	Tempo	Percentual
BioCompress	Pequeno	?	?
GenCompress	Pequeno	?	?
Cfact	Pequeno	?	?
DNACompress	Grande	Lento	?
DNAPack	?	?	?
GeMNL	?	Lento	Bom
DNAsequitur	Grande	?	Baixo
Xmcompress	Grande	Lento	Bom
DNABIT	Pequeno	?	?
COMRAD	Grande	Razoável	?
Deliminate	Grande	Rápido	Melhor
MFCompress	Grande	Razoável	Razoável

Apesar da ferramenta COMRAD estar entre as três melhores citadas, não foi informado no artigo do *benchmark* qual o critério utilizado para o percentual de compressão dessa ferramenta. Assim, o autor não deixa claro quais valores representam o percentuais de compressão: baixo, bom, razoável e melhor, bem como os tempos: lento, razoável e rápido.

Os autores BIJI; NAIR, propõem também em sua pesquisa um conjunto de dados como *dataset* para ser utilizado nos testes de *benchmark*. Esse *dataset* foi proposto como um conjunto de sequências genômicas para representar uma amostra científica válida do universo de sequências disponíveis no NCBI. Para compor essa amostra, foram selecionadas sequências genômicas de 1.105 organismos procariontes, 200 plasmídeos, 164 vírus e 65 organismos eucariontes, totalizando 1.469 organismos. Para a seleção das sequências como amostra, foram utilizados vários métodos que incluem: amostragem aleatória simples, amostragem sistemática, estratificação, clusterização e amostragem multiestágios (KALTON, 1983).

3.8 Métricas de avaliação da compressão de genomas

A taxa de compressão de dados é definida como a proporção entre o tamanho não comprimido e o tamanho comprimido dos dados (SALOMON, 2007). Assim, uma representação para demonstrar a taxa compressão de um arquivo de 100 *Megabytes* com redução para 20 *Megabytes* tem uma taxa de compressão de $\frac{100}{20} = 5$ (cf. Equação 3.2), frequentemente notada como taxa de compressão na razão de 5 : 1 (cinco para um) ou 5/1 (cinco por um).

$$CR = \frac{UncompressedFileSize}{CompressedFileSize} \tag{3.2}$$

onde CR é a taxa de compressão (Compression Ratio), $UncompressedFileSize$ é o tamanho do arquivo não comprimido e $CompressedFileSize$ é o tamanho do arquivo comprimido.

A economia de espaço representa o percentual de redução do tamanho comprimido em relação ao tamanho não comprimido (SALOMON, 2007). Assim, uma representação que comprime um arquivo de 100 *Megabytes* para 20 *Megabytes* resultaria em uma economia de espaço de $(1 - \frac{20}{100}) \times 100 = 80$ (cf. Equação 3.3), frequentemente anotada como porcentagem de economia de espaço de 80%.

$$SS = \left(1 - \frac{CompressedFileSize}{UncompressedFileSize}\right) \times 100 \tag{3.3}$$

onde SS é a economia de espaço (Space Savings), $CompressedFileSize$ é o tamanho do arquivo comprimido e $UncompressedFileSize$ é o tamanho do arquivo não comprimido.

O Teste de *Friedman* é um teste não paramétrico útil para fazer comparações sobre amostras independentes entre três ou mais hipóteses. *Friedman* faz um rankeamento dos dados ao invés de utilizar os valores brutos para o cálculo da estatística. O melhor algoritmo recebe o *rank* 1, o segundo melhor recebe o *rank* 2 e assim subsequentemente. O Teste de *Friedman* pode ser usado do seguinte modo para calcular se existem valores repetidos. Seja o *rank* r_j^k do k^{th} de m algoritmos no j^{th} de N conjunto de dados. O teste de *Friedman* compara a média dos *rankings* dos algoritmos $R_j = \frac{1}{N}\sum_j r_j^k$. Sob a hipótese nula, todos os algoritmos são equivalentes se seus *ranks* são iguais. de acordo com a estatística de *Friedman* (cf. Equação 3.4), que é distribuída de acordo com X_F^2 com *(m - 1)* graus de liberdade (DEMŠAR, 2006).

$$X_F^2 = \frac{12N}{m(m+1)}\left[\sum_j R_k^2 - \frac{m(m+1)^2}{4}\right] \tag{3.4}$$

Se a hipótese nula for rejeitada, pode-se prosseguir com o teste de *Nemenyi* (NEMENYI, 1962). *Nemenyi* faz comparação entre todos os algoritmos uns com os outros. O desempenho de dois algoritmos é significativamente diferente se as classificações médias correspondentes diferirem em pelo menos uma diferença crítica (*Critical Difference* (CD)) dada pela Equação 3.5, onde $q\alpha$ é uma estatística de intervalo que depende de um nível de significância necessário α.

$$CD = q\alpha\sqrt[2]{\frac{m(m+1)}{6N}} \tag{3.5}$$

3.9 Considerações finais

Neste capítulo foi apresentado a revisão bibliográfica das ferramentas especializadas para compressão de sequências genômicas, com abordagem horizontal. De mesmo modo, foi apresentado um conjunto de sequências genômicas como conjunto de dados alvo para os testes dessa pesquisa. Por fim, foi apresentado alguns formatos de imagens que aceitam compressão sem perda de dados. Vale enfatizar aqui que existem na literatura outros formatos de imagens, por exemplo da família JPEG, que trabalham com compressão sem perda de dados. No entanto, como a proposta desse trabalho é apresentar um método de compressão baseado em formato de arquivo de imagem, não foi aprofundada a pesquisa quanto aos outros formatos de imagens, além dos que foram testados, que poderiam ser usados também para a compressão de dados genômicos. Para tal, é necessário uma pesquisa aprofundada abordando os ganhos em termos de compressão dos formatos de imagens existentes na literatura.

Capítulo 4

Método

Esta seção apresenta o método proposto para a compressão de sequências genômicas baseado em formato de arquivo de imagem. A seguir são apresentados o *dataset* e as ferramentas *baseline* bem como o diagrama do método proposto e as suas fases de compressão dos dados. O método possui três fases, a saber: a preparação dos dados, a transformação dos dados e por fim a codificação dos dados. Na fase de preparação é possível remover ou omitir a remoção dos símbolos não-ATCG. Na fase de transformação é possível aplicar ou omitir a aplicação das técnicas de transformação dos dados que tratam as redundâncias das sequências genômicas. Por fim, na fase de codificação o resultado da transformação, ou sem transformação, é codificado com um formato de arquivo de imagem. Para cada fase do método foi proposto um algoritmo que implementa a fase.

4.1 Dataset e Ferramentas *baseline*

O conjunto de dados utilizado para a avaliação do desempenho nessa pesquisa (cf. 4.1) é um subconjunto com 1.547 sequências genômicas selecionadas do conjunto definido por (BIJI; NAIR, 2016). Assim, não foi utilizado nenhum genoma de vírus ou arquivos multi-FASTA. Os arquivos multi-FASTA não foram utilizados por serem apenas uma concatenação de arquivos FASTA simples. Também não foi utilizada nessa pesquisa sequências genômicas que possuem mais que 268.435.456 bases nitrogenadas. Essa limitação, embora possa ser contornada a divisão da sequência em mais de um arquivo, ocorre por causa do formato de imagem *WebP*. Na versão utilizada, o formato *WebP* não permite imagens com a matriz de pixeis superiores a 2^{28} pixeis. Assim, o conjunto final de sequências genômicas utilizadas na avaliação consiste no genoma de 1.163 organismos, variando dos seguintes reinos biológicos: *Animalia*, *Archaea*, *Bacteria*, *Fungi Plant* e *Protist*, conforme é apresentado na 4.1. Como em alguns casos, o genoma

de um organismo é dividido em mais de um arquivo, no final, o conjunto de arquivos no formato *FASTA* consiste em 1.547 arquivos distintos.

Tabela 4.1: Distribuição de sequências genômicas por reinos. Em algumas situações, a quantidade de sequências é maior do que a quantidade de organismos, uma vez que o genoma destes organismos está dividido em arquivos FASTA distintos.

Reinos	**Número de Organismos**	**Número de Sequências (*)**
Animalia	3	14
Archaea	24	24
Bacteria	1.101	1.101
Fungi	27	275
Plant	3	23
Protist	5	110
Total	**1.163**	**1.547**

(*) Cada sequência representa um arquivo FASTA distinto.

A partir da lista de ferramentas apresentadas na 3.8, duas ferramentas foram escolhidas como ferramentas *baseline* de comparação com o método de compressão baseado em formato de imagem, proposto nessa pesquisa. A ferramenta *COMRAD*, apesar de ser apresentada entre as três melhores no artigo do *benckmark*, não foi utilizada nessa pesquisa devido aos fatos: trabalha apenas com o alfabeto genômico {A, T, C, G}; os autores BIJI; NAIR relataram não ter conseguido comprimir 70% das sequências do *dataset* proposto com a ferramenta *COMRAD*.

4.2 Fases da compressão dos dados

4.2.1 Fase de preparação dos dados

A fase de preparação dos dados, ou pré-processamento dos dados, é a fase responsável por organizar o conjunto de dados que serão utilizados no método de compressão baseado em imagem (cf. Figura 4.1 - Fase de Preparacão). O método proposto aceita como parâmetro de entrada arquivos do tipo *FASTA* simples. A compressão de arquivos multi-FASTA será discutido na seção de trabalhos futuros.

4.2.1.1 Leitura do arquivo FASTA

O processo de compressão é iniciado com a leitura linha a linha do arquivo no formato *FASTA* para a extração dos dados (cf. Algoritmo 1, linhas 2 até 9). Os dados extraídos são então divididos em dois segmentos, sendo o primeiro segmento o cabeçalho

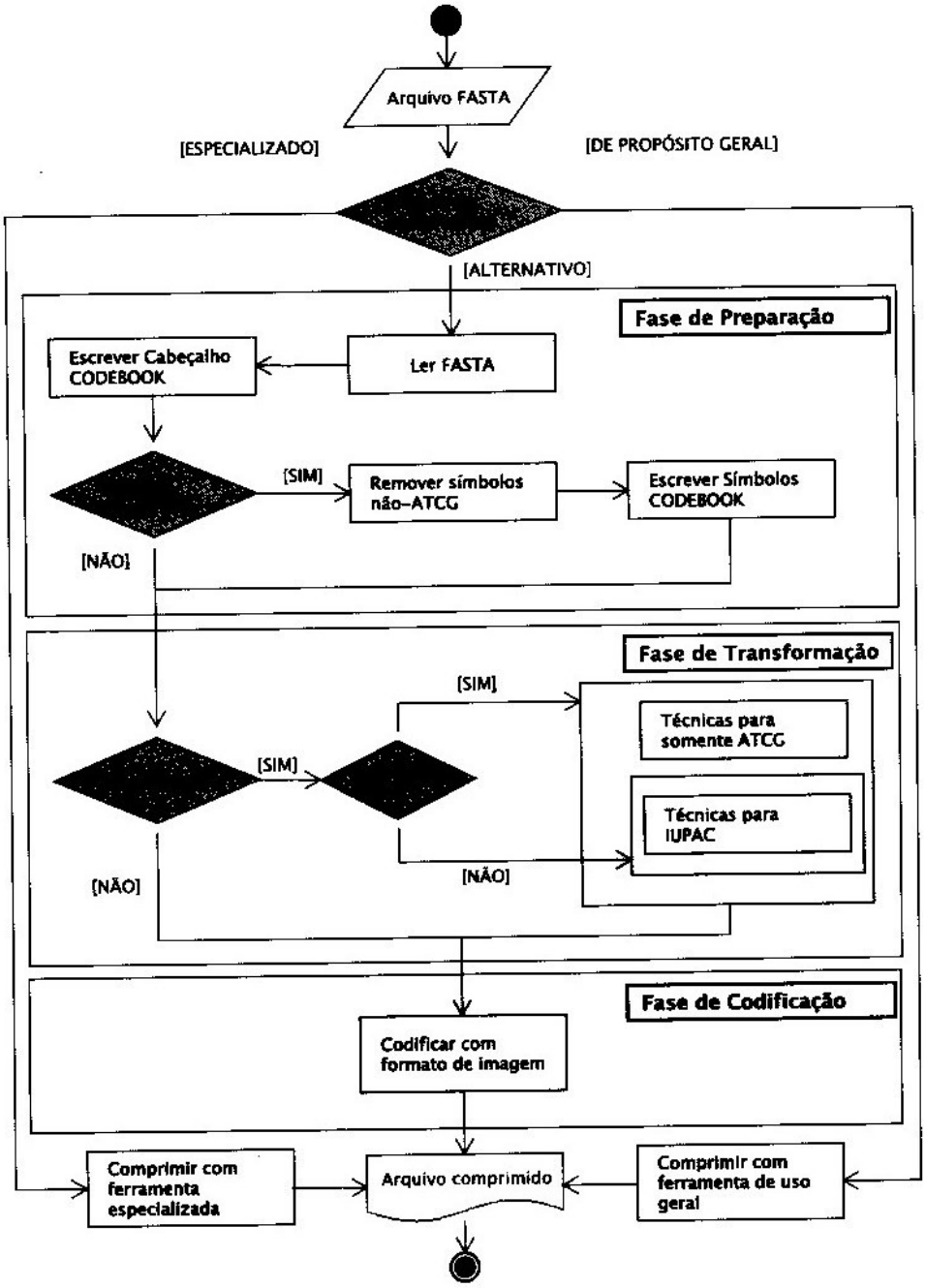

Figura 4.1: Diagrama do método proposto.

do formato *FASTA* e o segundo segmento a sequência genômica propriamente dita (cf. 3). Esses segmentos são carregados em memória para serem manipulados.

4.2.1.2 Criação do *codebook*

Ao ser iniciado a leitura do arquivo FASTA um arquivo auxiliar, chamado de *codebook*, é criado em memória (cf. Algoritmo 1, linha 10). O *codebook* consiste em duas partes: o cabeçalho e o mapeamento dos caracteres não-ATCG. A primeira parte é re-

Algorithm 1 Criação do *codebook*

```
1: procedure CODEBOOK(inputFile, removeNonATCG)
2:     file ← readFile(inputFile)
3:     genome ← [ ]
4:     firstLine ← file.readLine()
5:     secondLine ← file.readLine()
6:     genome.append(secondLine)
7:     while line ← file.readLine() ¬ NULL do
8:         genome.append(line)
9:     end while
10:    codebook ← [ ]
11:    totalSize ← sizeOf(genome)
12:    columns ← sizeOf(secondLine)
13:    codebook.add(firstLine, columns, totalSize)
14:    atcgSequence ← genome
15:    if removeNonATCG is True then
16:        atcgSequence ← [ ]
17:        lastDelta ← 0
18:        s ← sizeOf(atcgSequence)
19:        i ← 0
20:        while i < s do
21:            symbol ← genome[i]
22:            if symbol ∈ (A, T, C, G) then
23:                atcgSequence.append(symbol)
24:                i ← i + 1
25:            else
26:                delta ← i − lastDelta
27:                lastDelta ← i
28:                length ← i
29:                while i < (s − 1) ∧ symbol = genome[i − 1] do
30:                    length ← length + 1
31:                    i ← i + 1
32:                end while
33:                codebook.add(delta, symbol, length)
34:                i ← i + 1
35:            end if
36:        end while
37:    end if
38:    Text text ← new Text(codebook)
39:    SevenZip.compress(text.save())
40:    return atcgSequence
41: end procedure
```

presentada na primeira linha do *codebook*, onde é escrito o cabeçalho de forma literal, o tamanho original da sequência genômica e também o comprimento das colunas, quando acontece a quebra de linha na sequência. A segunda parte do *codebook* recebe as informa-

ções que estão representadas no formato *FASTA* a partir da segunda linha da sequência. Naquela região são armazenadas, linha a linha, as informações dos caracteres não-ATCG que ocorrem na sequência genômica original.

4.2.1.3 Escrita do cabeçalho no *codebook*

Ao final da leitura do conteúdo do arquivo no formato *FASTA*, o algoritmo armazena em variáveis o cabeçalho, o comprimento das colunas e o tamanho original da sequência genômica (cf. Algoritmo 1, linhas 4, 11 e 12). O comprimento das colunas pode variar de 70 a 80 colunas (cf. 3). Nesse ponto da execução o algoritmo já fez a contagem das colunas para obter essa informação. Por fim, os valores são escritos no *codebook* e são de suma importância para o processo de descompressão (não tratado nesse trabalho) caso necessário (cf. Algoritmo 1, linha 13).

4.2.1.4 Procurar caracteres não-ATCG

Uma segunda iteração ocorre para a leitura dos símbolos da sequência genômica extraída do arquivo FASTA. O algoritmo avalia símbolo por símbolo existente com o objetivo de verificar se o símbolo em questão pertence ao alfabeto {A, T, C e G} (cf. Algoritmo 1, linhas 20 até 36). Quando o símbolo pertence ao alfabeto mencionado ele é acrescentado à uma *stream*, criada em memória, nomeada de *atcgSequence*, formando assim uma nova sequência genômica livre de caracteres não-ATCG.

4.2.1.5 Calcular delta da posição não-ATCG

Para o armazenamento dos símbolos não-ATCG são criadas variáveis de controle chamadas de *delta*, *lastDelta* e *length* (cf. Algoritmo 1, linhas 17, 27 e 28). A variável *delta*, inicializada com valor 0, é responsável por guardar o índice da ocorrência do símbolo não-ATCG subtraído do valor de i menos *lastDelta*. A variável *length*, inicializada com o valor 1 armazena a quantidade de repetições encadeadas ocorridas para um determinado símbolo. O incremento de *delta* e *length* acontece quando um símbolo que não pertence ao alfabeto {A, T, C e G} é encontrado, então a sua posição no índice da sequência genômica original, subtraída do valor de *lastDelta* é atribuída para *delta*. Enquanto o próximo símbolo, na sequência original, for igual o símbolo corrente, o algoritmo incrementa o valor de *length* até atingir o fim das repetições da *substring*.

4.2.1.6 Escrita do delta e símbolo não-ATCG no *codebook*

A escrita do delta e do símbolo não-ATCG no *codebook* é de suma importância para os processos de busca na imagem (trabalhos futuros), sem precisar descomprimir, e também para o processo de descompressão (não tratado nesse trabalho) caso fosse necessário. Para cada símbolo não-ATCG na sequência original ou um conjunto de símbolos não-ATCG subsequentes, uma linha é escrita no codebook no seguinte formato: $(delta, symbol, length)$ Ao final da execução, antes do Algoritmo 1 retornar a nova sequência, livre de símbolos não-ATCG, o *codebook* é escrito em um arquivo de texto e comprimido com a ferramenta de compressão de propósito geral 7-zip.

4.2.2 Fase de transformação dos dados

A ideia básica desta fase é a aplicação de algumas técnicas para a transformação dos dados da sequência genômica de entrada (cf. Figura 4.1 - Fase de Transformação). O objetivo principal é reorganizar os símbolos da sequência com vistas à uma das seguintes possibilidades: diminuir a entropia da informação, diminuir o tamanho da sequência ou rearranjar a sequência para ter um aproveitamento melhor no tratamento dos símbolos repetidos (cf. Algoritmo 2 linhas 3, 4 e 5). A sequência genômica também pode ser comprimida, sem aplicar técnica de transformação dos dados. Essa opção está ligada ao fato de que para algumas técnicas de transformação de dados, por exemplo o algoritmo BWT, o processo de busca nos dados, sem descomprimir, tem maior complexidade de tempo. No entanto, essa pesquisa não considera a descompressão da imagem para o formato FASTA original, embora, para todas as técnicas utilizadas como exemplo seria possível realizar o processo reverso.

Algorithm 2 Transformação dos dados

```
1: procedure TRANSFORM(sequence, method, atcg)
2:     switch method do
3:         case bwt (s,p) ← bwt(sequence)
4:         case mtf (s,p) ← mtf(sequence)
5:         case rle (s,p) ← rle(sequence)
6:         case bitcombine
7:             if atcg is TRUE then (s,p) ← bitCombine(sequence)
8:             end if
9:     return (s,p)
10: end procedure
```

Este método propõe, para a fase de transformação dos dados, operar com algoritmos que aceitam como parâmetro de entrada sequências genômicas originais (sem a

remoção dos símbolos não-ATCG) e algoritmos que aceitam sequências genômicas (cujos símbolos não-ATCG são removidos). É importante salientar que o método de compressão proposto não trata de que forma são aplicas as técnicas de transformação, se encadeadas (mais de uma técnica aplicadas uma após a outra) ou simples (apenas uma técnica).

4.2.2.1 Algoritmos para transformação dos dados

Para a fase de transformação dos dados alguns algoritmos existentes na literatura, tais como BWT, MTF e RLE, podem ser utilizados. Para o rearranjo da sequência com BWT e MTF é necessário guardar algumas informações inerentes à cada algoritmo, definido pela variável p. Para BWT é preciso guardar a posição no índice da sequência original do caractere não terminal. De mesmo modo, para MTF é preciso guardar o alfabeto na ordem quando ocorre a última "movimentação para frente". Essas informações, tanto para BWT quanto para MTF são armazenadas nos metadados da imagem, por se tratarem de no máximo 15 símbolos, que é a quantidade máxima de símbolos diferentes que a tabela IUPAC pode ter. O terceiro exemplo de algoritmo para a transformação da sequência é o RLE. Por se tratar de contagem dos símbolos repetidos cujo número de repetições compões a sequência transformada, nenhuma informação é guardada nos metadados da imagem para RLE. Para cada caso a variável s recebe a sequencia transformada que é retornada pelo algoritmo.

4.2.2.2 Algoritmos para alfabeto somente ATCG

As transformações das sequências que possuem somente símbolos do alfabeto genômico {A, T, C, G}, além dos 3 algoritmos mencionados na última seção, para exemplificar pode ser utilizado o método de agrupamento binário *bitCombine* (cf. Algoritmo 2 linha 7). Esse algoritmo depende da transformação prévia com *Naïve-bit encoding* (cf. Algoritmo 3 linhas 3 até 11) sobre sequência genômica original para realizar o agrupamento. A ideia básica é agrupar os bits da *bitstring* com um dos números de combinações: 8, 16, 32 ou 64. Para cada combinação é atribuído um novo símbolo que, mais tarde, será substituído por uma cor do sistema RGB (cf. Algoritmo 3 linha 18). Dependendo da quantidade de *bits* na *bitstring* é necessário guardar a sobra de *bits* B' (cf. Algoritmo 3 linha 24) para o processo de descompressão, caso seja necessário descomprimir. Exemplo: Seja a *bitstring* $B = [b_1, ..., b_n]$, o conjunto das combinações $C = \{8, 16, 32, 64\}$, a quantidade de combinação $c \in C$, o comprimento de um elemento das combinações $x = \log_2 c$, o comprimento da sobra binária $l = c(\mathrm{mod} x)$, então, a sobra de *bits* $B' = B[b_{n-l}. ...b_n]$ *se* $l > 0$.

Algorithm 3 Agrupamento de bits

1: **procedure** BITCOMBINE(sequence, combinations)
2: $newSequence \leftarrow \emptyset$
3: $bitstring \leftarrow \emptyset$
4: $S(x) \leftarrow \begin{cases} 00, & if\ x = A \\ 11, & if\ x = T \\ 01, & if\ x = C \\ 10, & if\ x = G \end{cases}$
5: $s \leftarrow sizeOf(sequence)$
6: $i \leftarrow 0$
7: **while** $i < s$ **do**
8: $symbol \leftarrow genome[i]$
9: $bitstring.append(S(s))$
10: $i \leftarrow i + 1$
11: **end while**
12: $s \leftarrow sizeOf(bitstring)$
13: $bitGroup \leftarrow \emptyset$
14: $i \leftarrow 0$
15: **while** $i < s$ **do**
16: $bitGroup.append(bitstring[i])$
17: **if** $i + 1 \mod combinations = 0$ **then**
18: $newSymbol \leftarrow bitsToSymbol(bitGroup)$
19: $newSequence.append(newSymbol)$
20: $bitGroup \leftarrow \emptyset$
21: **end if**
22: $i \leftarrow i + 1$
23: **end while**
24: **return** $(newSequence, bitGroup)$
25: **end procedure**

4.2.3 Fase de codificação dos dados

A fase de codificação dos dados (cf. Figura 4.1 - Fase de Codificação) da sequência é a fase em que cada símbolo, seja da sequência original ou da sequência transformada. é convertido para um pixel, cuja cor é do sistema RGB, conforme ilustrado na Figura 4.2(A). Para isso, a partir do comprimento da sequência de símbolos, o algoritmo do método proposto gera uma matriz de pixeis (cf. Algoritmo 4). A ordem da matriz foi definida como um quadrado, uma vez que alguns formatos de imagem têm limitações no comprimento das linhas e altura das colunas, como *WebP*, e podem impor limitações quanto ao comprimento da sequência genômica a ser usada. No caso de *WebP* é reservado os primeiros 28 *bits* (no arquivo do formato), sendo 14 *bits* para o comprimento e 14 *bits* para a altura. Para definir a ordem da matriz, foi extraído o comprimento da sequência a ser comprimida e, em seguida, calculado de acordo com a equação na linha 2 do Algoritmo 4. O arredondamento

para cima ou para baixo está ligado ao fato de que raramente o comprimento de uma sequência será de raiz exata. Após a criação da matriz cada símbolo da sequência é codificado para um pixel.

Algorithm 4 Compressão com imagem

1: **procedure** COMPRESS(sequence, format, metadata)
2: $squareRoot \leftarrow \sqrt{sizeOf(sequence)}$
3: $rows \leftarrow 0$
4: $columns \leftarrow 0$
5: **if** $squareRoot \mod 2 = 0$ **then**
6: $rows \leftarrow squareRoot$
7: $columns \leftarrow squareRoot$
8: **else if** $(squareRoot - \lfloor squareRoot \rfloor) > 0.5$ **then**
9: $rows \leftarrow \lceil squareRoot \rceil$
10: $columns \leftarrow \lceil squareRoot \rceil$
11: **else if** $(squareRoot - \lfloor squareRoot \rfloor) \leq 0.5$ **then**
12: $rows \leftarrow \lceil squareRoot \rceil$
13: $columns \leftarrow \lfloor squareRoot \rfloor$
14: **end if**
15: $M \leftarrow new\ Pixel[columns][rows]$
16: $S(x) \leftarrow \begin{cases} 0, & if\ x = A \\ 1, & if\ x = C \\ 2, & if\ x = G \\ 3, & if\ x = T \end{cases}$
17: $currentLine \leftarrow 0$
18: $currentColumn \leftarrow 0$
19: $i \leftarrow 0$
20: **while** $i < sizeOf(sequence)$ **do**
21: $s \leftarrow sequence[i]$
22: $M.setRGB(currentColumn, currentLine, RGB(S(s), 0, 0))$
23: $currentColumn \leftarrow currentColumn + 1$
24: **if** $currentColumn = columns$ **then**
25: $currentColumn \leftarrow 0$
26: $currentLine \leftarrow currentLine + 1$
27: **end if**
28: $i \leftarrow i + 1$
29: **end while**
30: $ImageFormat\ image \leftarrow new\ ImageFormat(format, M)$
31: $image.setMetadata(metadata)$
32: **return** *image*
33: **end procedure**

A ideia básica é converter cada símbolo em um pixel com uma cor diferente em nível de cinza. A Figura 4.2 mostra um exemplo (em escala macro) da imagem resultante da compressão de uma sequência em que cada símbolo foi convertido para uma cor. A representação esquemática da fase de codificação está com os pixeis em cores com níveis de

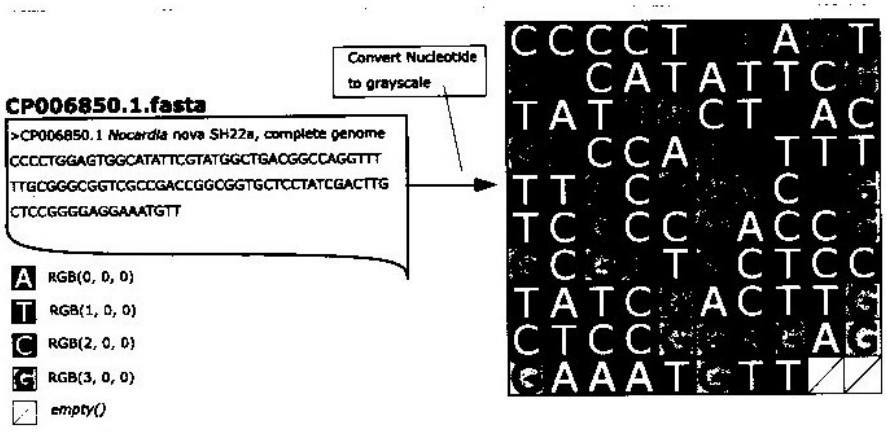

Figura 4.2: Representação esquemática da fase de codificação do método de compressão de dados genômicos baseado em formato de imagem. A sequência genômica, livre de símbolos não-ATCG, é convertida para uma matriz de pixeis e cada pixel recebe uma cor correspondente.
Fonte: Autor.

cinza. Para cada símbolo diferente, da sequência original ou da sequência transformada, é atribuída uma cor do sistema RGB. O uso do sistema RGB é necessário como parâmetro de entrada para gerar cada pixel da imagem nos formatos *WebP* e *FLIF*. Assim, a quantidade de cores podem variar de 4 cores, para compressão da sequência original e livre de símbolos não-ATCG, até 64 cores, para compressão, quando aplicada a transformação utilizando o agrupamento de bits de 64 combinações. Para manter uma variação de cinza e atender o número de cores necessário foi estabelecido que apenas a cor vermelha (*red*) do sistema RGB deve variar. Isso se deve ao fato de que em testes preliminares, apontaram um ganho, ainda que em torno de 1%, no percentual de compressão (discutido nos resultados). Exemplo: Seja o vetor de símbolos $S = [s_0, ..., s_{63}]$, e o símbolo s em que $s \in S$, então a cor c do sistema RGB se dará pela função $c = RGB(s_i, 0, 0)$. Por fim, ao concluir o mapeamento e a atribuição de cores para os pixeis da matriz, o Algoritmo 4 e linha 32, retorna a imagem de acordo com o formato especificado.

4.3 Considerações finais

Neste capítulo foi apresentado o método proposto de compressão de genomas com as seguintes características: sem perda de dados, com abordagem horizontal e baseado em formatos de imagem. Para a fase de transformação, as técnicas e algoritmos de transformação do alfabeto genômico apresentadas são para exemplificar a execução desta fase. Outras técnicas e algoritmos existente na literatura podem ser utilizados nesta fase, no entanto é necessário um aprofundamento na pesquisa para avaliar o impacto de cada

uma delas em termos de economia de espaço.

Capítulo 5

Resultados

O procedimento para a avaliação de desempenho dessa proposta bascia-se na análise de sequências genômicas a serem codificadas para um dado formato de imagem. Os formatos de imagem testados foram os seguintes: *WebP*, *FLIF*. *Gif* e *Png*. Também foram testados os compressores de propósito geral *7zip* e *Gzip* para representarem os compressores da família *Zip*. Assim, o tamanho do arquivo comprimido resultante é dado pela soma da quantidade de *bytes* do arquivo de imagem com a quantidade de *bytes* do tamanho do arquivo do *codebook*. A configuração computacional utilizada nos testes foi formada pelo seguinte conjunto de características: *INTEL Xeon* 24 Cores de $2.40GHz$ ($64bits$) e $182GB$ de memória RAM; e ambiente operacional *Linux CentOS versão 6.0*.

5.1 Análise dos cenários

O conjunto de dados utilizado para a avaliação do desempenho é um subconjunto do *dataset* definido por (BIJI; NAIR, 2016), composto por 1.547 sequências. Não foram utilizadas sequências genômicas de comprimento maiores que 2^{28} bases nitrogenadas. Essa limitação, embora pode ser contornada dividindo a sequência em mais de um arquivo, ocorre por causa do formato de imagem *WebP*. Na versão utilizada, o formato *WebP* não permite imagens com a matriz de píxeis maior que 2^{28} píxeis. Para avaliar o impacto na divisão de uma sequência genômica em mais de um arquivo foi realizado um teste com uma sequência genômica, gerada artificialmente, de 16.777.216 símbolos. Para compor cada base nitrogenada da sequência foi escolhido um dos símbolos do alfabeto genômico {A, T, C, G} randomicamente.

A Tabela 5.1 mostra o impacto, em termos de tamanho final dos arquivos comprimidos, quando uma sequência genômica é dividida em partes, e cada parte distinta é comprimida com formato de arquivo de imagem. Embora o formato *FLIF* não impõe res-

Tabela 5.1: Porcentagem de aumento no tamanho final dos arquivos comprimidos para uma sequência de 16.777.216 símbolos que foi dividida em partes iguais e cada parte comprimida com os formatos *WebP* e *FLIF*.

	WEBP			FLIF		
# Partes	**Bytes**	**Soma**	**Aumento (%)**	**Bytes**	**Soma**	**Aumento (%)**
1	4194370	4194370	0.00	4290855	4290855	0.00
2	2129756	4259512	1.53	2145792	4291584	0.02
4	1048642	4194568	0.00	1072757	4291028	0.00
8	532470	4259760	1.54	536620	4292960	0.05
16	262210	4195360	0.02	268233	4291728	0.02
32	133156	4260992	1.56	134289	4297248	0.15
64	65602	4198528	0.10	67123	4295872	0.12
128	33328	4265984	1.68	33707	4314496	0.55
256	16454	4212224	0.42	16835	4309760	0.44
512	8496	4349952	3.58	8544	4374528	1.91
1024	4176	4276224	1.91	4256	4358144	1.54
2048	2204	4513792	7.08	2159	4421632	2.96
4096	1096	4489216	6.57	1085	4444160	3.45
8192	614	5029888	16.61	575	4710400	8.91
16384	324	5308416	20.99	292	4784128	10.31
32768	208	6815744	38.46	180	5898240	27.25
65536	134	8781824	52.24	88	5767168	25.60
131072	98	12845056	67.35	63	8257536	48.04
262144	78	20447232	79.49	39	10223616	58.03
524288	64	33554432	87.50	34	17825792	75.93
1048576	54	56623104	92.59	27	28311552	84.84
2097152	56	117440512	96.43	24	50331648	91.47
4194304	38	159383552	97.37	20	83886080	94.88
8388608	38	318767104	98.68	15	125829120	96.59

trição quanto a altura e comprimento da imagem para a compressão, o mesmo foi incluído no teste para comparação. A coluna "*Bytes*"mostra o tamanho, em *bytes*, de cada parte, enquanto a coluna "Soma"mostra a soma das partes, em *bytes*, de cada parte. Como pode ser observado, para o formato *WebP* no caso de ter que dividir a sequência genômica em até 256 partes iguais, a soma final de todas as partes comprimidas com formato de imagem teria uma perda menor que 0,5% em termos de tamanho final. Acima de 256 partes existe perda acima de 1% em termos de compressão quando somada as partes, fazendo com que o formato *WebP* fique em desvantagem frente ao formato *FLIF* que não possui limitação de tamanho de imagem.

O conjunto final de dados utilizados na avaliação consiste no genoma de 1.163 organismos, variando dos seguintes reinos biológicos: *Animalia*, *Archaea*, *Bacteria*, *Fungi*, *Plant* e *Protist* (ver Tabela 4.1). Como em alguns casos o genoma de um organismo é dividido em mais de um arquivo no formato *FASTA*, na contagem final o conjunto de

arquivos utilizados neste experimento foi de 1.547 arquivos distintos.

Os resultados obtidos utilizando os formatos de imagem foram comparados com os resultados das compressões das seguintes ferramentas especializadas para compressão de dados genômicos: *DELIMINATE* (MOHAMMED et al., 2012) e *MFCOMPRESS* (PINHO; PRATAS, 2014). Para analisar os resultados foi utilizado como métrica a taxa de economia de espaço *Space Savings (SS)*, dada pela Equação 5.1.

$$SS = \left(1 - \frac{CFS}{UFS}\right) \times 100 \tag{5.1}$$

onde *Uncompressed File Size (UFS)* é o tamanho em *bytes* do arquivo descomprimido e *Compressed File Size (CFS)* é o tamanho em *bytes* do arquivo comprimido.

Em seguida, são apresentados os resultados da avaliação de desempenho do método proposto dividido em duas partes: (i) o método aplicado sobre as sequências genômicas sem a fase de transformação dos dados; (ii) o método aplicado sobre as sequências genômicas com a fase de transformação inclusa.

5.2 Resultados Sem Fase de Transformação

Após testes de compressão com formato de arquivo de imagem (discutido a seguir), para os formatos *WebP* e *FLIF*, foi observado que o uso de algumas cores do sistema RGB, que estão mais próximas da cor preta, impactaram no tamanho final do arquivo comprimido. Isso ocorre porque os formatos *WebP* e *FLIF* implementam métodos de otimização para economia de espaço, conforme documentação citada no Capítulo 3. Dessa forma, foram testadas duas configurações para a compressão com imagem das 1547 sequências do *dataset*.

Na primeira configuração, os símbolos do alfabeto genômico {A, T, C, G} foram convertidos para sistema RGB(Red, Green, Blue) usando as cores azul RGB(0. 0. 255) para a base A, verde RGB(0, 255, 0) para a base T, vermelho RGB(255, 0, 0) para a base C e branca RGB(255, 255, 255) para a base G (cf. Tabela 5.2). Para as bases nitrogenadas A, T e C foram elevados os valores de uma das cores do sistema RGB até o seu valor máximo (255) e então para a base nitrogenada G foram elevados os valores de todas as cores até 255.

Na segunda configuração, foram utilizadas variações de cinza, com apenas o valor da cor vermelha (*Red*) do sistema RGB variando entre 0 e 3 sendo RGB(0, 0, 0) para a base A, RGB(1, 0, 0) para a base T, RGB(2, 0, 0) para a base C e RGB(3, 0, 0) para a base G, para a compressão das sequências, livres de símbolos não-ATCG. Nessa

Tabela 5.2: Configurações de cores do sistema RGB para cada base nitrogenada utilizadas nos testes compressão.

Bases	Colorido	Variações de Cinza
A	RGB(255, 0, 0)	RGB(0, 0, 0)
T	RGB(0, 255, 0)	RGB(1, 0, 0)
C	RGB(0, 0, 255)	RGB(2, 0, 0)
G	RGB(255, 255, 255)	RGB(3, 0, 0)

configuração, os valores das cores verde e azul foram mantidos igual a 0. Após testes com as duas configurações, foi observado que a configuração com variação de cinza obteve uma taxa média economia de espaço melhor, em média 1,03% para *WebP* e 0,02% para *FLIF* quando comparada com a configuração em cores (cf. Figura 5.1).

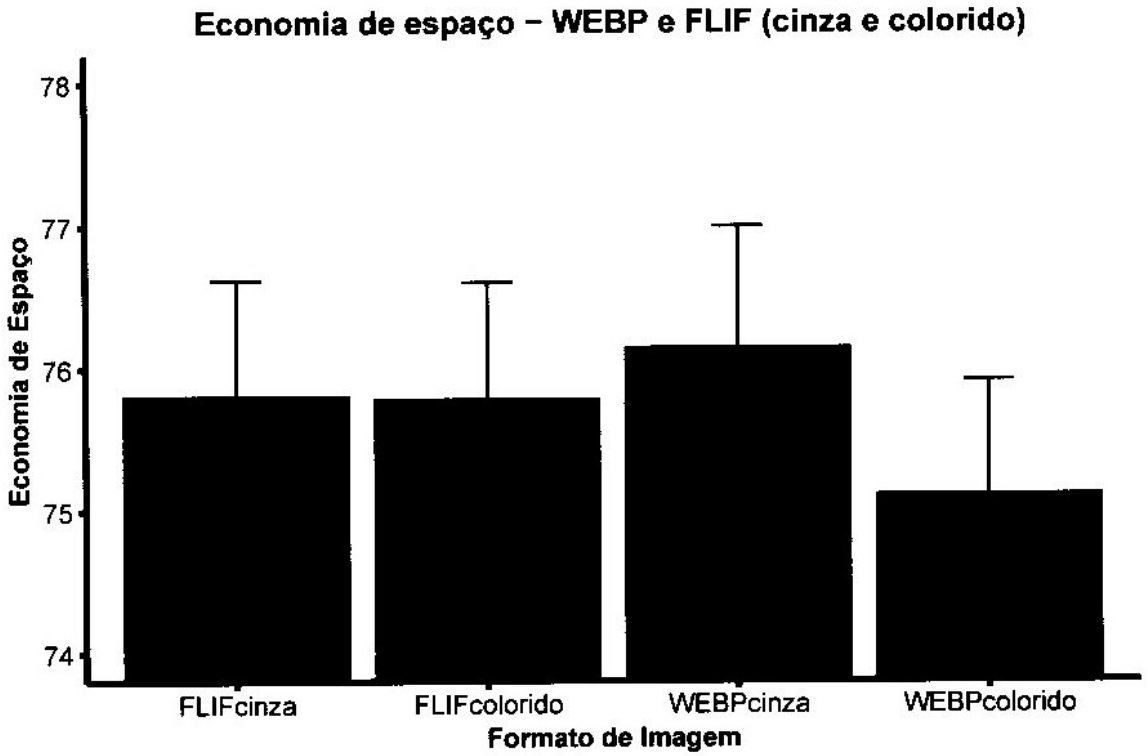

Figura 5.1: Taxa média de economia de espaço obtida por método utilizando variações de cinza e colorido.

Com base nos resultados da Figura 5.1, para a realização dos testes das compressões a partir daqui, o esforço foi concentrado na configuração com variações de cinza.

A Figura 5.2 apresenta a taxa média de economia de espaço obtida pelas ferramentas especializadas de compressão de dados genômicos e os formatos de imagem. Também está incluso entre os métodos de compressão a ferramenta de compressão de propósito geral 7-zip. O gráfico (cf. Figura 5.2) está ordenado pela taxa média de economia de

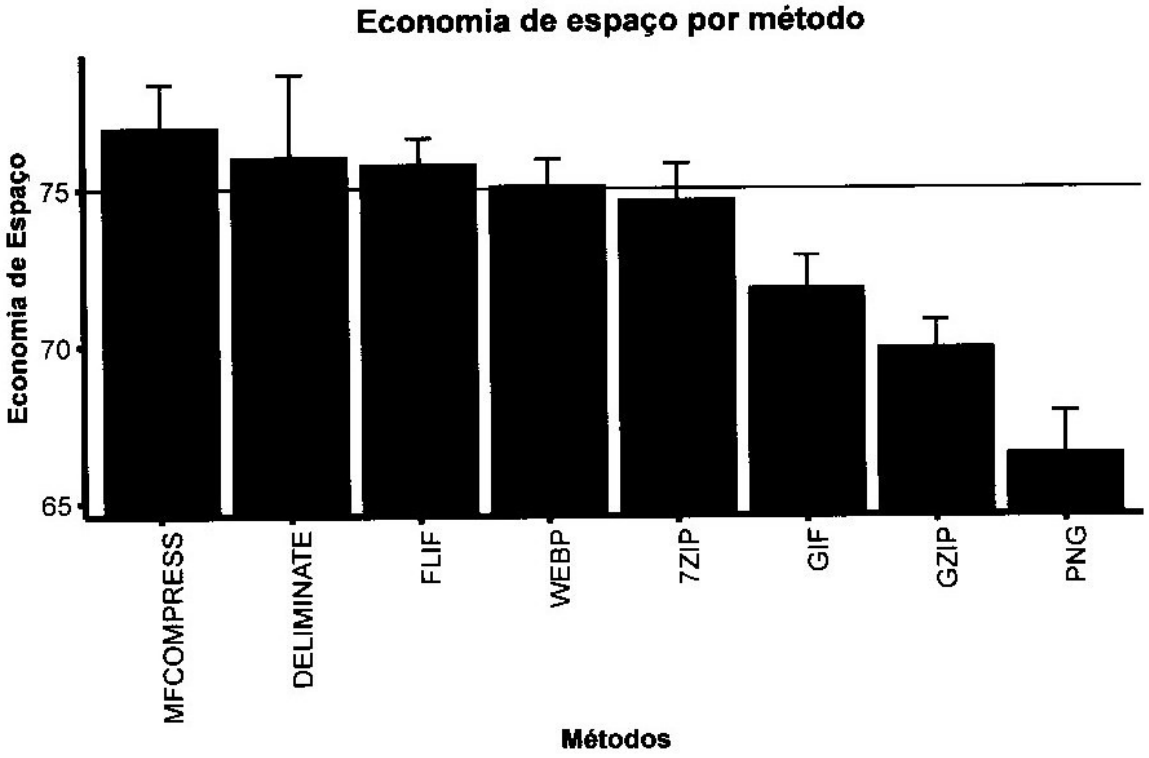

Figura 5.2: Taxa média de economia de espaço obtida pelos métodos de compressão MFCOMPRESS, DELIMINATE, FLIF, WEBP, PNG, GIF, 7ZIP e GZIP.

espaço obtida em ordem decrescente. No cálculo da média foi considerada economia de espaço de todos os 1.547 arquivos de sequências genômicas.

Ainda, na mesma figura foi adicionada uma linha *baseline* com o valor fixado em 75% de taxa de economia de espaço. Isso se deve ao fato de que a compressão do formato *FASTA* só ocorre, realmente, se a taxa de economia de espaço ficar acima de 75%. De outro modo, a codificação com *Naïve-bit encoding*, ou seja, a simples substituição de cada símbolo do alfabeto genômico {A, T, C, G} por dois *bits* já produz uma taxa de economia de espaço de 75%. Isso demonstra que qualquer esforço para construir algoritmos e técnicas de compressão de sequências genômicas que não produzam resultados acima de 75% é esforço em vão. Como pode ser observado na Figura 5.2 os métodos que ultrapassaram 75% de taxa de economia de espaço foram as ferramentas especializadas *MFCOMPRESS* e *DELIMINATE* e também os formatos de arquivo de imagem *FLIF* e *WebP*.

A Figura 5.3 apresenta a taxa média de economia de espaço agrupando as sequências genômicas pelo reino de cada organismo biológico. Como pode ser observado a ferramenta especializada *MFCOMPRESS* obteve resultados acima da *baseline*, ou seja, de 75% para todos os reinos. Já a ferramenta especializada *DELIMINATE* ficou abaixo de 75% apenas nas compressões das sequências genômicas do reino *Archaea*. Quanto aos for-

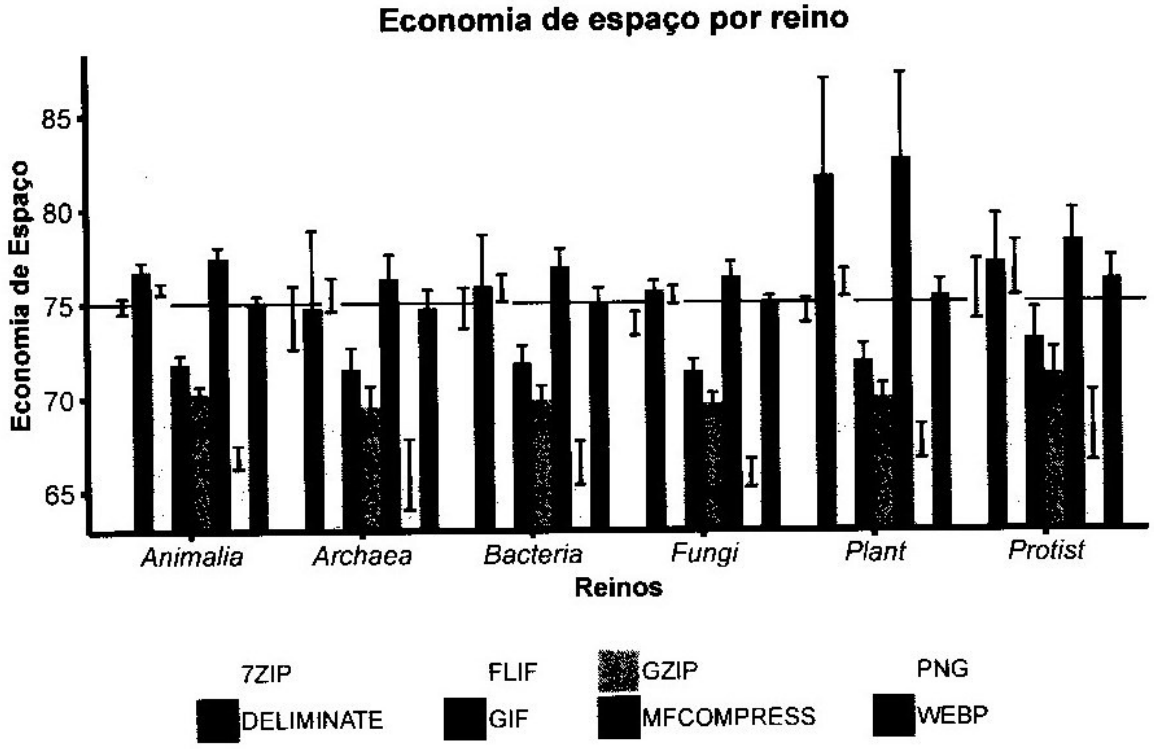

Figura 5.3: Taxa média de economia de espaço classificada por reino biológico.

matos de arquivo de imagem *FLIF* superou a marca de 75% de taxa média de economia de espaço em todos reinos enquanto o formato *WebP* teve economia acima de 75% apenas para os reinos *Plant* e *Protist*.

Baseado nos resultados das taxas médias de economia de espaço (cf. Figura 5.2), a partir daqui será avaliado apenas os formatos de imagem bem como as ferramentas especializadas que atingiram a média maior que 75% em pelo menos 2 reinos biológicos. Serão avaliados a seguir apenas os métodos: *MFCOMPRESS*, *DELIMINATE*. *FLIF* e *WebP*. Embora o compressor de propósito geral *7-zip* tenha obtido mais de 75% de taxa média de economia de espaço em pelo menos um dos reinos biológicos, esse formato não será avaliado.

Tabela 5.3: Taxa média de economia de espaço obtida pelos 4 melhores métodos na compressão sem a fase de transformação.

Métodos	*Animalia* (%)	*Bacteria* (%)	*Archaea* (%)	*Fungi* (%)	*Plant* (%)	*Protist* (%)	**Média geral** (%)
MFCOMPRESS *	77.43 ±0.56	76.92 ±0.97	76.35 ±1.22	76.33 ±0.80	81.69 ±4.60	78.26 ±1.69	76.97 ±2.54
WebP **	76.09 ±0.32	76.14 ±0.76	75.86 ±1.00	75.78 ±0.49	76.23 ±0.72	77.20 ±1.36	76.15 ±0.84
DELIMINATE *	76.75 ±0.45	75.94 ±2.63	74.90 ±3.90	75.64 ±0.50	80.89 ±5.02	77.17 ±2.49	76.03 ±2.54
FLIF **	75.76 ±0.31	75.81 ±0.69	75.58 ±0.72	75.42 ±0.48	75.99 ±0.65	76.85 ±1.43	75.81 ±0.80
Média por reino	76.50 ±0.63	76.20 ±0.43	75.67 ±0.52	75.79 ±0.33	78.69 ±2.90	77.36 ±0.53	66.54 ±1.29

*Ferramentas. **Formatos de imagem - (Valores são Média e Desvio padrão)

A Tabela 5.3 mostra o resultado obtido das compressões quando comparado so-

mente com as ferramentas especializadas de compressão de dados genômicos avaliadas e também com os dois formatos de imagem que atingiram taxas médias de economia de espaço acima de 75% em pelo menos dois reinos. A taxa média de economia de espaço está seguida pelo valor do desvio padrão. Como pode ser observado, a tabela mostra na última coluna (à direita) a taxa média "geral"de economia de espaço obtida por cada método. A ferramenta especializada *MFCOMPRESS* apresentou a melhor taxa média com percentual de 76,97%, seguida do formato de imagem *WebP* com percentual de 76,15%.

Tabela 5.4: Soma dos tamanhos em *megabytes* das sequências genômicas e das taxas médias de economia de espaço.

Reinos	Tamanho**	**DELIMINATE***		**MFCOMPRESS***		**WEBP***		**FLIF***	
Animalia	262.36	61.13	76.70	59.57	77.29	62.82	76.06	63.71	75.72
Archaea	41.35	9.92	76.01	9.65	76.67	9.88	76.10	10.04	75.72
Bacteria	3.900.30	929.50	76.17	898.60	76.96	928.65	76.19	942.33	75.84
Fungi	473.28	115.12	75.68	111.36	76.47	115.18	75.66	116.74	75.33
Plant	2.007.95	290.75	85.52	284.67	85.82	486.68	75.76	490.04	75.60
Protist	135.87	29.97	77.94	28.70	78.88	30.27	77.72	30.64	77.45
Total	**6.821,11**	**1.436,40**	**78,94**	**1.392,55**	**79,58**	**1.633,48**	**76,05**	**1.653,50**	**75,76**

** Tamanho em MB da sequência original. * Método (Tamanho em MB e Taxa média de economia de espaço)

Para simular um caso real de compressão, sem perda de dados e com abordagem horizontal, de um banco de dados genético utilizando os formatos de imagem a Tabela 5.4 apresenta a soma dos tamanhos em *megabytes* das sequências genômicas antes da compressão bem como a soma dos tamanhos pós compressão. Também é apresentado a taxa média de economia de espaço calculada para cada reino. De acordo com os resultados as ferramentas especializadas *DELIMINATE* e *MFCOMPRESS* obtiveram as taxas médias mais altas, principalmente para o reino *Plant*. Isso ocorreu porque essas ferramentas lidam melhor com trechos repetidos existentes na sequência genômica. Os resultados também deixam claros que o método de compressão baseado em formato de arquivo de imagem precisa ainda de melhorias. Em um cenário onde é necessário comprimir várias sequências genômicas de vários reinos distintos os resultados das compressões não seriam eficientes com taxa média de economia de espaço pouco acima de 75%.

Para a validação dos resultados, foram aplicados testes não paramétricos, *Friedman* e *Nemenyi*. O teste de *Friedman* (FRIEDMAN, 1937) é um teste de experimentos em blocos ao acaso equivalente a *ANOVA* para medidas repetidas. Este teste classifica os algoritmos para cada conjunto de dados separadamente com o objetivo de dizer se variações são possivelmente diferentes de população para população. Este teste faz um *rank* dos dados e depois utiliza-os ao invés usar os seus próprios valores brutos para o cálculo da estatística. Como o teste de *Friedman* não faz suposições sobre a distribuição, ele não é tão poderoso para dizer se as populações forem realmente normais.

Com o resultado obtido pelo teste de *Friedman* foi possível rejeitar a hipótese nula (quando o valor p fica acima de $0,05$) uma vez que o valor p foi inferior a $0,01$ ($p < 0,01$) para 1.547 observações e 4 métodos distintos (cf. Tabela 5.3). Assim, foi necessário aplicar um teste pós-avaliação para identificar quais métodos geraram resultados diferenciados. O teste *Nemenyi* (NEMENYI, 1962) é usado quando todos os métodos são comparados em uma base *peer-by-peer* (par a par). O desempenho de dois métodos é significativamente diferente se as taxas médias de economia de espaço correspondentes diferem pelo menos da diferença crítica.

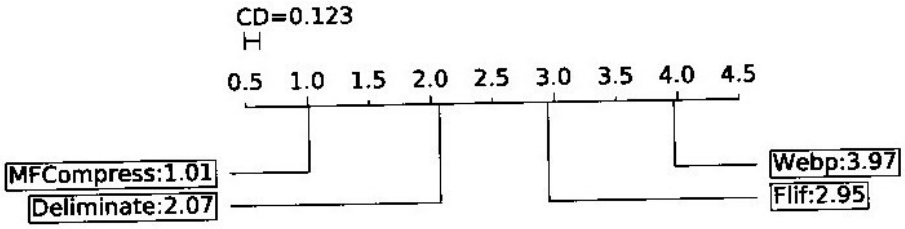

Figura 5.4: Teste de Nemenyi.

A Figura 5.4 apresenta o teste de *Friedman* e *Nemenyi* considerando a taxa média de economia de espaço obtida pelos métodos na compressão individual de todos os 1.547 arquivos do formato *FASTA*. O conjunto de testes mostrou que a diferença crítica é de $0,123$. Assim, nenhum formato de imagem ou ferramenta especializadas são estatisticamente semelhantes de acordo com o o resultado do teste de *Nemenyi*.

Após a avaliação do desempenho dos métodos, foi realizada uma análise para comparar a taxa média de economia de espaço dos métodos com as características das sequências genômicas submetidas à compressão com imagem. Esta investigação teve como objetivo avaliar em quais situações se destaca cada método, abordando as seguintes características: o reino, o tamanho do arquivo, em *bytes*, no formato *FASTA* antes da compressão, a entropia da informação e o índice de repetitividade.

5.2.1 Taxa média de economia de espaço e o reino biológico

Para a primeira análise comparativa foi avaliada a taxa média de economia de espaço obtida pelos métodos em relação ao reino de cada organismo. A distribuição dos reinos e organismos foi feita de acordo com a coluna Média na Tabela 5.3. A Figura 5.3 mostra a taxa média de economia de espaço agrupada por reino (cf. valores detalhados na Tabela 5.3). É possível observar que, na comparação entre os métodos *DELIMINATE*,

MFCOMPRESS, *WebP* e *FLIF*, o reino *Protist* apresentou os melhores resultados de compressão com a taxa média de economia de espaço de 77,36% e com o menor desvio padrão ($SD = 0,53$). No entanto, para o reino *Plant*, apesar da taxa média de economia de espaço estar em (78,69%), a variação foi maior ($SD = 2,90\%$). Os resultados mostram que o mesmo método pode ser melhor em termos de compressão para alguns reinos do que para outros reinos biológicos. Esse fato está intimamente ligado com as características biológicas de cada reino existentes nas sequências genômicas e o quanto cada método explora essas características com vistas a obter vantagem na compressão. Tais características podem ser, por exemplo, os palíndromos, as repetições de um determinado comprimento existentes ao longo da sequência genômica, ou mesmo as repetições por n vezes seguidas (avizinhadas) de uma determinada base nitrogenada.

5.2.2 Correlação entre a taxa média de economia de espaço e o tamanho da sequência

Para avaliar o impacto, positivo ou negativo, na taxa média de economia de espaço, esta foi avaliada em relação ao tamanho das sequências, sem passar pela fase de transformação dos dados. Depois de descartar o cabeçalho do formato *FASTA*, o tamanho de cada sequência foi calculado contando o número de símbolos, incluindo os símbolos não-ATCG. Para os testes de correlação foi utilizado o cálculo do coeficiente de correlação de *Pearson* (MUKAKA, 2012), cujo grau de correlação entre duas variáveis, que pode variar entre $+1$ e -1, é interpretado conforme Figura 5.5.

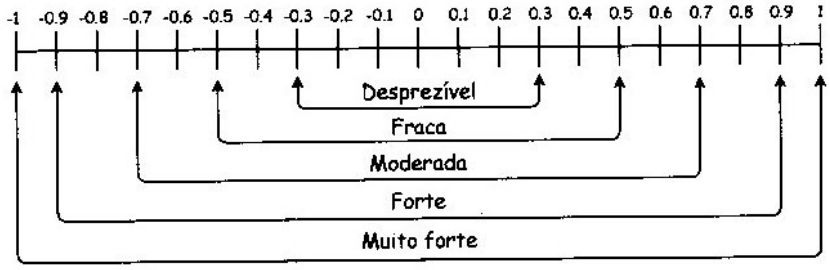

Figura 5.5: Grau de correlação de *Pearson*.
Fonte: Autor.

A Tabela 5.5 mostra a correlação entre a taxa média de economia de espaço e o tamanho, medido em *bytes*, das sequências genômicas. De acordo com os resultados é possível perceber uma forte correlação para o reino *Plant* (cf. linha destacada na Tabela 5.5). No entanto, essa correlação é positiva para as duas ferramentas especializadas de compressão genômica e negativas para os formatos de imagem. Isso ocorre porque os formatos de imagem não foram efetivos na compressão dos grandes genomas do reino

Tabela 5.5: Correlação de *Pearson* calculada comparando a taxa média de economia de espaço de cada método com o tamanho da sequência medido em *bytes*.

Reino	*DELIMINATE**	*MFCOMPRESS**	*WEBP***	*FLIF***
Animalia	-0.33	-0.71	-0.34	-0.40
Archaea	0.60	0.55	0.51	0.39
Bacteria	0.16	0.08	0.11	0.08
Fungi	0.05	0.16	-0.21	-0.16
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Protist	0.31	0.36	0.38	0.42

Correlação de *Pearson* – *Ferramentas especializadas e **Formatos de imagem

Plant. Isso ocorre porque a compressão com os formatos de arquivo de imagem, sem a fase de transformação, codifica cada símbolo existente na sequência genômica para um píxel. Quanto maior a sequência genômica menos efetivos, em termos de compressão foram os formatos de arquivo de imagem. Em geral, com a exceção do reino *Plant*, o tamanho da sequência não apresenta forte correlação para os formatos de imagem.

5.2.3 Correlação entre a taxa média de economia de espaço e o índice de repetitividade

Outra característica avaliada é o índice de repetitividade (IR). Conforme descrito pelos autores em (HAUBOLD; WIEHE, 2006), o índice de repetitividade é uma medida que tenta medir a quantidade de repetição intra-repetitiva presente em uma sequência de DNA. Espera-se que o IR seja zero na sequência de DNA aleatória de qualquer *G/C content* (percentual de G e C existente na sequência genômica). O valor mais alto é maior que zero para sequências que contém repetições expressivas.

Tabela 5.6: Correlação de *Pearson* calculada comparando a taxa média de economia de espaço de cada método com o Índice de Repetitividade.

Reino	*DELIMINATE**	*MFCOMPRESS**	*WEBP***	*FLIF***
Animalia	-0.08	0.26	-0.35	-0.31
Archaea	0.21	0.20	0.12	0.04
Bacteria	0.05	0.30	-0.08	-0.11
Fungi	0.11	0.45	-0.01	-0.14
Plant	-0.17	0.05	-0.42	-0.46
Protist	0.14	0.37	-0.03	-0.12

Correlação de *Pearson* – *Ferramentas especializadas e **Formatos de imagem

A Tabela 5.6 apresenta a correlação de *Pearson* entre a taxa média de economia de espaço de cada método com o Índice de Repetitividade. Os resultados desta medida não

apresentam forte ou moderada correlação com nenhum dos métodos testados conforme a Figura 5.5.

5.2.4 Correlação entre a taxa média de economia de espaço e a entropia da informação

A última característica avaliada foi a Entropia de Informação, calculada conforme definido por (SHANNON, 1948) para cada sequência genômica. Uma sequência aleatória restrita ao alfabeto genômico {A, T, C, G}, cujas frequências dos símbolos são iguais, tem o valor de entropia igual a 2, ou seja, são necessários 2 *bits* para representar cada símbolo da sequência de DNA. No caso de uma sequência de DNA, restrita ao alfabeto genômico {A, T, C, G}, quanto menor a entropia, mais desequilibrada é a frequência em que esses símbolos aparecem na sequência genômica. Isso pode ocorrer devido as características peculiares de cada sequência genômica e a forma com que as bases nitrogenadas estão organizadas.

Tabela 5.7: Correlação de *Pearson* calculada comparando a taxa média de economia de espaço de cada método com as entropia da informação.

Reino	*DELIMINATE**	*MFCOMPRESS**	*WEBP***	*FLIF***
Animalia	-0.34	-0.59	-0.09	-0.09
Archaea	-0.25	-0.54	-0.60	-0.80
Bacteria	-0.13	-0.40	-0.55	-0.66
Fungi	0.29	-0.06	-0.05	-0.15
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Protist	-0.33	-0.49	-0.61	-0.68

Correlação de *Pearson* – *Ferramentas especializadas e **Formatos de imagem

A taxa média de economia de espaço e a entropia foram utilizadas para calcular a correlação de *Pearson* (cf. Tabela 5.7). No caso das correlações entre a entropia e a taxa média de economia de espaço, quando o grau de correlação é negativo, significa que quanto menor a entropia da informação existente na sequência genômica maior a taxa média de economia de espaço. Existe grau de correlação moderado e positivo entre o reino *Plant* e os formatos de imagem, enquanto há uma correlação fraca e negativa para as ferramentas especializadas de compressão genômica (cf. linha destacada na Tabela 5.7). Esses resultados podem ser explicados pelo fato de que as ferramentas especializadas aplicam diferentes transformações sobre os dados antes da compressão. Assim, a fase de codificação dessas ferramentas não é baseada unicamente na distribuição original dos símbolos.

5.3 Resultados Com Fase de Transformação

O método de compressão de genomas baseado em formato de imagem também foi testado levando em conta a fase de transformação dos dados. O objetivo destes testes é observar o impacto que a fase de transformação causa no resultado final da taxa média de economia de espaço. Isso está ligado ao fato de que cada técnica de transformação proporciona um alfabeto diferente e uma organização peculiar da técnica sobre os símbolos para a sequência genômica.

A Figura 5.6 apresenta os resultados de todas as técnicas de transformação dos dados, abordadas nessa pesquisa, aplicadas sobre as sequências genômicas bem como os dois formatos de imagem aplicados sobre a sequência original.

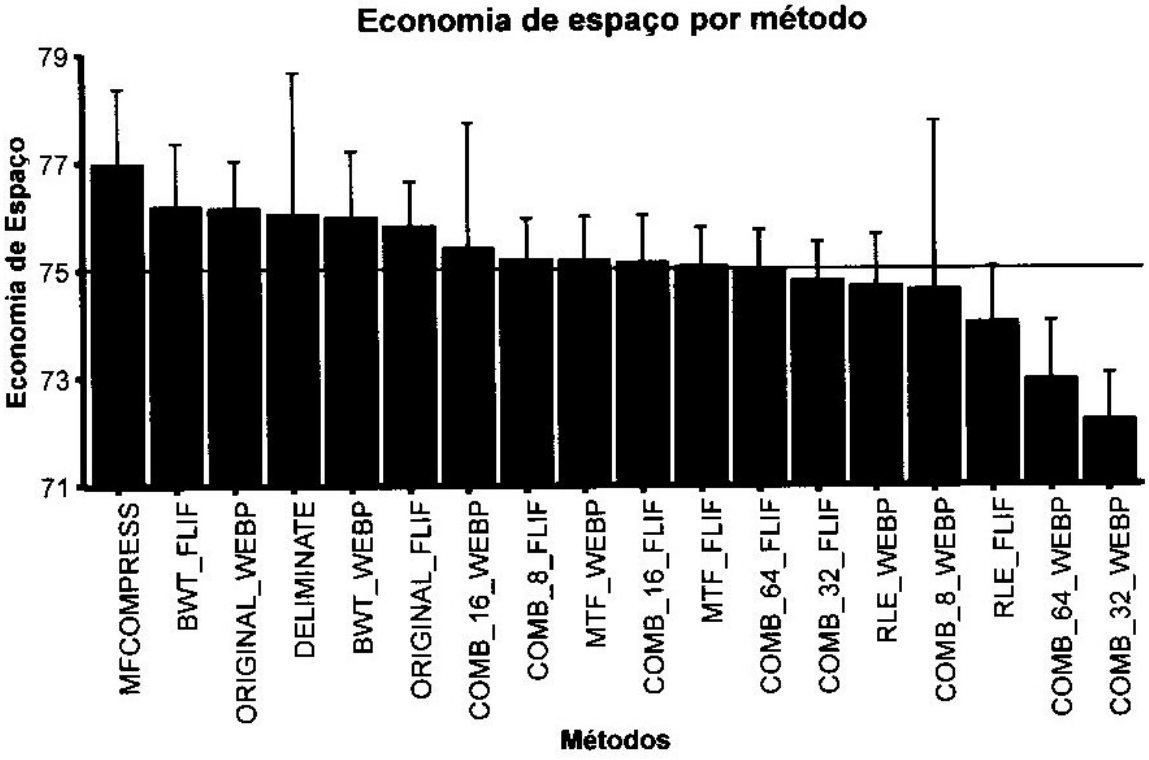

Figura 5.6: Taxa média de economia de espaço por método.

As técnicas de transformação do alfabeto genômico aplicadas foram: BWT, MTF, RLE e também os agrupamentos de combinações binárias para 8, 16, 32 e 64 combinações, em que cada combinação é substituído por uma cor na compressão com formato de arquivo de imagem. Para comparação, as duas ferramentas especializadas para compressão de genomas também estão inclusas. Os nomes que são apresentados no gráfico estão exibindo primeiramente o nome da técnica de transformação seguido do formato de imagem. Para melhor entendimento dos nomes e as técnicas aplicadas eles também estão descritos na Tabela 5.8.

Como pode ser observado na Figura 5.6, das compressões com formatos de imagem, 7 com a aplicação de técnicas de transformação e 2 sem técnicas de transformação aplicadas obtiveram taxas médias de economia de espaço acima da *baseline*, ou seja, acima de 75%. No entanto, em relação as ferramentas especializadas, apenas dois métodos, um com transformação (*BWT+FLIF*) e um sem transformação (*ORIGINAL + WEBP*) obtiveram taxas médias superiores a pelo menos uma das ferramentas especializadas.

Tabela 5.8: Discriminação dos nomes de métodos no gráfico da Figura 5.6 de acordo com a ferramenta, formato de imagem e tipo de transformação aplicada.

Nome	Método	Transformação
[illegible]	[illegible]	[illegible] Wheeler Transform
BWT_WEBP	WEBP	Burrows Wheeler Transform
ORIGINAL_FLIF	FLIF	Sem transformação
[illegible]	[illegible]	[illegible]
COMB_8_FLIF	FLIF	BitCombine 3 bits (8 cores)
COMB_16_FLIF	FLIF	BitCombine 4 bits (16 cores)
COMB_32_FLIF	FLIF	BitCombine 5 bits (32 cores)
COMB_64_FLIF	FLIF	BitCombine 6 bits (64 cores)
COMB_8_WEBP	WEBP	BitCombine 3 bits (8 cores)
COMB_16_WEBP	WEBP	BitCombine 4 bits (16 cores)
COMB_32_WEBP	WEBP	BitCombine 5 bits (32 cores)
COMB_64_WEBP	WEBP	BitCombine 6 bits (64 cores)
MTF_FLIF	FLIF	Move To Front
MTF_WEBP	WEBP	Move To Front
RLE_FLIF	FLIF	Run Length Encode
RLE_WEBP	WEBP	Run Length Encode

Como pode ser observado, das compressões com formatos de imagem, 7 com a aplicação de técnicas de transformação e 2 sem técnicas de transformação aplicadas obtiveram taxas médias de economia de espaço acima da *baseline*, ou seja, acima de 75%. No entanto, em relação as ferramentas especializadas, apenas dois métodos, um com transformação (*BWT+FLIF*) e um sem transformação (*ORIGINAL+WEBP*) obtiveram taxas médias superiores a pelo menos uma das ferramentas especializadas.

A partir daqui serão avaliadas somente os dois métodos que obtiveram taxas médias superiores a pelo menos uma das ferramentas especializadas. Isso se deve ao fato de que a pesquisa sobre as técnicas de transformação ainda precisa ser aprofundada, principalmente porque a aplicação de certas técnicas em conjunto com outras e certa ordem podem produzir melhores resultados do que a aplicação isolada das mesmas.

A Figura 5.7 mostra a taxa média de economia de espaço classificada pelo reino

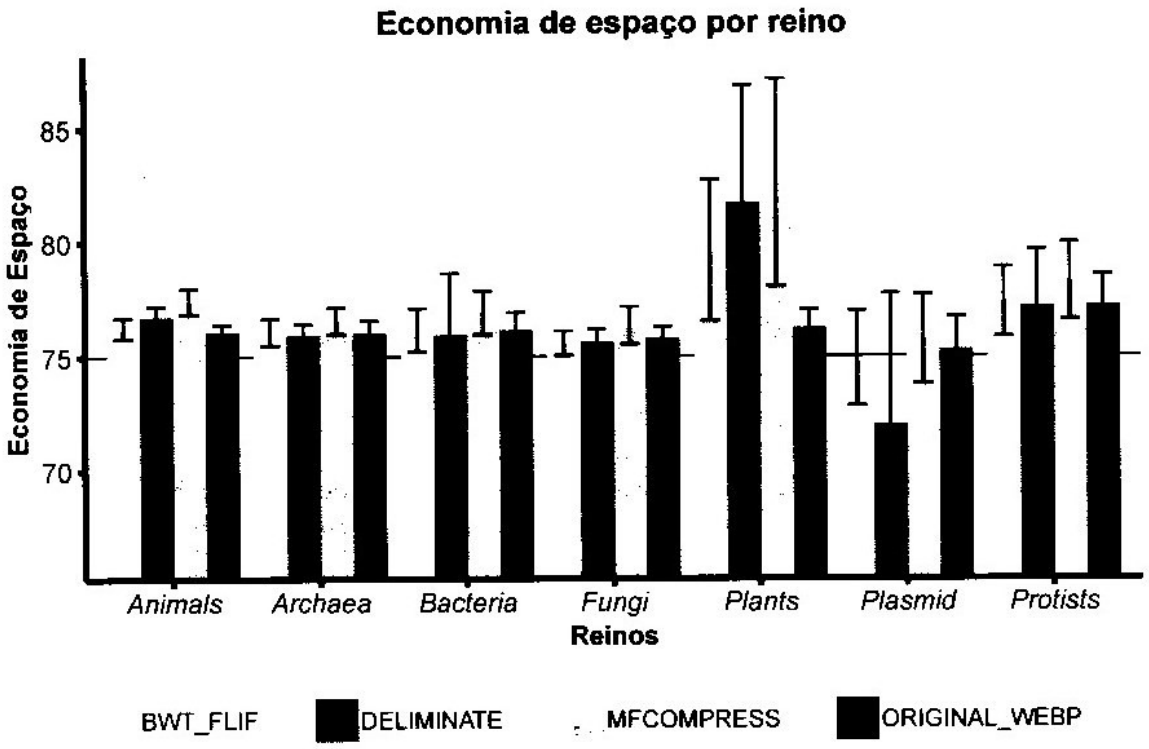

Figura 5.7: Taxa média de economia de espaço por reino biológico dos 4 melhores métodos na média geral.

biológico dos organismos. Como pode ser observado. as melhores taxas médias de economia de espaço foram para o reino *Plant*. Como mencionado anteriormente, os genomas desse reino possuem trechos de repetições longos cujas ferramentas especializadas exploram. Também, a técnica de transformação BWT em conjunto com o formato de imagem *FLIF* (*BWT+FLIF*) obteve melhora significativa na taxa média, em torno de 3,63%. em comparação com a sequência sem transformação para o reino *Plant*. Isso se deve ao fato de que a técnica de transformação BWT faz um rearranjo na sequência proporcionando que os símbolos iguais fiquem próximos uns dos outros. Dessa forma, os formatos de imagem utilizados para a compressão conseguem lidar melhor quando os píxeis vizinhos são repetidos.

Tabela 5.9: Taxa média de economia de espaço obtida pelos 4 melhores métodos na compressão com a fase de transformação inclusa.

Métodos	*Animalia* (%)	*Bacteria* (%)	*Archaea* (%)	*Fungi* (%)	*Plant* (%)	*Protist* (%)	**Média geral** (%)
[redacted]							
*ORIGINAL - WEBP***	76.13 ±0.30	76.15 ±0.77	76.04 ±0.52	75.80 ±0.50	76.24 ±0.78	77.20 ±1.36	76.17 ±0.85
DELIMINATE *	76.81 ±0.41	75.95 ±2.69	75.94 ±0.47	75.65 ±0.52	81.73 ±5.02	77.17 ±2.49	76.07 ±2.56
BWT - FLIF **	76.30 ±0.41	76.17 ±0.94	76.08 ±0.57	75.57 ±0.54	79.62 ±3.01	77.36 ±1.65	76.20 ±1.14
Média por reino	76.68 ±0.40	76.30 ±1.34	76.16 ±0.53	75.84 ±0.59	[redacted]	77.50 ±1.76	77.09 ±1.32

*Ferramentas, **Formatos de imagem. (Valores são Média e Desvio padrão)

A Tabela 5.9 mostra o resultado obtido das compressões quando comparado com

as ferramentas especializadas de compressão de dados genômicos avaliadas e também com os dois formatos de imagem que obtiveram resultados melhores em termos de compressão que pelo menos uma ferramenta especializada. A taxa média de economia de espaço está seguida pelo valor do desvio padrão. Como pode ser observado, a tabela mostra na última coluna (à direita) a taxa média "geral"de economia de espaço obtida por cada método. A ferramenta especializada *MFCOMPRESS* apresentou a melhor taxa média com um percentual de 77,00%, seguida do formato de imagem *BWT+FLIF* com um percentual de 76,20%.

Tabela 5.10: Soma dos tamanhos em *megabytes* das sequências genômicas e das taxas médias de economia de espaço quando aplicada a fase de transformação dos dados.

Reinos	Tamanho*	DELIMINATE**		MFCOMPRESS*		ORIGINAL−WEBP*		BWT+FLIF*	
Animalia	262.36	61,13	76.70	59.57	77.29	62.82	76.06	62.42	76.21
Archaea	41.35	9.92	76.01	9.65	76.67	9.93	75.98	9.91	76.04
Bacteria	3.900.30	929.50	76.17	898.60	76.96	928.21	76.20	924.75	76.29
Fungi	473.28	115.12	75.68	111.36	76.47	115.14	75.67	115.86	75.52
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Protist	135.87	29.97	77.94	28.70	78.88	30.27	77.72	29.79	78.08
Total	**6.821,11**	**1.436,40**	**78,94**	**1.392,55**	**79,58**	**1.633,43**	**76,05**	**1.501,50**	**77,99**

* Tamanho (MB) Sequência original. ** Método (Tamanho em MB e Taxa média de economia de espaço).

A Tabela 5.10 apresenta a soma dos tamanhos em *megabytes* das sequências genômicas, bem como dos tamanhos pós compressão. Também é apresentada a taxa média de economia de espaço calculada por reino. De acordo com os resultados da tabela, as ferramentas especializadas *MFCOMPRESS* e *DELIMINATE* obtiveram os melhores resultados. No entanto a transformação BWT com compressão em formato de imagem *FLIF* obteve resultados bem próximos (82,13%) para o reino *Plant*.

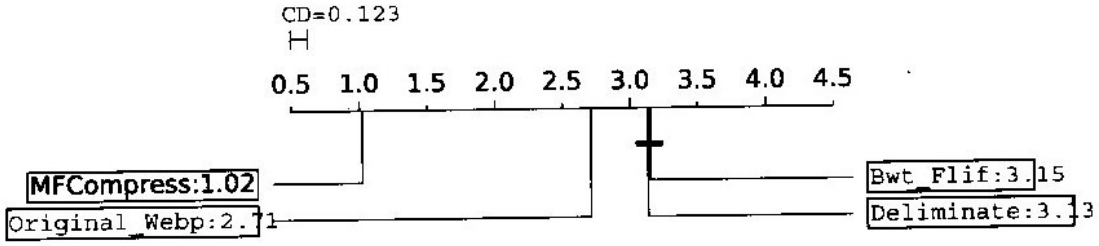

Figura 5.8: Diferença crítica entre os métodos com fase de transformação.

A Figura 5.8 mostra o teste de *Friedman* e *Nemenyi* aplicados sobre os 4 métodos de compressão. Com o resultado obtido pelo teste de *Friedman*, mais uma vez foi possível

rejeitar a hipótese nula, sendo que o valor p ficou inferior a $0,01$ ($p < 0,01$) para as 1.547 observações entre os 4 métodos distintos. Então, foi aplicado o teste de *Nemenyi* para observar se haveria diferença crítica entre os métodos. De acordo com o resultado a diferença crítica ficou em 0,123 mostrando que os métodos *DELIMINATE* e transformação BWT com formato de imagem *FLIF* são estatisticamente similares.

5.3.1 Taxa média de economia de espaço e o reino biológico

Para a análise comparativa com a fase de transformação dos dados inclusa foi avaliado a taxa média de economia de espaço obtida pelos métodos em relação ao reino de cada organismo. A Figura 5.7 mostra a taxa média de economia de espaço agrupada por reino. É possível observar na Tabela 5.9 que na comparação entre os métodos *DELIMINATE*, *MFCOMPRESS*, *ORIGINAL−WEBP* e *BWT+FLIF* o reino *Animalia* apresentou os resultados de compressão mais próximo da média 76.68% com o desvio padrão ($SD = 0,40$). No entanto, para o reino *Plant*, apesar da taxa média de economia de espaço estar em $(80,05\%)$, a variação foi maior ($SD = 3.31\%$) (cf. Tabela 5.9). Mesmo com a fase de transformação inclusa os resultados continuam mostrando que um método pode ser melhor em termos de compressão para alguns reinos e ser ineficiente para outros.

5.3.2 Correlação entre a taxa média de economia de espaço e o tamanho da sequência

A Tabela 5.11 mostra o grau de correlação de *Pearson* calculada para taxa média de economia de espaço com o tamanho da sequência. De acordo com a tabela é possível perceber uma forte correlação para o reino *Plant*. No entanto, essa correlação é negativa para a compressão da sequência original com formato de imagem *WebP*. Isso ocorre porque o formato de imagem *WebP* não foi efetivo na compressão dos grandes genomas do reino *Plant* sem a aplicação de transformação do alfabeto genômico. No caso da compressão com a sequência transformada utilizando a técnica BWT juntamente com o formato de imagem *FLIF* o rearranjo que a técnica proporcionou nos símbolos da sequência favoreceu a compressão. Conforme os resultados da Tabela 5.11, quanto maior a sequência genômica melhor é a taxa média de economia de espaço com a compressão de *BWT−FLIF* para o reino *Plant*.

Tabela 5.11: Correlação de *Pearson* calculada para taxa média de economia de espaço com o tamanho da sequência.

Reino	*MFCOMPRESS**	*DELIMINATE**	*BWT+FLIF***	*ORIGINAL+WEBP***
Animalia	-0.71	-0.33	-0.31	-0.35
Archaea	-0.26	-0.26	-0.21	-0.30
Bacteria	0.09	0.16	0.23	0.12
Fungi	0.17	0.06	-0.09	-0.22
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Protist	0.37	0.31	0.47	0.38

Correlação de *Pearson* - * Ferramenta especializada e ** Formato de imagem

5.3.3 Correlação entre a taxa média de economia de espaço e a entropia da informação

A tabela Tabela 5.12 apresenta a correlação de *Pearson* calculada para a entropia da informação. De acordo com a tabela é possível perceber uma correlação forte e positiva para o reino *Plant* apenas para as duas ferramentas especializadas e também para o formato *FLIF* quando aplicada a transformação BWT. Ainda para o reino *Plant* o grau de correlação é negativo apenas para a compressão da sequência original (sem transformação) com formato de imagem *WebP*.

Tabela 5.12: Correlação de *Pearson* calculada para taxa média de economia de espaço com entropia da informação.

Reino	*MFCOMPRESS**	*DELIMINATE**	*BWT+FLIF***	*ORIGINAL+WEBP***
Animalia	-0.88	-0.74	-0.61	-0.65
Archaea	-0.95	-0.98	-0.91	-0.98
Bacteria	-0.52	-0.19	-0.63	-0.73
Fungi	-0.33	-0.14	-0.67	-0.72
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Protist	-0.64	-0.48	-0.81	-0.85

Correlação de *Pearson* - * Ferramenta especializada e ** Formato de imagem

Baseado nos resultados da Tabela 5.12 é possível afirmar, com ressalvas, que quanto maior e entropia maior também é a taxa média de economia de espaço para *MFCOMPRESS*, *DELIMINATE* e *BWT + FLIF*. Porém, a correlação também está sendo influenciada, para o reino *Plant*, por causa do tamanho da sequência, pois, mesmo com a entropia aumentando a taxa média de economia de espaço também aumenta, pois os tamanhos das sequências também aumentam os tamanhos em *bytes*. Para o restante dos casos a entropia influenciou corretamente a taxa média de economia de espaço para maioria dos métodos. Somente a ferramenta DELIMINATE teve grau de correlação desprezível para os reinos *Bacteria* e *Fungi*.

5.4 Análise sem e com fase transformação

Quando é comparado a soma dos resultados da aplicação da fase de transformação dos dados e a não aplicação desta (cf. Tabela 5.4 e Tabela 5.10), é possível perceber que preparar (transformar) os dados antes da codificação permite aos formatos de arquivo de imagem obterem vantagem, em termos de compressão, quanto ao tamanho do arquivo final. Tal vantagem é demonstrada na Tabela 5.13 que apresenta os resultados de compressão somando os tamanhos de todas as sequências do *dataset* utilizado nesse trabalho. Os resultados da coluna do formato *FLIF*, quando aplicada transformação dos dados, teve ganho de mais de 2% na taxa de economia de espaço, com a utilização da técnica de transformação BWT antes de codificar com formato de arquivo de imagem.

Tabela 5.13: Comparativo da taxa de economia de espaço com e sem a fase de transformação.

Transformação	Tamanho**	**DELIMINATE***		**MFCOMPRESS***		**WEBP***		**FLIF***	
Não	6.821,11	1.436,40	78,94	1.392,55	79,58	1.633,48	76,05	1.653,50	75,76
Sim para FLIF	6.821,11	1.436,40	78,94	1.392,55	79,58	1.633,43	76,05	[redacted]	

** Tamanho em MB da sequência original. * Método (Tamanho em MB e Taxa média de economia de espaço)

Os resultados das compressões com formato de arquivo de imagem comprovam as hipóteses de que: (i) comprimir com formato de arquivo de imagem é um método viável para a compressão de sequências genômicas; (ii) Aplicar técnicas de transformação de dados antes da codificação com formato de arquivo de imagem melhora a taxa de economia de espaço.

5.5 Considerações finais

O Capítulo 5 apresentou os resultados obtidos para as compressões com os formatos de imagem *WebP*, *FLIF*, *Gif* e *Png* e também os resultados para os compressores de propósito geral *7zip* e *Gzip*. Apenas 2 formatos de imagem, *WebP* e *FLIF*, obtiveram mais de 75% de taxa média de economia de espaço quando comparados com as ferramentas especializadas de compressão de genomas, *DELIMINATE* e *MFCOMPRESS*, nas compressões sem a fase de transformação dos dados. Nos testes de compressão, levando em conta a fase de transformação dos dados da sequência genômica, os métodos *ORIGINAL+WEBP* e *BWT+FLIF* foram os únicos que superaram, em termos de compressão, pelo menos uma ferramenta especializada.

Capítulo 6

Conclusão

O problema tratado nesta pesquisa está ligado a alta complexidade dos dados de sequências genômicas, expresso em termos de espaço e tempo de processamento. A proposta, no entanto, empreendeu esforços apenas para reduzir parcialmente tal complexidade, centrando-se no problema de espaço. Nesta linha, assumiu-se que a compressão de dados é uma solução viável para reduzir a complexidade de espaço de armazenamento e transmissão de sequências genômicas, e para tal, certos formatos de imagens são métodos viáveis para mitigar o problema de armazenamento e transmissão de dados genômicos. E ainda, assumiu-se que a transformação do alfabeto de sequências genômicas, aplicada antes da compressão com formato de imagem, pode melhorar a taxa de compressão.

Deve-se também salientar que consecução do objetivo impunha restrições, a saber: o método de compressão de genomas deve realizar a tarefa sem perda de dados; a abordagem deve ser horizontal ou livre de referência. Nesta linha, experimentou-se diferentes formatos de imagens para codificar, armazenar e comprimir sequências de dados genômicos. Mostrou-se, a partir dos experimentos, que os arquivos de imagem resultantes, aplicados sobre as sequências genômicas, apresentaram taxa média de economia de dados semelhantes (em torno de 1%) quando comparados com ferramentas especializadas para compressão de dados genômicos.

De acordo com os resultados desse trabalho, os formatos de imagem que alcançaram pelo menos 75% de economia de espaço foram os formatos de imagem *WebP* e *FLIF*. Dessa forma, os formatos de imagem *Gif* e *Png* e também os compressores de propósito geral *Gzip* e *7zip* não atendem ao percentual mínimo viável para a compressão de genomas, pois não alcançam pelo menos 75% de economia de espaço. Não são viáveis à medida que a economia de espaço que eles produzem é menor que a simples recodificação de uma sequencia genômica usando *Naïve-bit encoding*, sem qualquer estratégia adicional.

Quando omitida a fase de transformação dos dados, os resultados mostraram que

a entropia está correlacionada com ganhos nas taxas médias de economia de espaço. Esses ganhos ficam latentes quando as compressões com imagem são avaliadas juntamente com ferramentas de compressão genômica especializadas; deve-se notar que estas últimas aplicam técnicas de transformação de dados antes da codificação, especialmente para o reino *Plant* cujas sequências genômicas são maiores em número de símbolos e também em termos de repetitividade.

Quando a fase de transformação de dados é aplicada como uma das fases da compressão observou-se melhora, acima de 2%, na taxa média de economia de espaço apenas para a técnica de transformação BWT seguida da codificação com o formato de imagem *FLIF*. As técnicas de transformações de dados MTF, RLE e também as combinações binárias com 8, 16, 32 e 64 combinações não produziram ganhos em termos de economia de espaço. O ganho de BWT se deve ao rearranjo na sequência genômica de modo que os símbolos semelhantes fiquem próximos uns dos outros. Esse rearranjo auxilia os formatos de imagens que aplicam técnica interna de transformação dos dados explorando o fato de que os pixeis vizinhos semelhantes são frequentemente correlacionados.

6.1 Trabalhos futuros

A partir dos resultados obtidos com a técnica BWT, parece um caminho interessante a explorar o uso de técnicas de transformação de dados de forma encadeada, e.g., utilizar uma técnica após a outra sobre os mesmos dados. Por exemplo: após aplicar a técnica MTF (que aumenta a frequência de símbolos iguais) aplicar a técnica BWT (que agrupa os símbolos iguais), poderia finalizar aplicando a técnica RLE (que faz a contagem dos símbolos repetidos). Assim, pode-se esperar que as taxas médias de economia de espaço sejam maiores. Também, além de comprimir as sequências genômicas com formato de imagem utilizando técnicas de transformação de forma encadeada, é necessário aprofundar a pesquisa para a possibilidade de busca por padrões e semelhanças que podem ser analisadas diretamente na imagem, sem a necessidade de descompressão para o arquivo no formato *FASTA* original. Desta forma, espera-se que os programas de análise genômica sejam aplicados diretamente sobre os dados no formato de imagem. A realização deste objetivo permite reduzir a complexidade de tempo de processamento.

Outra linha interessante de investigação é a compressão de sequências genômicas baseada em formato de vídeo por causa do aproveitamento dos píxeis de mesma cor de cada *frame*. Isso se daria com a divisão da sequência genômica em varias partes, codificadas como imagem, e a compressão destas imagens utilizando um formato de compressão de vídeo sem perda de dados. Essa mesma técnica também poderia ser aplicada em uma

coleção de sequências genômicas. Isso se daria com a união das sequências em uma única sequência e então a divisão da mesma em partes iguais para a compressão, sem perda de dados, com formato de vídeo.

Referências Bibliográficas

ALTSHULER, D. M.; DURBIN, R. M.; ABECASIS, G. R.; BENTLEY, D. R.; CHAKRAVARTI. An integrated map of genetic variation from 1,092 human genomes. *Nature*, v. 491, n. 7422, p. 56 65, 2012.

AUTON, A.; ABECASIS, G. R.; ALTSHULER, D. M.; DURBIN, R. M.; BENTLEY. A global reference for human genetic variation. *Nature*, v. 526, n. 7571, p. 68 74, 2015.

B. Alberts, A. Johnson, J. Lewis, M. Raff, K. Roberts and P. Walter. *CEUR Workshop Proceedings*. 6th edition. ed. New York: Garland Science, 2002.

BELLARD, F. *BPG Image format*. 2015. <http: bellard.org bpg >. Acessado em: 19/04/2018.

BENTLEY, J. L.; SLEATOR, D.; TARJAN, R. E.; WEI, V. K. A locally adaptive data compression scheme. *Commun. ACM*, v. 29, n. 4, p. 320–330, 1986.

BIJI, C. L.; NAIR, A. S. Benchmark dataset for Whole Genome sequence compression. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, PP, n. c, p. 1–10, 2016.

CHEN, X.; LI, M.; MA, B.; TROMP, J. DNACompress : fast and effective DNA sequence. *Bioinformatics*, v. 18, n. 12, p. 1696 1698, 2002.

COLLINS, F. S.; LANDER, E. S.; ROGERS, J.; WATERSON, R. H. Finishing the euchromatic sequence of the human genome. *Nature*, v. 431, n. 7011, p. 931–945, 2004.

Colt Mcanlis, A. H. *Understanding Compression*. 1st edition. ed. Sebastopol, CA: O'Reilly Media, Inc., 2016.

CompuServe Incorporated. *Graphics Interchange Format*. 1990. 35 p. <http: www.w3.org/Graphics/GIF/spec-gif89a.txt>, Acessado em: 19/04/2018.

DEMŠAR, J. Statistical Comparisons of Classifiers over Multiple Data Sets. *Journal of Machine Learning Research*, v. 7, p. 1–30, 2006. ISSN 1532-4435.

DUCE, D. *Portable Network Graphics (PNG) Specification (Second Edition)*. 2003. 1 p. <http://www.w3.org/TR/2003/REC-PNG-20031110/>, Acessado em: 19/04/2018.

FLEISCHMAN, E. *{WAVE} and {AVI} Codec Registries*. Internet: RFC Editor, 1998. 71 p. <http://www.rfc-editor.org/rfc/rfc2361.txt>, Acessado em: 19/04/2018.

FRIEDMAN, M. The Use of Ranks to Avoid the Assumption of Normality Implicit in the Analysis of Variance. *Journal of the American Statistical Association*, v. 32, n. 200, p. 675–701, 1937.

GIANCARLO, R.; SCATURRO, D.; UTRO, F. Textual data compression in computational biology: a synopsis. *Bioinformatics*, v. 25, n. 13, p. 1575–1586, 2009.

GIANCARLO, R.; SCATURRO, D.; UTRO, F. Textual data compression in computational biology: Algorithmic techniques. *Computer Science Review*, Elsevier Inc., v. 6, n. 1, p. 1–25, 2012.

GOOGLE. *WebP A new image format for the Web*. Internet: Google, 2018. <https://developers.google.com/speed/webp/>, Acessado em: 19/04/2018.

GRUMBACH, S.; TAHI, F. Compression of DNA sequences. *[Proceedings] DCC '93: Data Compression Conference*, p. 340–350, 1993.

GUO, H.; CHEN, M.; LIU, X.; XIE, M. Genome Compression Based on Hilbert Space Filling Curve. n. Meici, p. 1685–1689, 2015.

GUPTA, A.; AGARWAL, S. a Novel Approach for Compressing Dna Sequences Using Semi-Statistical Compressor. *International Journal of Computers and Applications*, v. 33, n. 3, 2011.

HAUBOLD, B.; WIEHE, T. How repetitive are genomes? *BMC bioinformatics*, v. 7, n. 1, p. 541, 2006.

IUBMB-IUPAC. International Union of Pure and Applied Joint Commission on Biochemical Nomenclature. *Pure & Applied Chemistry*, v. 56, n. 5, p. 595–624, 1984.

JPEG. *JPEG*. 2018. <http://www.jpeg.org>, Acessado em: 19 04 2018.

KAHN, S. D. On the future of genomic data. *Science*, v. 331, p. 728–729, 2011.

KALTON, G. *Introduction to Survey Sampling (Quantitative Applications in the Social Sciences)*. Newbury Park, CA: Sage Publication, 1983. 96 p.

KORODI, G.; TABUS, I. An efficient normalized maximum likelihood algorithm for DNA sequence compression. *ACM Transactions on Information Systems*, v. 23, n. 1, p. 3 34, 2005.

KURUPPU, S.; BERESFORD-SMITH, B.; CONWAY, T.; ZOBEL, J. Iterative dictionary construction for compression of large DNA data sets. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, v. 9, n. 1, p. 137–149. 2012.

LANDER, E. S.; LINTON, L. M.; BIRREN, B.; NUSBAUM, C.; ZODY, Y. J.; International Human Genome Sequencing, C. Initial sequencing and analysis of the human genome. *Nature*, v. 409, n. 6822, p. 860–921, 2001.

LEVY, S.; SUTTON, G.; NG, P. C.; FEUK, L.; HALPERN. The diploid genome sequence of an individual human. *PLoS Biology*, v. 5, n. 10, p. 2113–2144. 2007.

MA, B.; TROMP, J.; LI, M. PatternHunter: Faster and more sensitive homology search. *Bioinformatics*, v. 18, n. 3, p. 440–445. 2002.

MANOLIO, T. A.; CHISHOLM, R. L.; OZENBERGER, B.; RODEN, D. M.; WILLIAMS. Implementing genomic medicine in the clinic: The future is here. *Genetics in Medicine*, v. 15, n. 4, p. 258–267, 2013.

MIKKELSEN, T. S.; HILLIER, L. W.; EICHLER, E. E.; ZODY, M. C.; JAFFE. Initial sequence of the chimpanzee genome and comparison with the human genome. *Nature*, v. 437, n. 7055, p. 69–87, 2005.

MOHAMED, S.; FAHMY, M. Binary image compression using efficient partitioning into rectangular regions. *IEEE Transactions on Communications*, v. 43, n. 5, p. 1888–1893, 1995.

MOHAMMED, M. H.; DUTTA, A.; BOSE, T.; CHADARAM, S.; MANDE, S. S. DELIMINATE–a fast and efficient method for loss-less compression of genomic sequences: sequence analysis, v. 28, n. 19, p. 2527–9, 2012.

MOORE, G. E. Cramming more components onto integrated circuits (Reprinted from Electronics, pg 114-117, April 19, 1965). *Proceedings Of The Ieee*, v. 86, n. 1, p. 82 -85, 1965.

MUKAKA, M. M. Statistics corner: A guide to appropriate use of correlation coefficient in medical research. *Malawi Medical Journal*, v. 24, n. 3, p. 69–71, 2012.

National Human Genome Research Institute. All About the Human Genome Project. *National Human Genome Research Institute (NHGRI)*, p. 1, 2015. Disponível em: <https://www.genome.gov/10001772>.

National Human Genome Research Institute. *DNA Sequencing Costs: Data - National Human Genome Research Institute (NHGRI)*. Internet: NIH, 2018. 1 p. <https://www.genome.gov/27541954/dna-sequencing-costs-data/>. Acessado em: 19/04/2018.

National Human Genome Research Institute. *The Cost of Sequencing a Human Genome - National Human Genome Research Institute (NHGRI)*. Internet: NIH, 2018. 1 p. <https://www.genome.gov/sequencingcosts >. Acessado em: 19/04/2018.

NCBI. *Genome Information by Organism*. Internet: NIH, 2018. <https://www.ncbi.nlm.nih.gov/genome/browse >. Acessado em: 19 04 2018.

NEMENYI, P. *Distribution-free Multiple Comparisons*. 263 p. Tese (Doutorado). Princeton. NJ, 1962.

NIH, N. *Sequence Read Archive*. Internet: NCBI. 2018. <https: trace.ncbi.nlm.nih.gov Traces/sra/sra.cgi?>, Acessado em: 19 04 2018.

NILSSON, M. *The audio/mpeg Media Type*. Internet: RFC Editor, 2000. 5 p. <http://www.rfc-editor.org/rfc/rfc3003.txt>. Acessado em: 19 04 2018.

OLSON, S.; BEACHY, S. H.; GIAMMARIA, C. F.; BERGER, A. C. *Roundtable on Translating Genomic-Based Research for Health Board on Health Sciences Policy*. [S.l.: s.n.], 2013. ISBN 9780309220347.

PADMANABHAN, R.; PADMANABHAN, R.; WU, R. Nucleotide sequence analysis of DNA. IX. Use of oligonucleotides of defined sequence as primers in DNA sequence analysis. *Biochemical and Biophysical Research Communications*, v. 48, n. 5, p. 1295-1302. 1972.

PAVLOV, I. *7-Zip*. 2018. <https://www.7-zip.org>, Acessado em: 19/04 2018.

PENG, F. M. Novel composition test functions algorithm for numerical optimization. *2011 International Conference on Computer Science and Service System, CSSS 2011 - Proceedings*, p. 3348 3352, 2011.

PINHO, A. J.; FERREIRA, P. J.; NEVES, A. J.; BASTOS, C. A. On the representability of complete genomes by multiple competing finite-context (Markov) models. *PLoS ONE*, v. 6, n. 6, 2011.

PINHO, A. J.; PRATAS, D. Mfcompress: A compression tool for fasta and multi-fasta data. *Bioinformatics*, v. 30, n. 1, p. 117–118, 2014.

PINHO, A. J.; PRATAS, D.; FERREIRA, P. J. Bacteria DNA sequence compression using a mixture of finite-context models. *IEEE Workshop on Statistical Signal Processing Proceedings*, p. 125–128, 2011.

PKWARE. *ZIP Application Note*. Internet: PKWARE, 1989. <https://pkware.cachefly.net/webdocs/casestudies/APPNOTE.TXT>, Acessado em: 19/04/2018.

PRATAS, D.; PINHO. A. J. Exploring deep markov models in genomic data compression using sequence pre-analysis. p. 2395–2399. Sept 2014. ISSN 2219-5491.

PRATAS, D.; PINHO. A. J.; FERREIRA. P. J. Efficient Compression of Genomic Sequences. *Data Compression Conference Proceedings*, p. 231–240, 2016.

REGALADO, A. *Illumina Says 228.000 Human Genomes Will Be Sequenced This Year*. Internet: MIT. 2014. <https: www.technologyreview.com s 531091 emtech-illumina-says-228000-human-genomes-will-be-sequenced-this-year>. Acessado em: 19/04/2018.

REUTER, J. A.; SPACEK. D.; SNYDER. M. P. High Throughput Sequencing Technologies. *Molecular Cell*, v. 58, n. 4, p. 586–597, 2016.

RIVALS, E.; DELAHAYE. J.-p. A guaranteed compression scheme for repetitive DNA sequences. *Data Compression ...*, p. 59655, 1996.

ROBINSON, A. H.; CHERRY, C. Results of a Prototype Television Bandwidth Compression Scheme. *Proceedings of the IEEE*, v. 55, n. 3, p. 356–364, 1967. ISSN 15582256.

SALOMON, D. *Data Compression: The Complete Reference*. 4th edition. ed. London, UK: Springer-Verlag London, 2007. v. 53. 1689–1699 p.

SANGER, F.; NICKLEN, S.; COULSON, a. R. DNA sequencing with chain-terminating inhibitors. *Proceedings of the National Academy of Sciences of the United States of America*, v. 74, n. 12, p. 5463–7, 1977.

SCHLOSS, J. A. *Nature Biotechnology*. Internet: Nature Biotechnology, 2008. 1113–1115 p. <https://www.nature.com/articles/nbt1008-1113>. Acessado em: 19/04/2018.

SHANNON, C. E. A mathematical theory of communication. *The Bell System Technical Journal*, v. 27, n. July 1928, p. 379 423, 1948.

SIEST, G.; MARTEAU, J.; VISVIKIS-SIEST, S. Personalized therapy and pharmacogenomics: future perspective. *Pharmacogenomics*, n. May, p. 927–930, 2009.

SNEYERS, J.; WUILLE, P. FLIF: Free lossless image format based on MANIAC compression. *Proceedings - International Conference on Image Processing, ICIP*, v. 2016-Augus, p. 66–70, 2016.

STEPHENS, Z. D.; LEE, S. Y.; FAGHRI, F.; CAMPBELL, R. H.; ZHAI. Big data: Astronomical or genomical? *PLoS Biology*, Public Library of Science, v. 13, n. 7, 2015.

TABUS, I.; KORODI, G.; RISSANEN, J. DNA Sequence Compression Using the Normalized Maximum Likelihood Model for Discrete Regression. *Dcc*, p. 253, 2003.

TREES, S. Algorithmica. v. 26, p. 249–260, 1995.

Wetterstrand KA. DNA Sequencing Costs: Data. *National Human Genome Research Institute*, 2016. Disponível em: <https: www.genome.gov sequencingcostsdata >.

XIE, X.; ZHOU, S.; GUAN, J. CoGI: Towards Compressing Genomes as an Image. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*. v. 12, n. 6, p. 1275–1285, 2015.

Xin Chen; KWONG, S.; Ming Li. A compression algorithm for DNA sequences. *IEEE Engineering in Medicine and Biology Magazine*, v. 20, n. 4, p. 61–66, 2001.

ZHU, Z.; ZHOU, J.; JI, Z.; SHI, Y.-H. DNA Sequence Compression Using Adaptive Particle Swarm Optimization-Based Memetic Algorithm. *IEEE Transactions on Evolutionary Computation*, v. 15, n. 5, p. 643–658, 2011.

ZIV, J.; LEMPEL, A. A Universal Algorithm for Sequential Data Compression. *IEEE Transactions on Information Theory*, v. 23, n. 3, p. 337–343, 1977.