UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Campus de São José do Rio Preto

José Augusto Sabino de Oliveira

Melhoramento do resultado final do alinhamento múltiplo de sequências por meio de métodos de rearranjo de árvores e do princípio da evolução mínima

São José do Rio Preto
2019

José Augusto Sabino de Oliveira

Melhoramento do resultado final do alinhamento múltiplo de sequências por meio de métodos de rearranjo de árvores e do princípio da evolução mínima

Dissertação apresentada como parte dos requisitos para obtenção do título de Mestre em Ciência da Computação, junto ao Programa de Pós-Graduação em Ciência da Computação, do Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista "Júlio de Mesquita Filho", Campus de São José do Rio Preto.

Orientador: Prof. Dr. Geraldo Francisco Donegá Zafalon

São José do Rio Preto
2019

O48m

Oliveira, José Augusto Sabino de

Melhoramento do resultado final do alinhamento múltiplo de sequências por meio de métodos de rearranjo de árvores e do princípio da evolução mínima / José Augusto Sabino de Oliveira. -- São José do Rio Preto, 2019

95 f. : il., tabs.

Dissertação (mestrado) - Universidade Estadual Paulista (Unesp), Instituto de Biociências Letras e Ciências Exatas, São José do Rio Preto

Orientador: Geraldo Francisco Donegá Zafalon

1. Ciência da computação. 2. Bioinformática. 3. Alinhamento de sequências. 4. Rearranjo de Árvore. I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Biblioteca do Instituto de Biociências Letras e Ciências Exatas, São José do Rio Preto. Dados fornecidos pelo autor(a).

Essa ficha não pode ser modificada.

José Augusto Sabino de Oliveira

Melhoramento do resultado final do alinhamento múltiplo de sequências por meio de métodos de rearranjo de árvores e do princípio da evolução mínima

Dissertação apresentada como parte dos requisitos para obtenção do título de Mestre em Ciência da Computação, junto ao Programa de Pós-Graduação em Ciência da Computação, do Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista "Júlio de Mesquita Filho", Campus de São José do Rio Preto.

Comissão Examinadora

Prof. Dr. Geraldo Francisco Donegá Zafalon
UNESP – Câmpus de São José do Rio Preto
Orientador

Prof. Dr. Jorge Rady de Almeida Junior
USP – Câmpus de São Paulo

Prof. Dr. Leandro Alves Neves
UNESP – Câmpus de São José do Rio Preto

São José do Rio Preto
07 de agosto de 2019

Ao meu pai, que sempre me guiou pelo caminho correto, acreditou e me inspirou na busca pelo conhecimento.
A minha esposa Maria, pela força e companheirismo ao longo dessa caminhada.
Para Alexandre, Leonardo, Mirella e Arthur.

Agradecimentos

À Deus por ser a luz que me guia e não me deixa desanimar frente aos desafios;

À toda minha família, por toda força e confiança;

Em especial, meu pai que sempre me guia e inspira a fazer sempre o melhor;

A minha esposa pelo companheirismo e pelo tempo abdicado;

Aos meus filhos, por serem o motivo de toda minha luta;

Ao meu orientador, por ter acredito e viabilizado a realização desse trabalho;

A todos do Grupo de Computação Cientifica da Unesp de São José do Rio Preto;

Aos colegas do Laboratório de Bioinformática da Unesp de São José do Rio Preto;

A todos os meus professores.

"*Aquele que ousa perder uma hora de seu tempo não sabe o valor da vida.*"
(CHAPIN; CHAPIN, 1935)

Resumo

Os recentes avanços científicos e tecnológicos proporcionaram um grande aumento dos dados biológicos disponíveis para análise e técnicas manuais são inviáveis para executar a análise desses dados e, assim, a Bioinformática surgiu devido a necessidade de analisar-se esse volume de dados com ferramentas computacionais e possibilitar inferências relevantes. Neste contexto, técnicas de alinhamento de sequências são ferramentas imprescindíveis. No entanto, alinhar sequências tem alto custo computacional e neste cenário, adota-se o Alinhamento Múltiplo de Sequências no qual diversas heurísticas são adotadas para amenizar o problema do custo computacional. Uma destas é o Alinhamento Progressivo que funciona basicamente em três fases: alinhamento par a par, construção da árvore guia e o alinhamento de perfis. Diversos estudos destacam a importância da árvore guia e de refiná-la para manter as sequências mais similares em ramificações próximas e assim, produzir um resultado com melhor significância biológica. Assim, no presente trabalho apresenta-se conceitos e técnicas essenciais da Bioinformática e de construção da árvore guia e, por fim apresenta-se um método de refinamento da árvore guia aplicado em ferramentas que usam o Alinhamento Progressivo e que trabalha entre as etapas de construção da árvore guia e do profile final. O método recebe como entrada uma árvore guia e sua matriz de distâncias produzida pelo Alinhamento Progressivo, promove mudanças em suas ramificações internas e avalia se a probabilidade da árvore produzida é mais próxima da árvore verdadeira. Se probabilidade da árvore produzida é maior que a da árvore guia inicial, a nova árvore é devolvida como a árvore guia para a ferramenta de alinhamento construir o profile final, o que proporciona resultados com melhor significado biológico.

Palavras-chave: Bioinformática. Alinhamento Múltiplo de Sequências. Alinhamento Progressivo. Rearranjo de Árvores. Refinamento da Árvore Guia.

Abstract

Recent scientific and technological advances have provided a large increase in the biological data available for analysis and manual techniques are unfeasible to perform the analysis of this data and thus Bioinformatics arose due to the need to analyze this data volume with computational tools and enable relevant inferences. In this context, sequence alignment techniques are essential tools. However, aligning sequences has a high computational cost and in this scenario, Multiple Sequence Alignment is adopted in which several heuristics are adopted to alleviate the computational cost problem. One of these is Progressive Alignment which works basically in three phases: pairwise alignment, guide tree construction and profile alignment. Several studies highlight the importance of the guide tree and refining it to maintain the most similar sequences in close branches and thus produce a result with better biological significance. Thus, in the present work we discuss essential concepts and techniques of Bioinformatics and the construction of the guide tree and, finally, we present a method of refinement of the guide tree applied in tools that use Progressive Alignment and that works between the guide tree build step and the final profile build step. The method receives as input a guide tree and its distance matrix produced by Progressive Alignment, makes changes in its internal branches and evaluates if the probability of the produced tree is closer to the true tree. If the produced tree's probability is greater than that of the initial guide tree, the new tree is returned as the guide tree for the alignment tool to build the final profile, which gives better biological significance results.

Keywords: Bioinformatics. Multiple Sequence Alignment. Progressive Alignment. Tree Rearrangement. Guide Tree Refinement.

Lista de Figuras

Lista de Tabelas

Lista de Abreviações

AMS *Alinhamento Múltiplo de Sequências*

DNA *Deoxyribonucleic Acid*

FFT *Fast Fourier Transform*

GA *Algoritmos Genéticos*

GPU *Graphics Processing Unity*

GPU *Graphics Processing Unit*

HMM *Hidden Markov Model*

IDE *Integrated Development Environment*

MAFFT *Multiple Alignment using Fast Fourier Transform*

ME *Minimum Evolution Principle*

ML *Maximum-Likelihood*

MP *Maximum-Parsimony*

MUSCLE *Multiple Sequence Comparison by Log-Expectation*

NGS *Next Generation Sequencing*

NJ *Neighbor-Joining*

NNI *Nearest Neighbor Interchange*

PCR *DNA Polimerase*

PD *Programação Dinâmica*

PHYML *Phylogenetic Maximum Likelihood*

Q *Q Score*

RAxML *Randomized Axelerated Maximum Likelihood*

RNA *Ribonucleic Acid*

SA *Simulated Annealing*

SP *Sum-of-Pairs*

SPR *Subtree Pruning and Regrafting*

TBR *Tree Bisection and Reconnection*

TBR *Tree Bisection and Reconnection*

TC *Total Column Score*

TRMLE *Tree Rearrangement + Maximum Likelihood + Minimum Evolution*

UPGMA *Unweighted Pair Group Method using Arithmetic averages*

WSP *Weighted Sum-of-Pairs*

Sumário

Capítulo 1

Introdução

1.1 Contextualização

São muitas as revoluções e descobertas que ocorreram em todas as áreas da ciência no século XX e nesse princípio do século XXI. Entre todas ciências, a Biologia enquadra-se como uma das que mais se destacou, devido aos vários avanços que tem proporcionado (COHEN, 2004). Apesar de haver registros históricos de que os antigos chineses, gregos e egípcios estudavam alguns aspectos dos seres vivos e de estruturas das células em séculos anteriores, a Biologia só teve sua considerável revolução a partir da descoberta da estrutura 3D, em forma de hélice, do DNA, por Watson e Crick em 1953 (WATSON; CRICK et al., 1953).

Por outro lado, a Ciência da Computação é uma área que se desenvolveu no século XX, com importantes contribuições do matemático britânico Alan Turing, que lançou as suas bases ao introduzir, em 1936, um modelo matemático simples, de uma máquina hipotética, para o processo computacional e que atualmente é conhecida como Máquina de Turing (TURING, 1937). Além disso, houve também uma contribuição relevante do matemático húngaro John von Neumann, que trabalhou com seus colegas do Instituto de Estudos Avançados de Princeton, entre 1946 e 1952, para desenvolver um computador que executasse programas armazenados, que foi chamado de IAS

Computer e se tornou o protótipo de todos os computadores modernos (STALLINGS, 2003).

No entanto, a Ciência da Computação não é um fim específico, mas um meio pelo qual as demais ciências podem obter melhores resultados para os seus trabalhos e pesquisas. Dessa forma, solucionando problemas complexos de forma automática, mais rápido do que o trabalho e a mente humana são capazes de realizar. A aplicação das técnicas e ferramentas da Ciência da Computação na área das Ciências Biológicas deu origem a uma área de conhecimento multidisciplinar conhecida como Bioinformática (EDGAR; BATZOGLOU, 2006).

Um dos principais fatores que contribuíram para a evolução da Bioinformática e que a torna cada vez mais importante, relaciona-se com os avanços na medicina. O progresso fundamental na medicina depende da elucidação de como ocorrem os vários processos biológicos, levando-se a uma necessidade crescente e constante de avanços tecnológicos (COHEN, 2004). O projeto Genoma Humano e seus desmembramentos podem ser citados como exemplos de impulsionadores desses avanços citados na Bioinformática (PROSDOCIMI et al., 2002).

Além desses fatores de avanço relatados anteriormente, o surgimento dos sequenciadores de nova geração *NGS (Next Generation Sequencing)* revolucionou ainda mais a pesquisa genômica, devido ao grande volume de dados que ele é capaz de gerar, quando comparado com a tecnologia anterior, a Sanger (LIU et al., 2012). Para se ter um efeito comparativo, o que um NGS produz em um dia, um Sanger produz em anos (BEHJATI; TARPEY, 2013). Esse fato tem aumentando de maneira considerável a quantidade de dados biológicos de sequências de DNA e de proteínas que precisam ser analisados e armazenados. Com esses avanços tecnológicos, tornou-se infactível a análise manual desses dados coletados pelos biólogos (ZAFALON et al., 2015), que têm a necessidade de interpretá-los. Dessa forma, estratégias computacionais para analisá-los tornaram-se essenciais, especialmente considerando-se que os dados obti-

das por meio do sequenciamento são expressados na forma de cadeia de caracteres, ou seja, sequências de símbolos (COHEN, 2004).

No caso do DNA, que é base da hereditariedade, trata-se de um polímero composto por moléculas chamadas nucleotídeos, que podem ser mapeados por quatro bases nitrogenadas: Adenina (A), Citosina (C), Guanina (G) e Timina (T). Uma sequência de DNA é, portanto, especificada completamente por uma sequência composta por um alfabeto de quatro letras A, C, G, T (LIEW; YAN; YANG, 2005). Essa mesma abstração aplica-se aos aminoácidos, no entanto, usa-se um alfabeto de vinte letras, cujas combinações formam as proteínas.

A Bioinformática desempenha um importante papel na coleta e processamento dos dados genômicos obtidos pelos biólogos no estudo das funções proteicas, bem como a determinação das regiões conhecidas como pontos quentes *(hotspots)*, que são relevantes para a indústria farmacêutica (COHEN, 2004). O conhecimento detalhado das estruturas das proteínas facilita o desenvolvimento de medicamentos capazes de atuar nessas regiões de interesse e atacar as causas das potenciais doenças. Desta forma, algumas das tarefas típicas realizadas na Bioinformática incluem indicar a forma e a função de uma proteína a partir de uma determinada sequência de aminoácidos, encontrar todos os genes e proteínas em um dado genoma e determinar os locais na estrutura proteica onde moléculas de drogas podem ser anexadas (COHEN, 2004). Essa percepção apresenta-se como um fator de suma importância em diversos cenários, principalmente no auxílio à eventual cura de diferentes tipos de doenças (KHURI, 2008). Assim, biólogos têm a necessidade do poder computacional para a análise satisfatória dos dados que, geralmente, são obtidos em sua forma bruta e que precisam ser processados para a realização de posteriores inferências (KHURI, 2008).

1.2 Justificativas

Existem vários métodos que são usados em várias ferramentas computacionais para extrair informações genômicas das amostras biológicas sequenciadas e as que mais se destacam são as ferramentas de alinhamento de sequências (EDGAR; BATZOGLOU, 2006). Uma das principais estratégias utilizadas pelos métodos empregados para as análises de sequências biológicas é o Alinhamento Múltiplo de Sequências (AMS) (MORRISON; MORGAN; KELCHNER, 2015). O AMS é uma estratégia útil na previsão de estruturas de proteínas, análise filogenética, predição de função e no desenho dos iniciadores da reação em cadeia da *DNA polimerase (PCR)* (NOTREDAME; HOLM; HIGGINS, 1998).

No entanto, AMS é uma técnica não-determinística e nem sempre garante o resultado exato, mas sim um alinhamento ótimo, ao contrário do que pode ser obtido por meio de técnicas determinísticas, como a Programação Dinâmica (PD). Dentre essas técnicas de PD, destacam-se os algoritmos de Needleman-Wunsch (NEEDLEMAN; WUNSCH, 1970) e de Smith-Waterman (SMITH; WATERMAN, 1981). Estas técnicas são adequadas para tratarem pares de sequências (WANG; JIANG, 1994). Assim, a utilização de estratégias determinísticas para alinhamentos de múltiplas sequências torna-se uma tarefa com alto custo computacional, uma vez que os algoritmos de PD anteriormente citados enquadram-se como problemas da classe dos NP-completos, com complexidade de tempo e espaço $O(L^N)$, em que L é o comprimento da sequência e N é o número de sequências de uma conjunto (WATERMAN; SMITH; BEYER, 1976; WANG; JIANG, 1994; EDGAR, 2004a; WANG et al., 2015), com $1 \geq N \leq \infty$.

Devido à complexidade computacional das técnicas determinísticas, houve uma concentração de trabalhos nas técnicas de AMS, que podem ser baseadas em diversas heurísticas (EDGAR; BATZOGLOU, 2006). Destas, destaca-se atualmente o Alinhamento Progressivo (FENG; DOOLITTLE, 1987), que é o mais implementado nas diversas ferramentas bem conhecidas, como as da família Clustal (THOMPSON; HIG-

GINS; GIBSON, 1994; SIEVERS; HIGGINS, 2014; SIEVERS; HIGGINS, 2018), a MUSCLE (EDGAR, 2004a), a MAFFT (KATOH; ROZEWICKI; YAMADA, 2017), a MULTAL (TAYLOR, 1988), entre outras.

A característica que torna o Alinhamento Progressivo interessante deve-se ao fato de que as sequências mais semelhantes, ou menos distantes, são alinhadas primeiro por meio de uma árvore binária, que neste contexto recebe o nome de árvore guia. A árvore guia é construída com auxílio de uma matriz de distâncias, na qual são armazenadas as distâncias evolutivas computadas com o auxílio de uma função de medida, para todos os possíveis pares de sequências de um dado conjunto de sequências fornecido como entrada para o algoritmo. Os pares de sequências menos distantes, ou mais semelhantes, são agrupados primeiro e assume-se que, no Alinhamento Progressivo, o melhor resultado é obtido em cada nó ao alinhar os dois perfis com menor número de diferenças, mesmo que não sejam vizinhos evolutivos (BOYCE; SIEVERS; HIGGINS, 2015; EDGAR, 2004b).

O Alinhamento Progressivo tem como vantagem a velocidade, a simplicidade e a sensibilidade. Porém, erros ocorridos no início do processo não são corrigidos nas fases posteriores, devido à característica gulosa da estratégia e podem conduzir a distorções no resultado final (NOTREDAME; HOLM; HIGGINS, 1998). Isto leva à necessidade de esforços para melhorar a acurácia da árvore guia. O desenvolvimento de estratégias capazes de conciliar desempenho computacional e acurácia dos alinhamentos, dando um significado biológico relevante é um dos desafios na área da Bioinformática (MORRISON; MORGAN; KELCHNER, 2015). Uma vez que o Alinhamento Progressivo é sensível aos problemas que ocorrem na fase de cálculo das distâncias, as técnicas de rearranjo da árvore guia destacaram-se como um ponto fundamental, pois podem amenizar essa distorção (HSIEH et al., 2015).

Neste contexto, observam-se vários estudos apontando para a importância da árvore guia e para a necessidade de melhorar sua acurácia, de modo a garantir um ali-

nhamento final com maior significância biológica. Boyce (BOYCE; SIEVERS; HIGGINS, 2015) demonstrou que uma simples mudança na ordem dos conjuntos de entrada, pode produzir resultados diferentes para esse mesmo conjunto. Penn (PENN et al., 2010) mostrou que as incertezas na árvore guia são fontes importantes de incertezas no alinhamento. Capella-Gutierrez e Gabaldon (CAPELLA-GUTIÉRREZ; GABALDÓN, 2013) mostraram que a maioria das lacunas (**gaps**) são inseridas em padrões que seguem a árvore guia. Zhan (ZHAN et al., 2015) usou um método adaptativo de construção da árvore para demonstrar a melhora da acurácia de diversas ferramentas de AMS com o uso de melhores árvores guia. Algumas ferramentas também têm obtido melhores resultados por meio do uso de técnicas para construir melhores árvores guia, como GLProbs (YE et al., 2015) e a PnpProbs (YE; LAM; TING, 2016). Hsieh (HSIEH et al., 2015) usou uma adaptação da técnica de bissecção e reconexão da árvore *(TBR - Tree Bisection and Reconnection)*, a fim de melhorar a reconstrução de árvores guia, combinada com o princípio da evolução mínima *(ME - Minimum Evolution Principle)* (RZHETSKY; NEI, 1993) para filtrar posições reconectadas desnecessárias e reduzir o tempo de busca, demonstrando que o método pode ajudar outros algoritmos na construção de árvores mais precisas, com tempo de processamento aceitável.

1.3 Motivação e Escopo

A partir dos estudos citados na seção 1.2 demonstra-se a importância da árvore guia, também conhecida como árvore filogenética, que é produzida pelo AMS durante a execução da segunda fase do processo de alinhamento. Melhorar a sua acurácia, ou seja, aplicar técnicas que promovam melhorias a fim de gerar uma árvore de tamanho mínimo e, que desta forma, reflita com maior proximidade a linha da evolução das espécies, ainda é um problema na área da Bioinformática. Uma árvore guia mais acurada é uma árvore em que as sequências com a menor distância evolutiva estão agrupadas juntas, ou mais próximas. Dessa forma, pode-se gerar arestas com pesos

mínimos, a fim de representar melhor a evolução das espécies e, consequentemente, são capazes de produzir um resultado do AMS com maior significância biológica.

Técnicas que promovem mudanças na topologia da árvore guia tem por objetivo produzir uma árvore de tamanho mínimo, para a qual a soma dos tamanhos de todas as suas arestas, aproxima-se ao máximo do tamanho da árvore evolutiva verdadeira, que é uma árvore que representa a verdadeira evolução das espécies. Neste contexto, a acurácia de uma árvore guia é medida pelo resultado da diferença de seu tamanho quando comparado com o tamanho da árvore verdadeira, onde quanto menor a diferença, melhor é a árvore produzida.

Produzir resultados do AMS com maior significância biológica, com um resultado capaz de representar com maior proximidade a evolução das espécies e demonstrar suas similaridades, o mais próximo da verdadeira evolução, é um fator importante para os pesquisadores das áreas de saúde, da indústria farmacêutica e das ciências biológicas. Na área de saúde e na indústria farmacêutica, tais resultados são importantes para determinar quais são as terapias e medicamentos mais adequados para determinada enfermidade ou no combate de vírus e bactérias nocivos a saúde.

Sendo assim, constitui-se como motivação deste trabalho desenvolver um método capaz interagir com as ferramentas de AMS e promover melhorias na árvore guia, utilizando técnicas de rearranjo de árvore, convergindo para resultados finais do AMS com maior significância biológica e contribuir com os trabalhos de biólogos, pesquisadores e profissionais de saúde.

1.4 Objetivos

O presente trabalho tem por objetivo o desenvolvimento de um método de rearranjo que promova melhorias na acurácia da árvore guia e produza uma árvore guia com tamanhos mínimos de arestas, mais próxima da árvore evolutiva verdadeira, gerando um resultado com maior qualidade nos alinhamentos produzidos, sem degradar o de-

sempenho computacional em termos de tempo de execução e, assim, contribuir para a evolução da significância biológica dos resultados finais produzidos pelo AMS.

O método atua nas ferramentas de AMS entre as fases de construção da árvore guia e de construção do AMS, mas não se limita a uma única ferramenta. Desta forma, foi desenvolvido em um módulo separado que recebe como entrada uma árvore guia e sua matriz de distâncias, executa trocas em suas ramificações internas para melhorar a sua acurácia e retorna a árvore guia modificada para a ferramenta de AMS. Para o desenvolvimento do presente trabalho, foi escolhida uma única ferramenta que serviu de base para medição dos resultados, que, neste caso, foi a ferramenta MUSCLE (EDGAR et al., 2005).

Sendo assim, trata-se de uma estratégia de rearranjo de árvores com o propósito de produzir árvores guias vizinhas da árvore guia inicial, produzida pela ferramenta de alinhamento hospedeira. Desta forma, é possível dota-la da capacidade de selecionar uma árvore melhor e de tamanho reduzido, mais próximo da filogenia verdadeira. Com isso, majora-se a capacidade de produzir melhores resultados no alinhamento das sequências biológicas, com o uso do Alinhamento Progressivo.

1.5 Organização do Trabalho

O presente trabalho está dividido em cinco capítulos, incluindo o atual. No Capítulo 2, realiza-se uma revisão bibliográfica referente aos assuntos e conceitos pertinentes a área da Biologia e da Bioinformática, bem como apresenta-se algumas das técnicas de AMS e de rearranjo de árvores pertinentes ao escopo do trabalho da dissertação desenvolvida. No capítulo 3, apresenta-se a proposta do método para rearranjo da árvore guia e todo o trabalho executado para o seu desenvolvimento. No capítulo 4, os resultados obtidos como o uso do método são medidos e comprados com os resultados obtidos pela ferramenta sem o uso do método proposto. Por fim, no Capítulo 5, apresentam-se as conclusões do presente trabalho e os trabalhos futuros.

Capítulo 2

Revisão Bibliográfica

Neste capítulo são apresentados os conceitos fundamentais sobre Biologia Molecular e Bioinformática, além das estratégias para alinhamentos de sequências e técnicas para reconstrução da árvore guia, que são indispensáveis para o entendimento do contexto de desenvolvimento do presente trabalho.

2.1 Biologia Molecular

A Bioinformática é uma ciência multidisciplinar que nasceu da necessidade de se compreender as funções biológicas dos genes, por meio da vasta quantidade de dados coletados pelos biólogos, além do requisito de interpretar esses dados. Isso envolve disciplinas como a Matemática, a Física, a Química, a Estatística, a Ciência da Computação e a Biologia Molecular (COHEN, 2004; PROSDOCIMI et al., 2002). Desta forma, para que os cientistas da computação tenham um bom entendimento do contexto da Bioinformática, necessitam de compreensão dos conceitos básicos da Biologia Molecular, como organização molecular, filogenia e padrões (COHEN, 2004).

Embora a Biologia tenha se impulsionado no século XX, alguns estudos relacionados à área remontam aos tempos antigos. Os grandes avanços na área só foram possíveis a partir do surgimento dos microscópios e dos aperfeiçoamentos feitos pelo

holandês Anton van Leeuwenhoek (1632 – 1723), bem como pelos posteriores trabalhos do inglês Robert Hooke (1635 – 1703). Este criou o microscópio composto de duas lentes – a ocular e a objetiva – o que possibilitou a primeira observação da célula (GEST, 2004).

O microscópio possibilitou, por exemplo, a descoberta dos espermatozoides, bem como o surgimento da teoria celular, por volta do ano 1838. Nesse período, os alemães Mathias Schleiden (1804 – 1881) e Theodor Schwann (1810 – 1882) concluíram que todas as plantas e animais eram compostos por células e, posteriormente, a descoberta do DNA em 1869 pelo também alemão Johann Friedrich Miescher (1844 – 1895) (DAHM, 2008).

Desta forma, a biologia evoluiu para vários campos de pesquisa, classificados pela escala em que os organismos vivos são estudados, os tipos de organismos e os métodos aplicados. A Biologia Molecular surge como uma subdisciplina da Biologia e um campo amplo de estudos que envolve a Bioquímica, a Biofísica e a Genética. Seu estudo está voltado para o nível molecular, com foco na estrutura e função do material genético e nas interações complexas entre as moléculas nos vários sistemas de uma célula, como as interações do DNA (*Deoxyribonucleic Acid*) , do RNA (*Ribonucleic Acid*), a síntese das proteínas e as regras que regem essas interações (COHEN, 2004; PROSDOCIMI et al., 2002).

2.1.1 Organização Celular

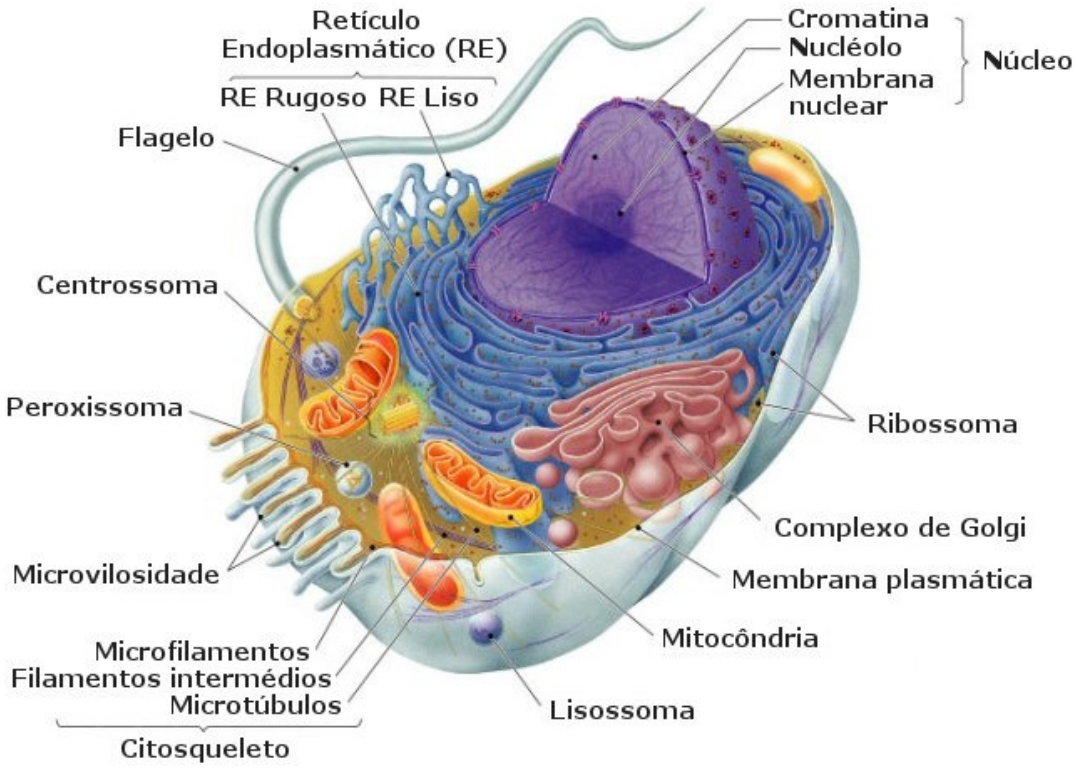
Nos estudos da Biologia busca-se a compreensão da diversidade dos organismos vivos existentes e isto levou à constatação de que independentemente do tipo e forma do organismo estudado, desde que sejam organismos vivos, a célula é a unidade básica que representa as suas estruturas e funções. Desta forma, para obter-se maior conhecimento dos organismos vivos, realiza-se um estudo celular e molecular nesses organismos, que permite compreender o seu funcionamento (LEMOS; ARAGAO;

CASANOVA, 2003).

As células são divididas entre os grupos dos eucariontes e dos procariontes (RUBIN et al., 2000; WHITMAN; COLEMAN; WIEBE, 1998). As células eucariontes são mais desenvolvidas e estão presentes nos organismos mais complexos, como plantas e animais. No entanto, apresentam algumas diferenças que podem ser notadas na representação das Figuras 2.1 e 2.2. As células procariontes, representadas na Figura 2.3, são células mais simples e que estão presentes em organismos menos complexos, com as bactérias (COHEN, 2004).

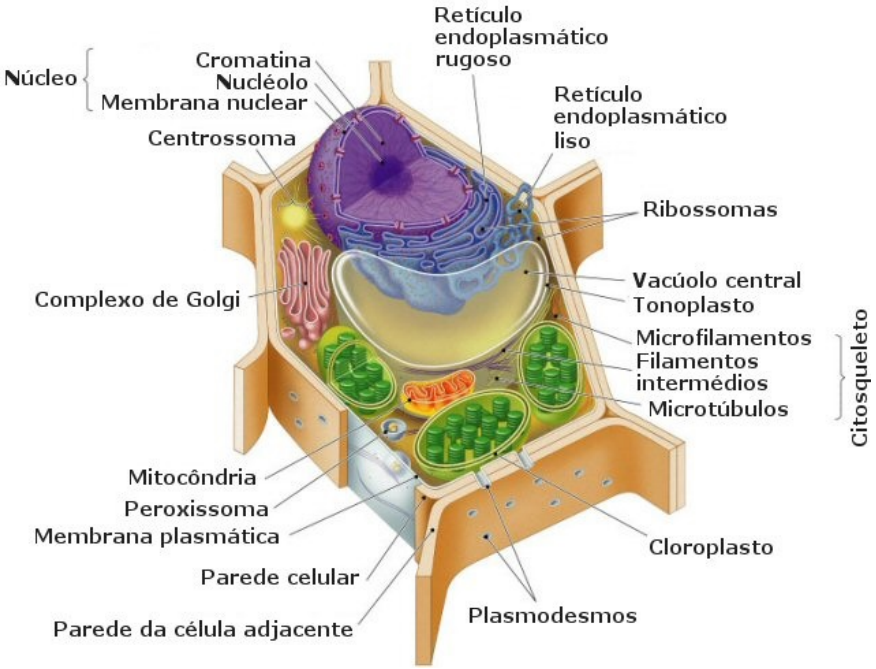
A principal diferença entre os dois grupos de células é a presença de um envoltório nuclear no grupo eucariontes, o que não ocorre com o grupo dos procariontes. Como observa-se na Figura 2.4, nessa região nuclear encontra-se o material genético, motivo de muitos estudos e pesquisas por parte dos biólogos (LEMOS; ARAGAO; CASANOVA, 2003).

Figura 2.1: Exemplo de célula eucarionte animal.



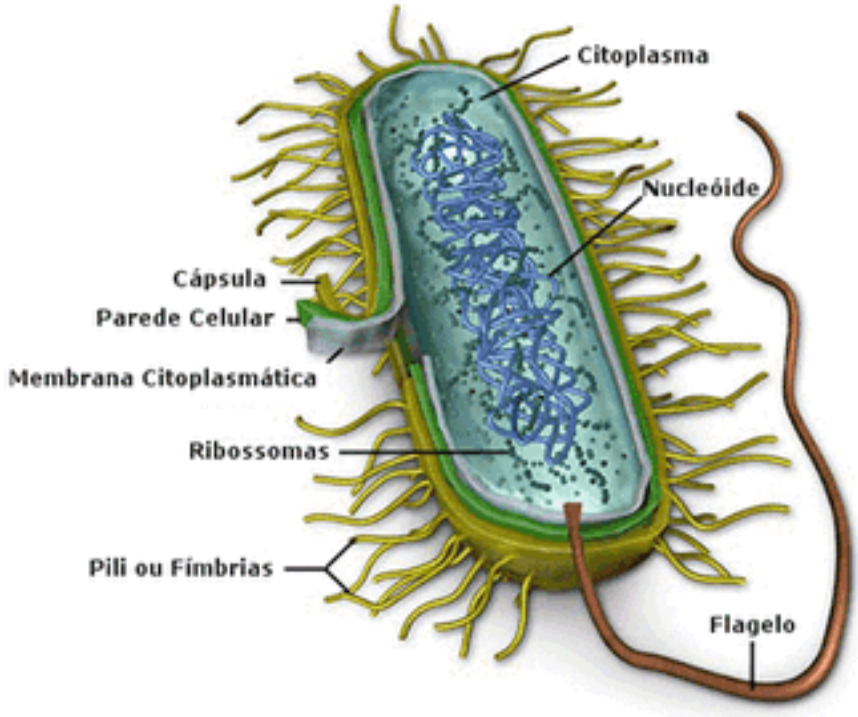
Fonte: http://www.simbiotica.org/imagens/celulaanimal.jpg.

Figura 2.2: Exemplo de célula eucarionte vegetal.



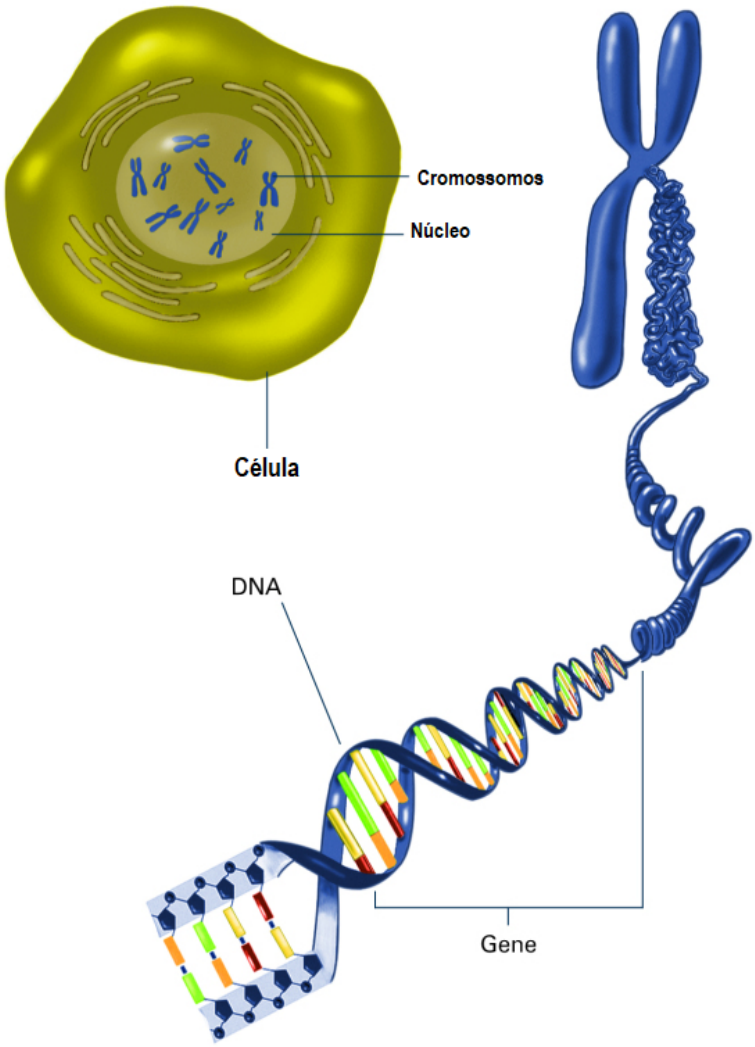
Fonte: http://www.simbiotica.org/imagens/celulavegetal.jpg.

Figura 2.3: Exemplo de células procariontes.



Fonte: (http://portaldoprofessor.mec.gov.br/fichaTecnicaAula.html?aula=39918).

Figura 2.4: Esquema do núcleo de uma célula eucarionte.



Fonte: https://www.sobiologia.com.br/conteudos/Citologia2/nucleo5.php.

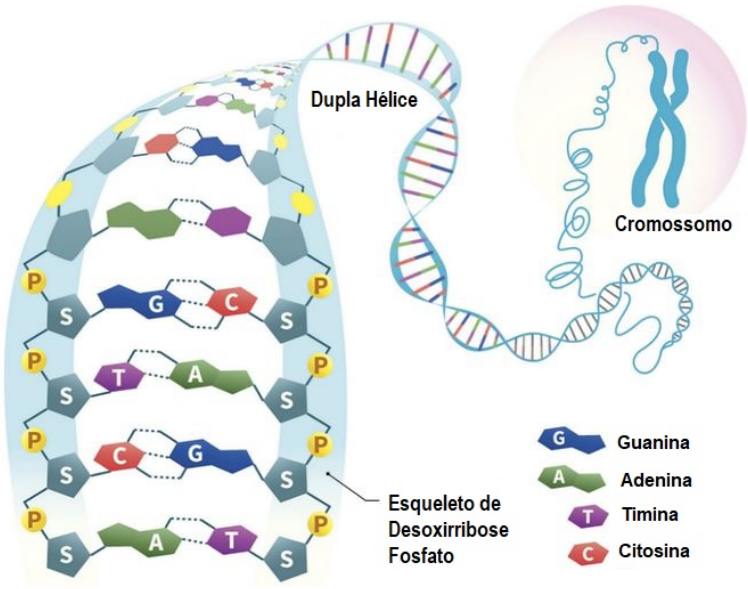
Os estudos das células na área da genômica tiveram grandes avanços desde a descoberta da estrutura básica do DNA, que junto com o RNA e as proteínas, são as três macromoléculas biológicas primordiais aos seres humanos (LEMOS; ARAGAO; CASANOVA, 2003; COHEN, 2004). O estudo dessas macromoléculas torna-se interessante, especialmente, para o entendimento do funcionamento dos organismos vivos, sendo necessária uma avaliação mais aprofundada para a compreensão das características de cada indivíduo (ADAMS et al., 1992; BAJORATH; STENKAMP; ARUFFO, 1993).

2.1.2 DNA, RNA e Proteínas

Para se compreender as estruturas e as funções dos organismos vivos é necessário um entendimento das macromoléculas de DNA, RNA e proteínas e suas interações. No contexto da Bioinformática, estas estruturas são mapeadas como biossequências, ou sequências biológicas. As cadeias que representam o DNA e o RNA são classificadas como sequências de nucleotídeos e, analogamente, as cadeias de proteínas são chamadas de sequências de aminoácidos (EIDHAMMER; JONASSEN; TAYLOR, 2000; LEMOS; ARAGAO; CASANOVA, 2003).

O DNA é a base da hereditariedade e armazena todas as informações dos indivíduos (LEMOS; BASÍLIO; CASANOVA, 2003). A sua estrutura básica é composta por uma fita dupla de nucleotídeos, os quais podem ser um das quatro bases nitrogenadas, conforme pode ser visto na Figura 2.5. As bases nitrogenadas são a Adenina (A), a Timina (T), a Citosina (C) e a Guanina (G). Essas bases se ligam por meio de uma relação conhecida como ponte de hidrogênio (WATSON; CRICK et al., 1953; LIEW; YAN; YANG, 2005).

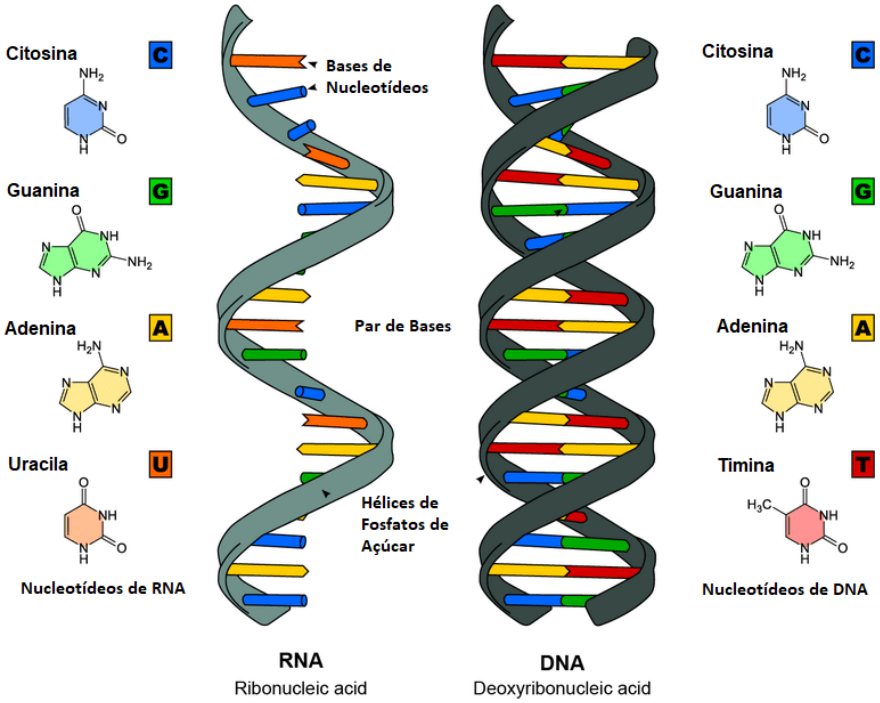
Figura 2.5: Estrutura do DNA.



Fonte (https://www.thoughtco.com/nucleic-acids-373552)

A estrutura do RNA, conforme pode ser visto na Figura 2.6, é uma fita simples e, assim como o DNA, é formada pela variação de quatro bases nitrogenadas. Porém, durante o processo enzimático chamando de Transcrição do DNA para o RNA, as posições da Timina (T) são substituídas pela Uracila (U). Na Transcrição, as informações contidas no DNA são então transcritas para as moléculas de RNA, as quais possuem o código para a ordenação dos aminoácidos. Estes são sintetizados em diferentes proteínas por meio de um processo chamado de Tradução do RNA (LEMOS; ARAGAO; CASANOVA, 2003; PROSDOCIMI et al., 2002), o que pode ser observado na Figura 2.7.

Figura 2.6: Estrutura do DNA e do RNA.



Fonte (https://www.thoughtco.com/nucleic-acids-structure-and-function-4025779)

As proteínas são sintetizadas por meio de um códon, que é a unidade do código genético composta por três nucleotídeos consecutivos e que determina a posição de um aminoácido na cadeia polipeptídica. Existem vinte aminoácidos principais, ou essenciais, que podem ser observados na Tabela 2.1 (LEMOS; CASANOVA, 2000).

Figura 2.7: O Dogma central da biologia molecular.

Fonte (PROSDOCIMI et al., 2002).

Tabela 2.1: Os vinte aminoácidos que são codificados.

Nome	**Símbolo**	**Abreviação**
Glicina ou Glicocola	Gly, Gli	G
Alanina	Ala	A
Leucina	Leu	L
Valina	Val	V
Isoleucina	Ile	I
Prolina	Pro	P
Fenilalanina	Phe ou Fen	F
Serina	Ser	S
Treonina	Thr, The	T
Cisteina	Cys, Cis	C
Tirosina	Tyr, Tir	Y
Asparagina	Asn	N
Glutamina	Gln	Q
Aspartato ou Ácido aspártico	Asp	D
Glutamato ou Ácido glutâmico	Glu	E
Arginina	Arg	R
Lisina	Lys, Lis	K
Histidina	His	H
Triptofano	Trp, Tri	W
Metionina	Met	M

Além do conjunto dos vinte aminoácidos essenciais, existem outros três conjuntos também utilizados pelos pesquisadores, porém são combinações de outros aminoácidos. Os três conjuntos podem ser vistos na Tabela 2.2.

Tabela 2.2: Conjuntos extras de aminoácidos.

Nome	Símbolo	Abreviação
Asparagina/Ácido Aspartâmico	Glx	Z
Glutamina/Ácido Glutâmico	Asx	B
Qualquer aminoácido		X

Com as descrições fundamentais da formação dos organismos vivos apresentadas, tornam-se possíveis análises para classificar esses organismos, segundo algumas características. Estas análises são apresentadas nas seções subsequentes.

2.1.3 Filogenia e Padrões

A filogenia é o estudo da evolução dos seres vivos por meio da análise da homologia entre as sequências de DNA. Homologias estão relacionadas com as similaridades resultantes da herança genética advinda de um ancestral comum. Nesse sentido, na filogenia busca-se encontrar uma árvore evolucionária dos seres vivos, partindo do pressuposto que as diferentes espécies derivam de um ancestral comum (GOULD, 1977).

Com a descoberta dos padrões que ocorrem em regiões das sequências de DNA e proteínas, possibilita-se a compreensão da relação entre essas biossequências, que pode indicar elementos importantes em suas estruturas e funções. Com isso, tornam-se viáveis estudos mais aprofundado sobre os organismos. Existem suposições de que certas regiões são melhor conservadas durante a evolução, porque são importantes para a estrutura ou função da molécula (LEMOS; ARAGAO; CASANOVA, 2003).

Além disso, outro objetivo da descoberta de padrões é a classificação das proteínas em famílias de acordo com os padrões identificados. Tais padrões somente podem ser considerados classificadores se, dada uma proteína desconhecida, ela possuir todos os padrões de uma determinada família e, portanto, será classificada como membro da família (BRAZMA et al., 1998).

Com os avanços ocorridos na biologia molecular possibilitou-se o entendimento

dos fatores e características que são herdadas pelos diversos seres vivos ao longo das gerações e neste contexto, filogenia e padrões são importantes para a bioinformática. Tanto a filogenia como a análise de padrões buscam determinar como o processo evolutivo ocorreu e as características que ao longo do curso da evolução permaneceram preservadas nos organismos (GOULD, 1977).

No entanto, a quantidade de dados biológicos disponíveis cresceu substancialmente nos últimos anos, devido aos avanços tecnológicos e aos numerosos trabalhos de pesquisas. Determinar a filogenia e reconhecer padrões nessa expressiva massa de dados é uma tarefa que consome uma quantidade considerável de recursos computacionais. Neste contexto, o problema da reconstrução da filogenia é fundamental na biologia molecular, na bioquímica e na bioinformática, para garantir boas análises em termos de significância biológica e otimizá-las (HSIEH et al., 2015; WANG; JIANG, 1994). Assim, as estratégias de alinhamento de sequências são imprescindíveis para a realização de inferências sobre esses dados (EDGAR; BATZOGLOU, 2006). Identificou-se que um dos pontos principais em um alinhamento é a construção da árvore guia, ou filogenética. Uma árvore guia bem construída proporcionará um alinhamento com maior precisão, permitindo uma análise mais próxima da relação evolutiva verdadeira (ZHAN et al., 2015; HSIEH et al., 2015).

2.2 Alinhamento de Sequências

O alinhamento de sequências é uma técnica para se identificar as regiões de similaridade em um conjunto de biossequências, o que pode ser consequência das relações funcionais, estruturais ou evolutivas (LIEW; YAN; YANG, 2005). Tal abordagem consiste em investigar por sequências homólogas ou proteínas com genes determinados, e com estruturas disponíveis em bancos de dados públicos de genomas e proteínas. Como citado anteriormente, a homologia entre duas sequências é um bom indicador que sugere um antepassado em comum. Executar a análise manual dos dados obtidos

por meio dos estudos e pesquisas biológicas é uma tarefa inviável e dispendiosa (ZAFALON, 2014), devido à quantidade de informações disponíveis nas bases de dados. Estas crescem ano após ano, especialmente com o advento dos avanços tecnológicos (LIU; SCHMIDT; MASKELL, 2010). Assim, os alinhamentos assumiram um papel preponderante para assistir os pesquisadores em suas análises.

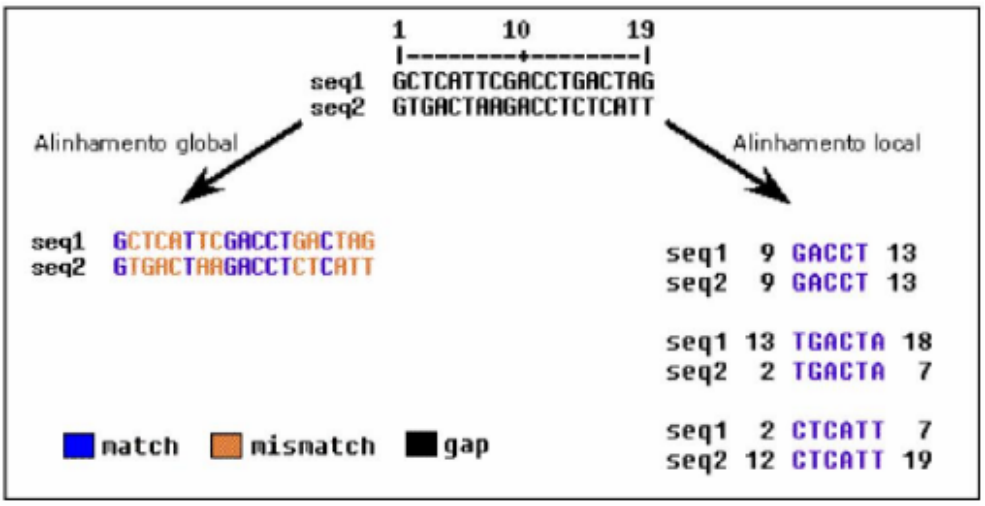
As técnicas de alinhamento de sequências são estratégias que auxiliam os biólogos na execução das diversas aferições nos grupos de biossequências com variabilidade biológica. Destacam-se como uma das ferramentas mais importante da bioinformática para análise dessas sequências (NOTREDAME; HOLM; HIGGINS, 1998). Estas técnicas são utilizadas para: reconstrução filogenética (HSIEH et al., 2015), para predição de estruturas (KOCEV et al., 2013), análises filogenéticas, predição de função e reação em cadeia da polimerase (NOTREDAME; HOLM; HIGGINS, 1998), auxiliando também no processo de reconhecimento de padrões (EDGAR; BATZOGLOU, 2006; RIDDER; RIDDER; REINDERS, 2013), no qual, atualmente, técnicas de *Deep Learning* têm sido aplicadas (GAO; ZHANG; WEI, 2018; BUSIA et al., 2018).

De uma maneira simplificada, os algoritmos que implementam as estratégias de alinhamento de sequências devem executar uma comparação par-a-par entre todos os resíduos de duas cadeias de caracteres que representam os nucleotídeos ou aminoácidos. O objetivo é localizar as regiões de igualdade (*matches*). Para concluir o seu objetivo, o algoritmo pode promover rearranjos nas biossequências, com inserção de espaços, também conhecidos como *gaps*, nas regiões em que diferenças (*mismatches*) forem localizadas. A ideia é obter-se um resultado que reflita o maior nível de similaridade, ou a menor distância entre as sequências e, para isso, alguma função que possa mensurar o nível de identidade das sequências deve ser utilizada (EDGAR; BATZOGLOU, 2006). Tal função, deve usar uma medida, ou esquema de pontuação, que pondere a quantidade de *matches* e de *mismatches*, recompensando os *matches* e penalizando os *mismatches*. Assim, ao fim do processo de alinhamento, a melhor solução

ou resultado ótimo será produzido com a maior pontuação possível (EDGAR; BATZOGLOU, 2006; KHURI, 2008).

Pode-se observar que o alinhamento de sequências é um problema de otimização e, para o qual, as estratégias que utilizam a Programação Dinâmica produzem os melhores resultados, ou seja, a melhor pontuação ou alinhamento exato. Dentre as abordagens existentes, as mais utilizadas são a de Needleman-Wunsch (NEEDLEMAN; WUNSCH, 1970), para alinhamentos globais, e a de Smith-Waterman (SMITH; WATERMAN, 1981), para alinhamentos locais. Exemplos de resultados dessas abordagens são apresentados na Figura 2.8 e melhor explicados nas próximas seções. No entanto, ambas estratégias possuem alto custo computacional, o que as inviabiliza para operações com mais do que duas sequências.

Figura 2.8: Alinhamento global e alinhamento local.



Fonte: (PROSDOCIMI et al., 2002).

2.2.1 Algoritmo de Needleman-Wunsch

Trata-se do algoritmo que busca por uma solução com o maior nível de similaridade, baseando-se nas sequências como um todo. Desta forma, nas estratégias que usam alinhamentos globais, o objetivo é analisar as sequências envolvidas inteiras e não apenas localizar pontos em particular (ZAFALON, 2014).

O algoritmo de Needleman-Wunsch é a abordagem mais conhecida e que utiliza a Programação Dinâmica para alinhar globalmente sequências de nucleotídeos ou aminoácidos (NEEDLEMAN; WUNSCH, 1970), utilizando uma abordagem determinística. Esta sempre retornará o mesmo resultado, para o mesmo conjunto de entrada, com o melhor alinhamento possível para duas sequências dadas (GUSFIELD, 1997).

Resumidamente, dadas duas sequências de entrada, uma matriz de pontuação e uma função para preencher a matriz são fornecidas ao algoritmo, que então executa um alinhamento par-a-par das sequências fornecidas. A matriz deve ter tamanho (*N, P*), em que *N* é o tamanho da maior sequência e *P* é o tamanho da menor sequência. Para que o algoritmo produza o resultado desejado, ou seja, o melhor alinhamento possível, a matriz de pontuação deve ser preenchida de forma que quando um *match* for encontrado, uma pontuação positiva seja atribuída à posição da matriz que corresponda aos dois resíduos comparados e quando um *gap* precisar ser inserido, ou um *mismatch* for localizado, uma pontuação negativa seja atribuída a essa posição (ZAFALON, 2014; NAIDU; NARAYANAN, 2016). Os caracteres das sequências são dispostos na primeira linha e primeira coluna da matriz e o algoritmo percorre todos os elementos da matriz, fazendo uma comparação de todos contra todos, preenchendo a matriz com as pontuações resultantes de cada comparação. Ao término do preenchimento da matriz, realiza-se uma escalada reversa (*backtracking*) para a obtenção do melhor alinhamento, conforme observa-se no exemplo da Figura 2.9 (NEEDLEMAN; WUNSCH, 1970), iniciando pela última posição da matriz e seguindo para a próxima de maior pontuação, podendo ser a posição da diagonal, a superior ou anterior esquerda.

Como mencionado anteriormente, o algoritmo de Needleman-Wunsch é uma estratégia adequada quando aplicada aos alinhamentos globais, em que se busca por similaridade nas sequências inteiras. No entanto, mesmo em sequências com pouca similaridade, existem regiões que são de interesse para os pesquisadores e nesses casos,

Figura 2.9: A matriz de pontuação gerada pelo algoritmo de Needleman-Wunsch.

1. A matriz de pontuação

		G	A	A	T	T	C	A	G	T	T	A
	0	-4	-8	-12	-16	-20	-24	-28	-32	-36	-40	-44
G	-4	5	1	-3	-7	-11	-15	-19	-23	-27	-31	-35
G	-8	1	4	0	-4	-8	-12	-16	-14	-18	-22	-26
A	-12	-3	6	9	5	1	-3	-7	-11	-15	-19	-17
T	-16	-7	2	5	14	10	6	2	-2	-6	-10	-14
C	-20	-11	-2	1	10	13	15	11	7	3	-1	-5
G	-24	-15	-6	-3	6	9	12	14	16	12	8	4
A	-28	-19	-10	-1	2	5	8	17	13	15	11	13

2. A escalada reversa

		G	A	A	T	T	C	A	G	T	T	A
		←	←	←	←	←	←	←	←	←	←	←
G	↑	↖	←	←	←	←	←	←	←	←	←	←
G	↑	↑	↖	←	←	←	←	←	↖	←	←	←
A	↑	↑	↖	↖	←	←	←	←	←	←	←	↖
T	↑	↑	↑	↑	↖	←	←	←	←	←	←	←
C	↑	↑	↑	↑	↑	↖	↖	←	←	←	←	←
G	↑	↑	↑	↑	↑	↑	↖	↖	↖	←	←	←
A	↑	↑	↑	↖	↑	↑	↑	↖	←	↖	←	↖

3. O alinhamento produzido

G	A	A	T	T	C	A	G	T	T	A
|			|		|		|			|
G	G	A	T	–	C	–	G	–	–	A

Fonte: Adaptado de (ZAFALON, 2014).

a abordagem mais adequada que pode ser aplicada são os algoritmos de alinhamentos locais (SMITH; WATERMAN, 1981).

2.2.2 Algoritmo de Smith-Waterman

Os alinhamentos locais têm por pressuposto localizar pontos de similaridade específicos nas duas sequências analisadas. Estes pontos de similaridades locais podem representar uma característica procurada por pesquisadores durante determinada análise e que são muito utilizados na indústria farmacêutica, bioquímica, genômica, entre outras, pois podem indicar locais em que uma determinada molécula pode atuar (ZAFALON, 2014).

O algoritmo de Smith-Waterman (SMITH; WATERMAN, 1981) é uma abordagem que se baseia no algoritmo de Needleman-Wunsch (NEEDLEMAN; WUNSCH, 1970), sendo a abordagem mais conhecida e utilizada para alinhamentos locais. Assim

como o algoritmo de Needleman-Wunsch, a estratégia de Smith-Waterman é determinística e utiliza Programação Dinâmica para obter sempre o melhor alinhamento para as sequências fornecidas (ZAFALON, 2014). Desta forma, os algoritmos compartilham diversos pontos, como uma matriz de pontuação, preenchida de acordo com uma função para pontuar os *matches* e penalizar os *gaps* e *mismatches*. A principal diferença entre os dois, é que no algoritmo de Smith-Waterman somente valores positivos são atribuídos na matriz de pontuação. Quando uma pontuação negativa ocorre, o valor 0 é atribuído ao local da matriz correspondente ao par de resíduos comparados, conforme demonstra-se na Figura 2.10 (SMITH; WATERMAN, 1981; GOTOH, 1982).

Figura 2.10: A matriz de pontuação gerada pelo algoritmo de Smith-Waterman.

1. A matriz de pontuação

		G	G	A	T	C	G	C	C
	0	0	0	0	0	0	0	0	1
G	0	5	5	0	0	0	5	0	0
G	0	5	10	8	6	4	5	3	1
A	0	0	8	15	13	11	9	7	5
C	0	0	6	13	13	18	16	14	12
T	0	0	4	11	18	16	16	14	12
C	0	0	2	9	16	23	21	21	19
A	0	0	0	7	14	21	21	19	19
G	1	5	5	5	12	19	26	24	22
T	2	0	3	3	10	17	24	24	22
T	3	0	1	1	8	15	22	22	22
G	4	8	6	4	6	13	20	20	20

2. A escalada reversa

		G	G	A	T	C	G	C	C
		←	←	←	←	←	←	←	←
G	↑	↖	←	←	↖	↖	↖	←	↑
G	↑	↑	↖	←	←	←	↑	↖	←
A	↑	↑	↑	↖	←	←	←	←	←
C	↑	↖	↑	↑	↖	↖	←	←	←
T	↑	↖	↑	↑	↖	↖	↖	↖	↖
C	↑	↖	↑	↑	↑	↖	←	←	←
A	↑	↖	↑	↑	↑	↑	↖	↖	↖
G	↑	←	←	↑	↑	↑	↖	←	←
T	↑	↑	↖	↖	↑	↑	↑	↖	↖
T	↑	←	↑	↖	↑	↑	↑	↖	↖
G	↑	←	←	←	↑	↑	↑	↖	↖

2. O alinhamento produzido

G	G	A	–	T	C	–	G
|	|	|		|	|		|
G	G	A	C	T	C	A	G

Fonte: Adaptado de (ZAFALON, 2014).

Da mesma forma que ocorre no algoritmo Needleman-Wunsch, ao finalizar o pre-

enchimento da matriz de valores, realiza-se uma escalada reversa (*backtracking*) para a obtenção dos locais de alinhamento, iniciando sempre pelo maior valor da matriz e terminando o rastreamento quando a posição inicial da matriz for localizada (ZAFALON, 2014).

Os algoritmos de Needleman-Wunsch e Smith-Waterman foram idealmente desenvolvidos para alinhamentos de pares de sequências utilizando-se da Programação Dinâmica, que é uma abordagem com alto custo computacional e, como mencionado anteriormente, pode compreender uma complexidade de tempo e espaço $O(L^N)$ (WATERMAN; SMITH; BEYER, 1976), onde L representa o tamanho da sequência e N representa o número de sequências do conjunto analisado, com $1 \geq N \leq \infty$. Isso é um ponto relevante e que impossibilita o seu uso com conjuntos com três ou mais sequências (WANG; JIANG, 1994). Este fato apresenta-se como um problema na conjuntura atual da Bioinformática, visto que cada vez mais deseja-se realizar alinhamentos com inúmeras sequências, dos mais variados organismos, simultaneamente (COHEN, 2004), especialmente devido aos resultados providos pelos sequenciadores NGS (GLENN, 2011; LIU et al., 2012).

Para contornar esse problema, que se apresenta com um desafio na área de Bioinformática, surgiram várias estratégias baseadas em heurísticas, que são os alinhamentos múltiplos de sequências. No entanto, todas as estratégias que se utilizam dessas heurísticas, possuem em comum o mesmo problema, que é a necessidade de balancear os resultados para que se possa entregar alinhamentos com significância biológica e desempenho de execução do algoritmo aceitáveis (EDGAR; BATZOGLOU, 2006; COHEN, 2004). Neste ponto, técnicas de reconstrução filogenética e rearranjo de árvores, surgem como uma possibilidade de otimizar os resultados obtidos, por meio de abordagens que visam melhorar a árvore guia e produzir, consequentemente, melhores resultados com alinhamentos de sequências.

2.3 Alinhamento Múltiplo de Sequências

Sabe-se que a abordagem usando Programação Dinâmica para alinhar múltiplas sequências simultaneamente, produz um alinhamento exato. No entanto, é uma abordagem impraticável, devido ao alto custo computacional destes algoritmos (LIU; SCHMIDT; MASKELL, 2010; WANG; JIANG, 1994).

Neste contexto, os algoritmos de Alinhamento Múltiplo de Sequências referem-se a uma série de soluções heurísticas para o alinhamento de sequências, levando em consideração eventos evolutivos como mutações, inserções, exclusões e rearranjos dos resíduos biológicos, sob certas condições (CHATZOU et al., 2015). Esta é uma tarefa básica e de importância central na Bioinformática e tem muitas aplicações nas pesquisas e análises biológicas, como a inferência filogenética e a predição da estrutura da proteína 3D (ZHAN et al., 2015; OGDEN; ROSENBERG, 2006). Considera-se que, durante o processo evolutivo, resíduos podem ser inseridos ou removidos do DNA de um determinado organismo, o que se conhece com *indels*, gerando uma mutação e, portanto, uma nova ramificação na árvore evolutiva.

Com o objetivo de amenizar o problema do custo computacional, diversas pesquisas focam no desenvolvimento dessas heurísticas, que têm suas bases na teoria das probabilidades e que possibilitam a construção de alinhamentos múltiplos de sequências, na maioria das vezes, com uma boa significância biológica e com custo computacional viável (EDGAR; BATZOGLOU, 2006). No entanto, devido a esse caráter probabilístico, existem perdas no resultado final do alinhamento, mas que geralmente são aceitáveis. Mesmo assim, várias pesquisas buscam formas de melhorar o resultado final, como por exemplo as técnicas de reconstrução filogenética e rearranjo de árvores.

Existem diversas heurísticas aplicadas ao contexto da Bioinformática que podem ser aplicadas a diversos cenários do problema do Alinhamento Múltiplo de Sequências, que se discute a seguir.

2.3.1 Simulated Annealing

Simulated Annealing (SA) (KIRKPATRICK et al., 1983) é uma heurística que se baseia na analogia com a termodinâmica, sendo uma metáfora do processo térmico *annealing* ou recozimento, no qual o material é resfriado lentamente de forma que, enquanto sua estrutura congela, ocorra um estado de energia mínima. É uma estratégia comumente utilizada em problemas de otimização que envolvem a busca por um máximo global. De forma análoga ao recozimento, o algoritmo define um ponto inicial *i* em um estado *j*, gerando um ponto de vizinhança *i'* do ponto *i* e movendo-se do ponto *i* para o *i'* usando um critério probabilístico que é dependente da *temperatura* no estado *j*. Essa temperatura é análoga à do recozimento físico e é usada como parâmetro de controle. Se a solução encontrada em *i'* é melhor que a solução existente, então este novo ponto é aceito e o anterior descartado. Se a nova solução é pior do que a solução existente, então a probabilidade de aceitar o ponto é definida como *exp(-(f (i') - f (i)) / T (j))*, em que *f(.)* é o valor da função objetivo em um determinado ponto, e *T(j)* é a temperatura no estado *j*. Depois que um certo número de pontos de vizinhança é avaliado, a temperatura é diminuída e o novo estado é *j+1* é criado. Devido à forma exponencial, a probabilidade de aceitação de um ponto de vizinhança é maior em alta temperatura, e é menor à medida que a temperatura é reduzida. Desta forma, o algoritmo procura em muitos pontos de vizinhança no início, mas em um número menor de pontos à medida que a temperatura é reduzida e converge para uma solução global ótima (AMARAN et al., 2016).

2.3.2 Busca Tabu

A Busca Tabu (*Tabu Search*) (RIAZ; WANG; LI, 2004) é um método de otimização linear combinado com estratégias de inteligência artificial, indicado para problemas de decisões críticas e que necessitam de uma vasta busca no seu espaço de soluções (GLOVER; LAGUNA, 1999). Pode ser aplicado no alinhamento global de bi-

ossequências ou na tentativa de refinamento em descoberta de padrões. Utiliza-se de estruturas de memória de curto e longo prazo no processo de busca, que permitem explorar, além da otimização local, regiões do espaço de pesquisa. Em sua forma básica é um procedimento de busca de vizinhança modificado que emprega memória adaptativa para acompanhar o histórico de solução relevante, juntamente com estratégias para explorar essa memória (AMARAN et al., 2016).

2.3.3 Algoritmos Genéticos

Algoritmos Genéticos (GA) (NOTREDAME; HIGGINS, 1996; GONDRO; KINGHORN, 2007) são inspirados na Teoria da Evolução das Espécies e utilizam conceitos evolutivos, como hereditariedade, mutação, recombinação e seleção natural (REEVES, 1997). São muito aplicados em problemas de busca e otimização com vasto espaço de soluções (SCHWEFEL; RUDOLPH, 1995) e em reconhecimentos de padrões (PAL; WANG, 2017). Em alinhamentos múltiplos de sequências, são utilizados por obter bons resultados em termos de significância biológica e os elementos que representam as soluções possíveis são submetidos aos processos de mutação, recombinação e seleção gênica, para evoluir sua população e, então, são mensuradas por uma função objetivo (GOLDBERG; HOLLAND, 1988). Em geral, com um algoritmo genético cria-se uma população de cadeias e cada uma dessas cadeias é chamada de cromossomo. Cada uma dessas cadeias cromossômicas é basicamente um vetor de pontos no espaço de busca. Novos cromossomos são criados usando funções de seleção, mutação e cruzamento. O processo de seleção é guiado pela avaliação da aptidão ou função objetivo de cada cromossomo e seleção dos cromossomos, que é feita de acordo com seus valores de aptidão. Cromossomos adicionais são então gerados usando funções de cruzamento e mutação. As funções de cruzamento e mutação asseguram que uma diversidade de soluções seja mantida. Algoritmos genéticos são comumente utilizados porque são fáceis de implementar e adotados em vários pacotes comerciais de software

de otimização e simulação (AMARAN et al., 2016).

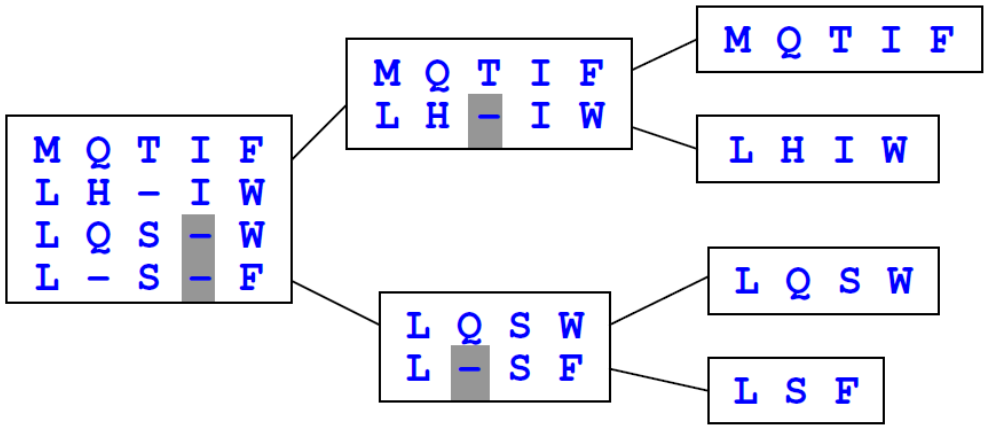
2.3.4 Colônia de Formigas

Colônia de Formigas (MOSS; JOHNSON, 2003) é uma técnica bioinspirada comumente utilizada para a resolução de problemas de otimização e busca de caminho mínimo, que se apoia a um modelo matemático consistente que permite obter bons resultados em tempo hábil (PAPADIMITRIOU; STEIGLITZ, 1998). Nesta heurística, busca-se entender a maneira como as formigas estabelecem o melhor caminho do ninho até as fontes de comida (DORIGO; BLUM, 2005; LÓPEZ-IBÁÑEZ; STÜTZLE; DORIGO, 2016). O algoritmo segue o padrão de forrageamento de formigas em busca de alimento, no qual a identificação do caminho é feita pelas formigas por meio de uma decisão probabilística, usando seus traços de feromônio que agem como mensageiros. É aplicada em alinhamentos múltiplos de sequências, de modo que o alinhamento é feito pelas formigas que atravessam a sequência para coincidir com os resíduos correspondentes, até que uma solução seja encontrada e o feromônio é atualizado (KAUR; CHAND, 2016).

2.3.5 Alinhamento Progressivo

O Alinhamento Progressivo (FENG; DOOLITTLE, 1987), esquematizado na Figura 2.11, é uma das técnicas mais utilizadas, sendo a primeira estratégia de construção prática do AMS e é a chave para a maioria dos programas de AMS (WANG et al., 2015). Utiliza-se de uma abordagem capaz de produzir resultados com boa significância biológica, em um tempo computacional $O(S^2)$ (SIEVERS et al., 2011), onde S representa o número de sequências do conjunto analisado, com $1 \geq S \leq \infty$. O Alinhamento Progressivo usa fatores probabilísticos combinados com algoritmos de Programação Dinâmica, com o objetivo de resolver ou minimizar o problema do alto custo computacional do Alinhamento Múltiplo de Sequências.

Figura 2.11: Exemplo de um Alinhamento Progressivo.



Fonte: (EDGAR, 2004a).

O Alinhamento Progressivo é uma técnica implementada em ferramentas amplamente utilizadas por pesquisadores da área de Bioinformática, como as da família Clustal (HIGGINS; SHARP, 1988; SIEVERS; HIGGINS, 2018), a T-COFFEE (NOTREDAME; HIGGINS; HERINGA, 2000), a MAFFT (KATOH; STANDLEY, 2013; KATOH; ROZEWICKI; YAMADA, 2017), a Kalign (LASSMANN; SONNHAMMER, 2005), a MUSCLE (EDGAR, 2004b), entre outras.

Em sua abordagem original, constrói-se uma árvore guia comparando iterativamente todas as sequências do conjunto de entrada entre si. Isto é utilizado para calcular uma matriz de pontuação, também conhecida como matriz de distâncias, de modo a mensurar as distâncias entre todas as sequências do conjunto. Nesta comparação, um alinhamento inicial par-a-par é construído por meio do algoritmo de Needleman-Wunsch (NEEDLEMAN; WUNSCH, 1970) ou por meio de uma comparação vetorial rápida como a k-tupla (GUERRA; BUCKLER, 2017), que é implementado pela MAFFT, a MUSCLE e a T-Coffee (CHATZOU et al., 2015).

A árvore guia, também conhecida como árvore filogenética, é construída para representar o relacionamento evolutivo entre sequências de entrada (FENG; DOOLITTLE, 1987), com o auxílio da matriz de distâncias. As árvores guia, para o Alinhamento Progressivo, geralmente são obtidas por abordagens simples e baseadas em mé-

todos que medem a distância evolutiva entre as sequências, como Neighbor-Joining ou UPGMA (GUSFIELD, 1997). Por fim, existe o alinhamento dos perfis das sequências evolutivamente mais próximas, utilizando-se da árvore guia para alinhar primeiro as sequências que possuem as menores diferenças entre si e que, portanto, encontram-se mais próximas na árvore guia (WANG et al., 2015).

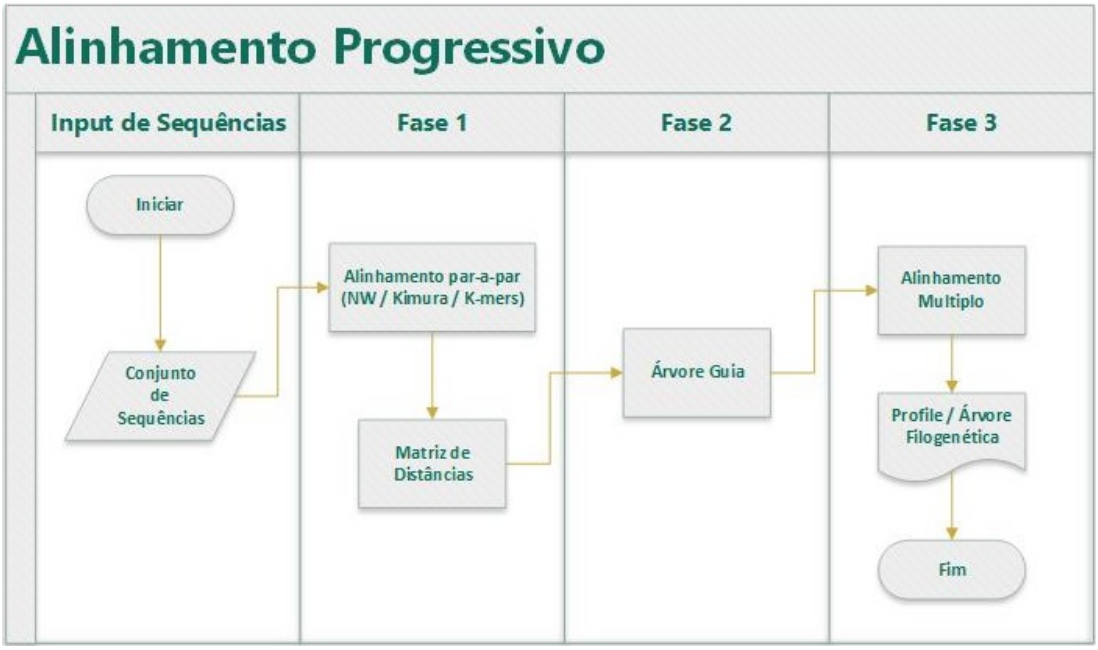
Como percebe-se, um algoritmo que aborda a heurística de Alinhamento Progressivo segue um pipeline de execução bem definido, que pode ser dividido em três fases (WALLACE; BLACKSHIELDS; HIGGINS, 2005; WANG et al., 2015), como observa-se na Figura 2.12:

1. **Alinhamento par-a-par e construção da matriz de pontuação:** cada possível par de sequências do conjunto de entrada é alinhado e calculam-se as distâncias evolutivas destes pares, com algoritmos de Programação Dinâmica e alguma função objetivo, por exemplo, como soma de pares ponderada *(WSP - Weighted Sum-of-Pairs)* (ALTSCHUL; CARROLL; LIPMAN, 1989). Em um conjunto com S sequências, haverá $S*(S-1)/2$ pares. O resultado desta etapa é uma matriz de pontuação preenchida com os valores das distâncias e que será utilizada na próxima fase do pipeline;

2. **Construção da árvore guia:** a matriz calculada na fase anterior serve de entrada para esta fase, que é responsável pela construção da árvore guia. Esta agrupa as sequências de acordo com a ordenação da matriz de pontuação, utilizando-se de algum método de arranjo (XU; TIAN, 2015), sendo que as menos distantes são agrupadas primeiro. A árvore guia servirá de entrada na etapa posterior e há diversos métodos para a sua construção, como o *UPGMA (Unweighted Pair Group Method using Arithmetic averages)* (SOKAL, 1958) ou *NJ (Neighbor-Joining)* (SAITOU; NEI, 1987), que agrupam sucessivamente as sequências mais similares, conforme observa-se na Figura 2.13. Nesta, pode-se concluir que humanos e macacos são as raças mais similares e, portanto, são agrupadas.

3. **Alinhamento múltiplo:** com a árvore guia construída na fase anterior, o algoritmo conduz o alinhamento pelos grupos de sequências mais similares, que na árvore guia estão agrupadas, ou seja, inicia selecionando as duas sequências mais similares, cuja pontuação de distância é a menor na matriz. Em seguida, na árvore guia é selecionada a próxima sequência mais similar às duas anteriores, que é incorporada ao alinhamento. Desta forma, todas as sequências são adicionadas ao alinhamento, até que a última sequência seja agrupada e o alinhamento final seja obtido. O processo executado $n-1$ vezes, em que n é o número de sequências do conjunto de entrada.

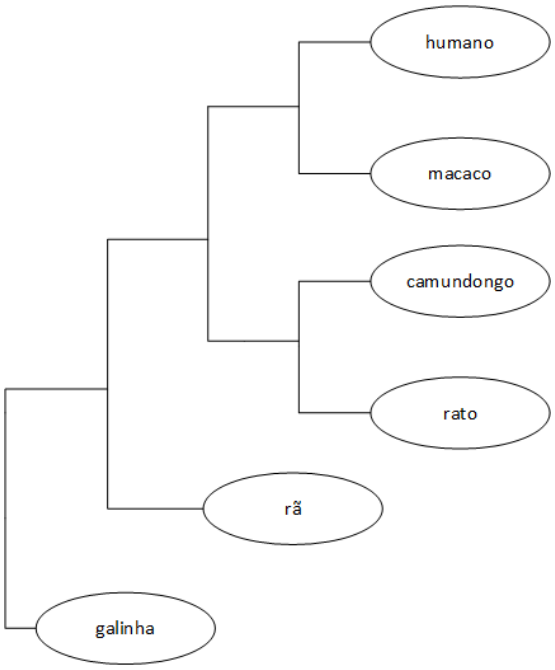
Figura 2.12: As etapas do Alinhamento Progressivo.



Fonte: Elaborada pelo autor.

O Alinhamento Progressivo é uma estratégia que na maioria das vezes é capaz de alinhar conjuntos de algumas milhares de sequências com comprimentos variados. Embora possa produzir resultados com boa significância biológica e em tempo factível, enfrenta dificuldades com conjuntos na escala milhões de sequências (SIEVERS et al., 2011).

Figura 2.13: Exemplo de árvore guia.



Fonte: Elaborada pelo autor.

Acrescenta-se a isso uma forte dependência da ordem em que as sequências estão posicionadas no conjunto de entrada (BOYCE; SIEVERS; HIGGINS, 2015). Isto torna as técnicas de reconstrução da filogenia e rearranjo da árvore guia um importante campo de estudos, a fim de resolver o problema da ordem das sequências na árvore e, consequentemente, a sua acurácia. Assim, permite-se aproximar o máximo possível da árvore verdadeira, a qual refletirá as mutações ocorridas com as espécies durante o processo evolutivo (HSIEH et al., 2015).

2.4 O Princípio da Evolução Mínima

O princípio da evolução mínima (ME) (KIDD; SGARAMELLA-ZONTA, 1971) é uma das primeiras abordagens para construção de árvores guia baseadas em uma matriz de distâncias. Nela os comprimentos dos ramos são atribuídos a várias topologias de árvores, a fim de selecionar dentre todas as possíveis topologias, aquela que possui a menor soma dos comprimentos dos ramos (BASTKOWSKI et al., 2015).

Esta abordagem é utilizada no contexto da Bioinformática para inferência filogenética, baseando-se na premissa de que a evolução dos seres vivos ocorre de maneira lenta e gradativa. Portanto, a árvore guia com menores distâncias entre seus nós e, consequentemente, com a menor soma dos comprimentos estimados, tem maior probabilidade de ser uma árvore evolutiva verdadeira. Desta forma, procura-se pela topologia de árvore que fornece uma soma mínima do comprimento de suas ramificações (RZHETSKY; NEI, 1993).

Os métodos que utilizam essa abordagem são muito populares, pelo fato de seu uso ser mais adequado para a construção de árvores maciças, que resultam de alinhamentos de grandes conjuntos de sequências (BASTKOWSKI et al., 2015). Uma aplicação com o propósito de determinar a posição correta de uma sequência dentro de uma dado conjunto de entrada, pode ser visto em (FILIPSKI et al., 2015), no qual apresenta-se o método *PhyClass*, que busca determinar a posição filogenética apropriada de cada sequência na árvore guia, obtendo uma boa precisão da árvore em comparação com abordagens de máxima vizinhança. Como mencionado anteriormente, uma árvore guia mais precisa, ou seja, mais próxima da árvore evolutiva verdadeira é muito importante para obter-se um alinhamento de sequencia mais acurado.

2.5 Técnicas de Reconstrução e Rearranjo de Árvores

Mesmo com os bons resultados obtidos com o uso dos métodos e ferramentas anteriormente citados, Boyce (BOYCE; SIEVERS; HIGGINS, 2015) aponta que ferramentas baseadas em Alinhamento Progressivo possuem forte dependência da ordem das sequências de entrada, o que pode gerar resultados diferentes para o mesmo conjunto de entrada, apenas alterando a ordem das sequências do conjunto. Zhan (ZHAN et al., 2015) demonstrou que melhorias na árvore guia aumentam a acurácia de diversas ferramentas de AMS. Com essas questões, destaca-se a importância das técnicas que visam melhorar a qualidade da árvore guia, com métodos de reconstrução filogenética

e rearranjo da árvore guia.

Os algoritmos de AMS usam a árvore guia com alguma técnica de Alinhamento Progressivo para gerar o resultado final, porém empregam técnicas diferentes para construir a árvore (NELESEN et al., 2008). Nos últimos anos, várias teorias e métodos que usam modelos de evolução probabilística e de mutações dos sítios entre as sequências foram propostos para melhorar a precisão e a velocidade de reconstrução da árvore guia. Como citado anteriormente, o objetivo é encontrar uma árvore guia o mais próximo possível da árvore evolutiva verdadeira, o que não é obtido na maioria dos casos.

O problema da reconstrução da árvore guia é um desafio clássico da bioinformática e possui muita relevância devido a sua utilização para classificação das espécies e para demonstrar as similaridades entre os organismos vivos. Essa classificação segue uma hierarquia que categoriza as espécies em ordem, de modo a agrupar os organismos vivos de acordo com a sua similaridade ou distância evolutiva. Tal classificação deve ser feita a partir da história de ancestralidade comum entre as espécies, com vistas a encontrar-se uma topologia para a árvore guia mais próxima da filogenia verdadeira. Com isso, pode-se classificar as sequências biológicas de acordo com suas diferenças, o que deve ser um reflexo das distâncias evolutivas entre elas (GUSFIELD, 1997; HSIEH et al., 2015). A árvore guia é muito importante na análise e busca por solução de vários problemas biológicos, como na predição da estrutura e função das proteínas e no desenvolvimento de novos medicamentos (COHEN, 2004; HSIEH et al., 2015).

Como citado anteriormente, o Alinhamento Progressivo é sensível aos problemas que ocorrem na fase do alinhamento inicial, os quais não são possíveis de se resolver nas fases posteriores do processo. Isto se deve à característica gulosa dos algoritmos, o que impacta na precisão e qualidade da árvore guia. Sendo assim, algumas técnicas de refinamento da árvore guia foram propostas com o objetivo de otimizar a sua precisão, de modo a produzir melhores resultados finais (GIRIBET, 2007), ou seja, uma

alinhamento final mais preciso e mais próximo da história evolutiva dos seres.

Desta forma, algumas heurísticas foram propostas e utilizadas em algoritmos para implementar estratégias de rearranjo de árvores, com o objetivo de encontrar uma árvore ótima, que é a mais próxima daquela conhecida como árvore evolutiva verdadeira. Com isso, almeja-se obter uma reconstrução filogenética otimizada, em um espaço de busca reduzido (HSIEH et al., 2015). A seguir, serão discutidas algumas dessas técnicas e programas de AMS que implementam árvores guia.

2.5.1 Técnicas para Rearranjo de Árvores

Na análise filogenética, os métodos numéricos são conhecidos pela sua eficiência e repetição dos resultados para um dado conjunto de análise, o que torna o seu uso preferencial. Os métodos numéricos que fazem uso de critérios para otimizar os resultados são os mais desejáveis, devido à capacidade de gerarem árvores hipotéticas e compará-las por meio de alguma função de medida, a fim de selecionar a melhor árvore. No entanto, devido ao grande espaço de busca desses métodos, eles são conhecidos pelo seu alto custo computacional, ao contrário da maioria dos métodos numéricos que não utilizam critérios de otimização (GIRIBET, 2007).

No tocante aos métodos que não se utilizam de otimização, os mais aplicados são baseados em distância *(Distance-Based Methods)* e os baseados em caracteres *(Character-Based Methods)*. O primeiro é um método numérico que mede as distâncias entre as sequências e armazena os resultados em uma matriz, a qual é usada em fases posteriores do alinhamento para construir a árvore guia. Esse é o método adotado pelo NJ e pelo UPGMA e são métodos rápidos, mas sem critérios de otimização. No entanto, informações biológicas importantes podem ser perdidas quando ocorre essa transformação dos símbolos que representam os resíduos em distâncias contadas para-par. No segundo caso, as sequências são alinhadas com base em métodos probabilísticos, que medem as diferenças entre os caracteres para construir a árvore guia. Essa

é a abordagem usada por métodos como o da *Máxima Vizinhança (ML - Maximum-Likelihood)* e pelo método da *Máxima Parcimônia (MP - Maximum-Parsimony)*. Os dois são os principais métodos que utilizam estratégias com critérios de otimização da árvore, permitindo a análise completa de conjuntos de dados complexos com milhares, ou até milhões, de sequências (HSIEH et al., 2015; GIRIBET, 2007).

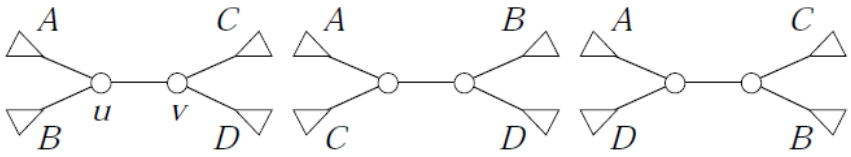
2.5.2 Estratégias para Rearranjo de Árvores

A fim de obter-se melhores resultados, uma classe de algoritmos chamados *Branch Swapping (troca de ramos)*, ou simplesmente *Swappers*, foi desenvolvida para promover a troca de ramos da árvore. O objetivo desta classe de algoritmos é melhorar uma árvore que tenha sido construída anteriormente, seja ela gerada pelo método de construção de árvore adotado pelo algoritmo ou gerada pelo método de troca de ramos utilizado no refinamento da árvore (GIRIBET, 2007).

Das diversas estratégias para rearranjo de árvores, a mais simples é o *Nearest Neighbor Interchange (NNI)* (DAY, 1983), usado pelo algoritmo *Phylogenetic Maximum Likelihood (PHYML)* (GUINDON; GASCUEL, 2003). Sabe-se que uma ramificação interna da árvore binária sem raiz possui quatro subárvores conectadas e que se pode promover trocas das subárvores e obter mais duas novas árvores, como pode-se verificar na Figura 2.14. O NNI promove essas trocas para obter as duas possíveis árvores e compara-as com a original. Desta forma, seleciona entre as três subárvores, aquela que possui a maior pontuação ou probabilidade de ser uma árvore verdadeira. Esse método tem como ponto positivo a velocidade de execução, uma vez que os movimentos podem ser calculados localmente. Seu ponto negativo é que o método pesquisa apenas localmente nas quatro possíveis subárvores e frequentemente obtém resultados ótimos locais *(local optima)*, principalmente quando analisa conjuntos de sequências de genes com sinais conflitantes (HSIEH et al., 2015; GIRIBET, 2007). Um resultado ótimo local, na maioria dos casos, não é o desejado pelos pesquisadores, pois os mé-

todos que conduzem a esses resultado acabam por ignorar as soluções ótimas globais *(global optima)*, que, em geral, são mais desejadas.

Figura 2.14: As duas possíveis operações NNI em uma ramificação interna da árvore.



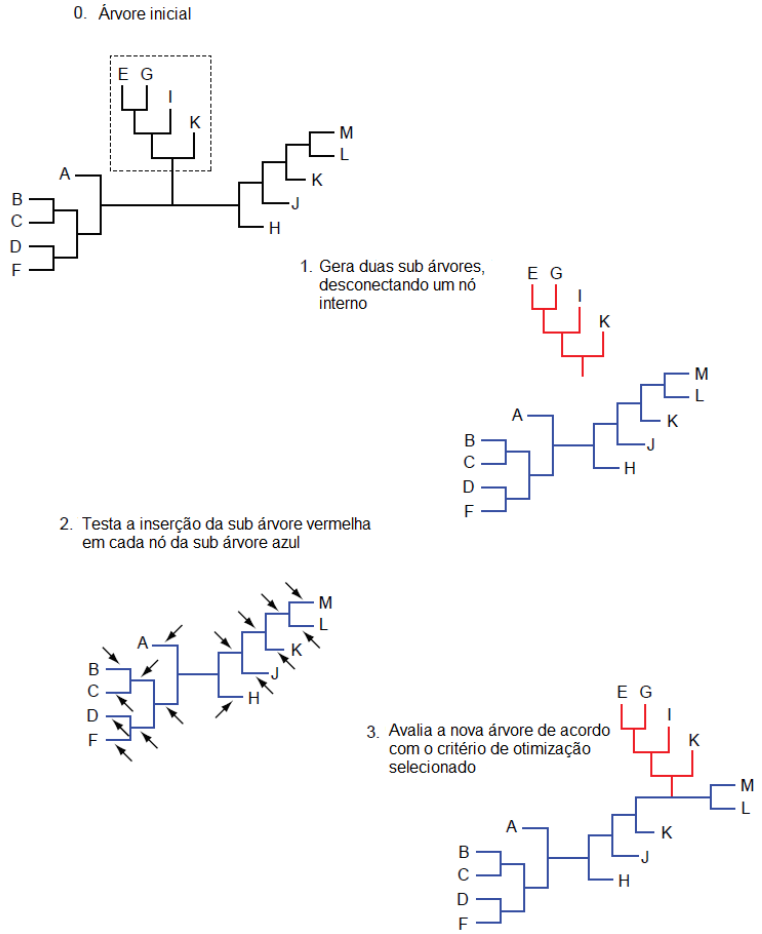
Fonte: (LI; TROMP; ZHANG, 1996).

Um método mais amplo é o *Subtree Pruning and Regrafting (SPR)* (BORDEWICH; SEMPLE, 2005), implementado pelo algoritmo *Randomized Axelerated Maximum Likelihood (RAxML)* (STAMATAKIS; HOOVER; ROUGEMONT, 2008). Na Figura 2.15, pode-se observar o funcionamento do SPR, em que uma subárvore interna aleatória é podada e então é pesquisado em toda a árvore principal qual o melhor ponto para reconectar a subárvore e produzir uma nova árvore de tamanho reduzido.

Pesquisando na árvore inteira pelo melhor ponto de reconexão, o método evita um resultado com um ótimo local ruim. Porém, a etapa 2 consome muito tempo de computação, uma vez que para cada possível ponto de reconexão da árvore, o SPR precisa recalcular toda a probabilidade da árvore a fim de comparar o resultado com a árvore anterior. Se o tamanho da nova árvore for maior que a anterior, ela é descartada, caso contrário, a árvore anterior é descarta e a nova árvore será usada nas novas iterações do algoritmo (HSIEH et al., 2015; GIRIBET, 2007; WU, 2008).

O *Tree Bisection and Reconnection (TBR)* (BRYANT, 2004) é um método de rearranjo mais elaborado que, da mesma forma que SPR, corta uma árvore em duas partes, quebrando uma ramificação interna. Em seguida, seleciona um ramo da subárvore e então reconecta-o em outra posição da árvore principal, formando uma nova árvore. Pode-se observar que ao contrário do SPR, o TBR pode modificar a topologia da subárvore podada e da árvore principal e isso deve-se ao fato de que, ao contrário do SPR em que somente a ramo podado é testado para a reconexão, o TBR testa todos

Figura 2.15: As operações SPR em uma ramificação interna da árvore.



Fonte: (GIRIBET, 2007).

os ramos internos da subárvore, como observa-se na Figura 2.16. No entanto, mesmo com o TBR não existe garantias de encontrar a árvore guia verdadeira. O TBR gera mais vizinhos do que NNI e o SPR, o que lhe permite reconstruir uma árvore guia verdadeira com mais frequência que os demais métodos (HSIEH et al., 2015; GIRIBET, 2007).

O espaço de busca do TBR é mais extenso, devido a necessidade de testar todos os possíveis pontos de reconexão e recalcular a probabilidade de cada possível árvore, o que o torna um método mais demorado do que o SPR e o NNI. Tipicamente, para um conjunto composto por n sequências, o NNI executará $2(n-3)$ iterações em busca de uma árvore melhor, enquanto que o SPR requer n^2 iterações e o TBR requer n^3.

Nem todos os pontos de reconexão geram bons resultados e evitar avaliar esses pontos é uma forma de reduzir o espaço de busca do método e melhorar o desempenho do método TBR (HSIEH et al., 2015). Isto pode ser feito baseando-se no Principio da Evolução Miníma (DESPER; GASCUEL, 2002; BORDEWICH et al., 2009). Em (HSIEH et al., 2015) foi demonstrado um método TBR melhorado que pode auxiliar outros algoritmos na construção de árvores mais precisas, com um espaço de busca reduzido. Com o método geram-se mais topologias vizinhas utilizando o ME para selecionar possíveis posições de reconexão que acelerem o tempo de busca. O ME adota uma matriz de distâncias entre as sequências biológicas, para selecionar uma árvore, cujo comprimento é mínimo e estimado dentro da estrutura de mínimos quadrados (DENIS; GASCUEL, 2003). Será abordado em mais detalhes, na seção 2.5.3, o método TBR melhorado, bem como a sua utilização para refinar árvores produzidas pelos métodos de Alinhamento Progressivo. (HSIEH et al., 2015; GIRIBET, 2007).

2.5.3 O Método TBR

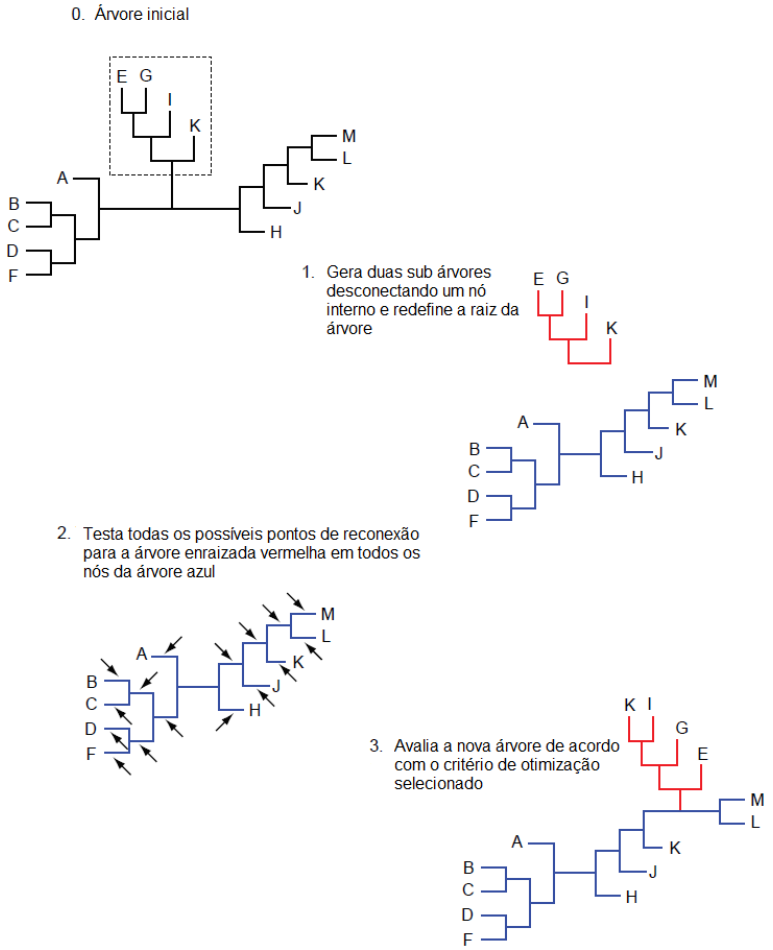
O TBR é o mais amplo dentre os três métodos citados anteriormente e pode produzir os melhores resultados. No entanto, o seu espaço de busca é maior que os demais métodos apresentados, o que gera maior consumo de tempo computacional. Isto pode levar à necessidade de busca por soluções que possam diminuir o espaço e acelerar o tempo de execução.

Nessa linha de entendimento, o método *Tree Rearrangement + Maximum Likelihood + Minimum Evolution (TRMLE)* (HSIEH et al., 2015), busca diminuir o espaço de busca com o uso da ME e eliminar testes desnecessários com soluções que não produzem resultados melhores. O método possui as seguintes etapas:

1. Modifica-se a topologia da árvore para gerar topologias vizinhas;

2. Estima-se os comprimentos dos ramos e a probabilidade da nova árvore;

Figura 2.16: As operações TBR em uma ramificação interna da árvore.



Fonte: (GIRIBET, 2007).

3. Caso a probabilidade da nova topologia seja melhor que a atual, a anterior é descartada e a nova árvore será utilizada na próxima iteração.

As etapas que podem ser observadas na Figura 2.17 são repetidas até que a probabilidade não sofra mais evolução ou uma condição de término seja satisfeita. O TBR escolhe as posições de reconexão aleatoriamente e a probabilidade da árvore resultante pode estar longe da filogenética verdadeira. Para resolver o problema, o TRMLE combina o TBR com ME para selecionar as posições corretas de reconexão e as operações do método TBR modificado são decompostas em duas fases descritas na sequência:

1. Fase 01:

(a) As arestas de uma árvore T são classificadas em ordem decrescente por seus comprimentos;

(b) A maior aresta de T (i_2, j_2) é selecionada e os seus nós finais são movidos, da seguinte forma:

   i. A subárvore de T induzida por i_2, j_2 e os descendentes de j_2 são separados de T;

   ii. Uma nova aresta (i_1, i_3) é adicionada em T;

   iii. É determinada a distância de movimento, denotado por d;

   iv. A subárvore é movida com base em d para a posição (i_d, i_{d+1});

(c) Considere T' como a árvore resultante do processo anterior (Figura 2.17 (b)):

   i. A nova distância média de T' é calculada;

   ii. A nova distância média de T' pode ser obtida ajustando com precisão a distância de T;

2. Fase 02:

(a) A subárvore J_2 de T' induzida por (i_2, j_2) e os descendentes de i_2, são separados de T';

(b) Uma nova aresta (h_1, h_3) é adicionada;

(c) É determinada outra distância de movimento, denotada por s;

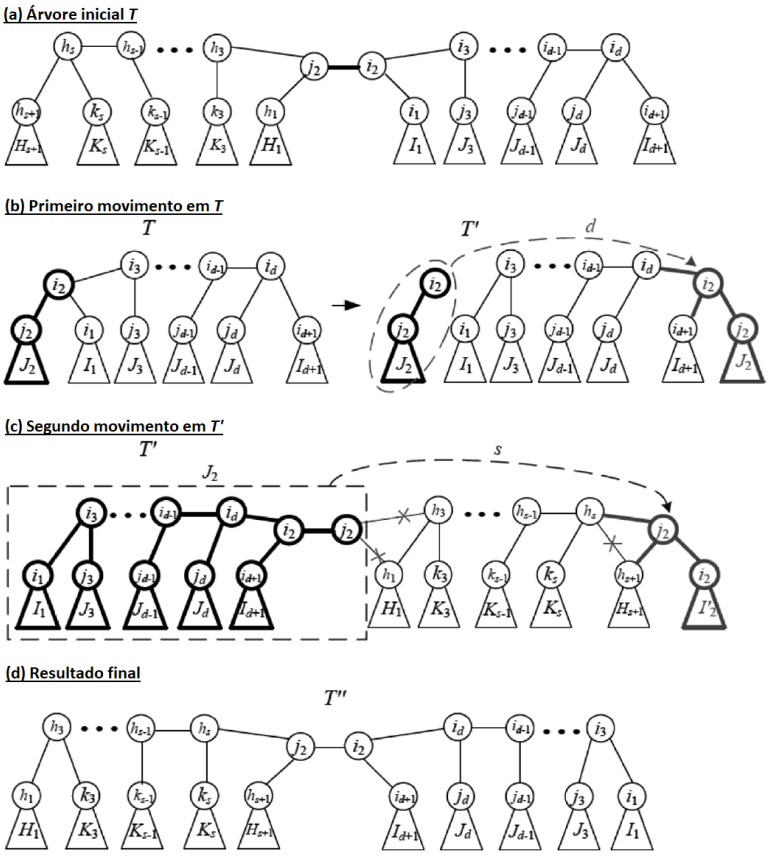
(d) A subárvore é movida com base na distância de movimento s para posição (h_s, h_{s+1}) (Figura 2.17 (c)).

(e) Considere T'' como a nova árvore resultante (Figura 2.17 (d)).

   i. A nova distância média de T'' pode ser obtida ajustando com precisão a distância de T'.

Após as fases 1 e 2, calcula-se a probabilidade de T'', denotada por $L_{T''}$. Se $L_{T''}$ for maior que a probabilidade da árvore original T, a topologia de T' está mais próxima da topologia verdadeira do que T. Quanto maior a distância entre L_T e $L_{T''}$, mais confiança tem-se no rearranjo de T para T''.

Figura 2.17: As operações da TBR modificadas.



Fonte: (HSIEH et al., 2015).

A dinâmica proposta no método TRMLE para selecionar a troca dos nós mais distantes e que tem maior probabilidade de produzir melhores resultados, demonstrou-se eficaz para melhorar a acurácia da árvore guia com consumo menor de tempo computacional. Esta característica pode auxiliar outros algoritmos conhecidos com o problema da reconstrução de árvores guia, como o MUSCLE, para produzir resultados finais com maior qualidade, maior significância biológica e sem degradar de maneira considerável o tempo computacional.

2.6 Ferramentas de AMS

As heurísticas citadas anteriormente são aplicáveis na busca por uma solução no problema do Alinhamento Múltiplo de Sequências, sendo implementadas em diversas ferramentas computacionais. Desta forma, essa seção dedica-se a expor as principais ferramentas computacionais usadas na Bioinformática e que implementam essas heurísticas.

As ferramentas atuais estão envolvidas com problema da grande quantidade de dados biológicos disponíveis, o que as obriga a uma evolução constante, principalmente devido o advento dos Sequenciadores de Nova Geração (NGS). Estes produzem volumes de dados de sequências de DNA consideravelmente superiores aos sequenciadores do tipo *Sanger*, até então muito utilizados (GLENN, 2011; LIU et al., 2012). Essa questão acaba por ser um desafio que precisa de constantes evoluções nas plataformas das ferramentas de AMS, com atenção para o melhoramento da precisão do alinhamento final e com relação ao tempo de execução e ao consumo de memória (EDGAR, 2004a; EDGAR; BATZOGLOU, 2006). Na sequência, serão discutidas algumas das ferramentas mais conhecidas e utilizadas.

2.6.1 Clustal Omega

As ferramentas da família Clustal foram implementadas por Des Higgins (HIGGINS; SHARP, 1988), que combinou um algoritmo de Programação Dinâmica com estratégia de Alinhamento Progressivo e, desde então, são amplamente aplicadas em AMS. Assim, a árvore guia também assumem um papel relevante para a operação dessas ferramentas.

Clustal Omega (SIEVERS et al., 2011) é a versão mais recente da família Clustal, que trouxe aumento considerável de escalabilidade em relação às versões anteriores, conseguindo alinhar conjuntos com centenas de milhares de sequências em algumas horas. Ela é capaz de fazer uso de multi-processadores, com qualidade de alinhamento

superiores as suas versões anteriores. Quando compara-se a Clustal Omega com outros programas de AMS de alta qualidade, usando-se conjuntos menores de sequências, a Clustal Omega pode alcançar resultados iguais. Como uso de conjuntos maiores, supera outros pacotes em termos de tempo e qualidade de execução.

A escalabilidade necessária para lidar com alinhamentos de grandes conjuntos, é obtida com a utilização do algoritmo *mBED* (BLACKSHIELDS et al., 2010) na etapa de construção da árvore guia. Isto permite obter uma complexidade *O(N log N)*, reduzindo consideravelmente o tempo de computação e os requisitos de memória necessários para agrupar grandes números de sequências e demonstrar a qualidade dos agrupamentos comparando-os com as árvores guia. Para produzir árvores guia com precisão semelhante aos métodos tradicionais, o algoritmo *mBed* trabalha incorporando cada sequência em um espaço de *n* dimensões proporcional a *log N*, substituindo cada sequência por um vetor de elementos *n*, em que cada elemento é simplesmente a distância de um dos *n* elementos. Em seguida, utilizando-se de métodos de agrupamento padrões (XU; TIAN, 2015), como *K-means* ou *UPGMA*, tais vetores são agrupados de forma rápida e eficiente. Na fase posterior de refinamento dos resultados finais, é utilizado o algoritmo *HHAlign*, que se baseia no modelo oculto de Markov *(HMM - Hidden Markov Model)*.

2.6.2 DIALIGN

DIALIGN (MORGENSTERN et al., 1998) é uma ferramenta para alinhamentos múltiplos de sequências de DNA e proteínas, que combinando recursos de alinhamento global e local, combinado com um fator probabilístico na busca por regiões de *motifs*, que podem representar importantes pontos de conservação. Essas características o torna útil para aplicação com sequências que não são alinhados de forma correta com abordagens tradicionais.

Os alinhamentos são construídos a partir de semelhanças locais das sequências

emparelhadas e é uma abordagem muito útil para descobrir regiões funcionais conservadas em sequências que compartilham apenas homologias locais, que não estão relacionadas de outra forma. Uma opção de ancoragem permite usar informações externas e conhecimentos especializados.

O alinhamento par-a-par permite obter um resultado exato ou pode utilizar abordagem estocástica para alinhamentos múltiplos de sequência. Porém, não faz uso de comparação de resíduos e nem de penalizações *(gaps)* (MORGENSTERN, 2014), baseando-se em um esquema de comparação em diagonal, como esquematizado na Figura 2.18;

A versão mais recente do DIALIGN opcionalmente usa correspondências para o banco de dados *PFAM* para detectar homologias fracas.

Figura 2.18: Cálculo de diagonal no DIALIGN.

Fonte: (MORGENSTERN et al., 1998).

2.6.3 MAFFT

A MAFFT *(Multiple Alignment using Fast Fourier Transform)* (KATOH et al., 2002) é um ferramenta de alinhamento de múltiplo sequências que possui várias opções para uso do método progressivo, do método de refinamento iterativo e de outros métodos. Comumente utilizada para Alinhamento Múltiplo de Sequências, utiliza-se da Transformada Rápida de Fourier (FFT) para alinhar sequências de nucleotídeos ou aminoácidos, obtendo alinhamentos com boa significância biológica e com custo computacional reduzido em termos de tempo de execução, quando comparada as abordagens existentes.

Com uso da FFT, as sequências de aminoácidos são convertidas em sequências compostas de volume e polaridade para cada resíduo, o que auxilia a identificação rápida de regiões homologas entre as sequências. O sistema de pontuação simplificado permite reduzir o tempo de CPU e aumentar a precisão do resultado final, mesmo em sequências muito distantes e que possuem extensas regiões com inserções.

O MAFFT implementa os métodos progressivos *FFT-NS-1 e FFT-NS-2* e o método de refinamento iterativo *FFT-NS-i*. O tempo de CPU do *FFT-NS-2* é drasticamente reduzido em comparação com a CLUSTAL, mantendo a precisão do resultado final. O *FFT-NS-i* chega a ser 100 vezes mais rápido que *T-COFFEE*, sem penalizar a precisão do resultado final, quando o conjunto das sequências de entrada excede 60 taxas (KATOH et al., 2002).

Na MAFFT, um Alinhamento Progressivo inicial *FFT-NS-1* é construído por meio de um cálculo aproximado das distâncias entre cada par de sequências do conjunto de entrada, baseado em um esquema de pontuação de *6-tuplas* compartilhadas entre as sequências sendo alinhadas. Essa matriz de distâncias inicial é utilizada para a construção de uma árvore guia por meio do método UPGMA, com tempo de UCP de $O(N^2)$, onde N é o número de sequências, e, então, são alinhadas progressivamente seguindo a ordem das ramificações da árvore guia (KATOH et al., 2005; KATOH et al., 2002).

Um segundo Alinhamento Progressivo *FFT-NS-2* é construído, porém, uma nova matriz de distâncias é calculada considerando o alinhamento construído pelo *FFT-NS-1*. Desta forma, um novo Alinhamento Progressivo baseia-se em uma árvore construída a partir da nova matriz de distância, do qual espera-se obter um resultado mais preciso (KATOH et al., 2005; KATOH et al., 2002).

O resultado obtido pelo método *FFT-NS-2* é submetido ao método de refinamento iterativo *FFT-NS-i*, a fim de obter melhorias na acurácia do alinhamento. O alinhamento é dividido em dois grupos e realinhados por meio da técnica de particionamento

restrito dependente de árvore (HIROSAWA et al., 1995). Este é repetido até que não seja mais obtido um resultado de pontuação melhor para o alinhamento em relação à pontuação anterior (KATOH et al., 2005; KATOH et al., 2002).

Atualmente o MAFFT encontra-se na versão 7 (KATOH; STANDLEY, 2013), com novas funcionalidades para interface web, opções para adicionar sequências não alinhadas em um alinhamento existente, ajuste de direção do alinhamento de nucleotídeos, alinhamento restrito e processamento paralelo por meio de *multithreading* e do uso de GPUs (*Graphics Processing Unity*) (ZHU et al., 2015).

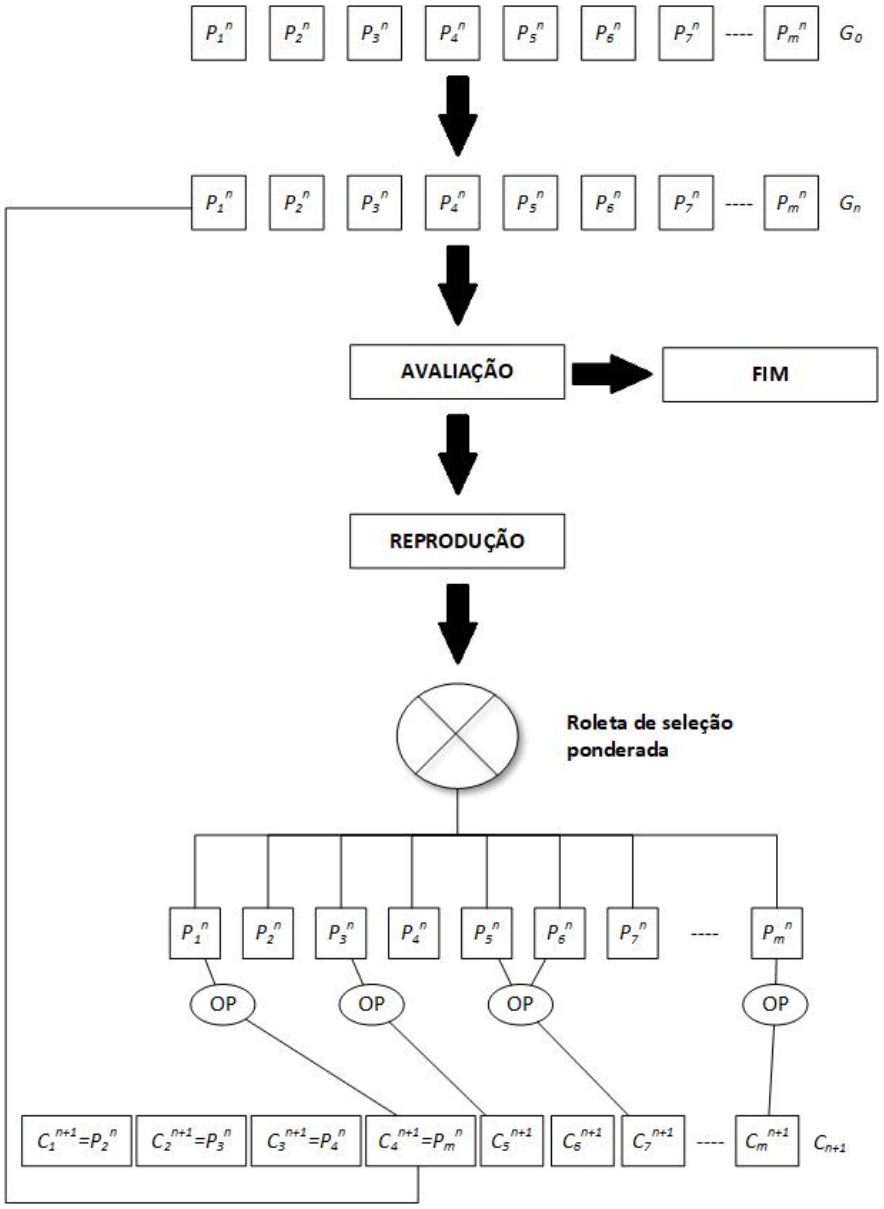
Novas funcionalidades foram adicionadas recentemente na interface web (KATOH; ROZEWICKI; YAMADA, 2017) para lidar com grandes volumes de dados e opções para refinar iterativamente os resultados do AMS. Funções adicionais permitem a seleção interativa de sequências e fazer inferência filogenética para pré-processamento e pós-processamento de MSA.

A principal limitação do MAFFAT, é a dificuldade para lidar com sequências pouco relacionadas ou biologicamente distantes.

2.6.4 SAGA

A SAGA é uma abordagem híbrida que usa Programação Dinâmica como parte do processo, sendo a primeira ferramenta que implementou a heurística do algoritmo genético para alinhamentos múltiplos de sequências. Na Figura 2.19 demonstra-se o funcionamento da ferramenta, que utiliza as técnicas da heurística de algoritmo genético para evoluir a população de alinhamentos até obter-se uma condição de parada (NOTREDAME; HIGGINS, 1996);

Figura 2.19: Esquema do algoritmo implementado na SAGA.



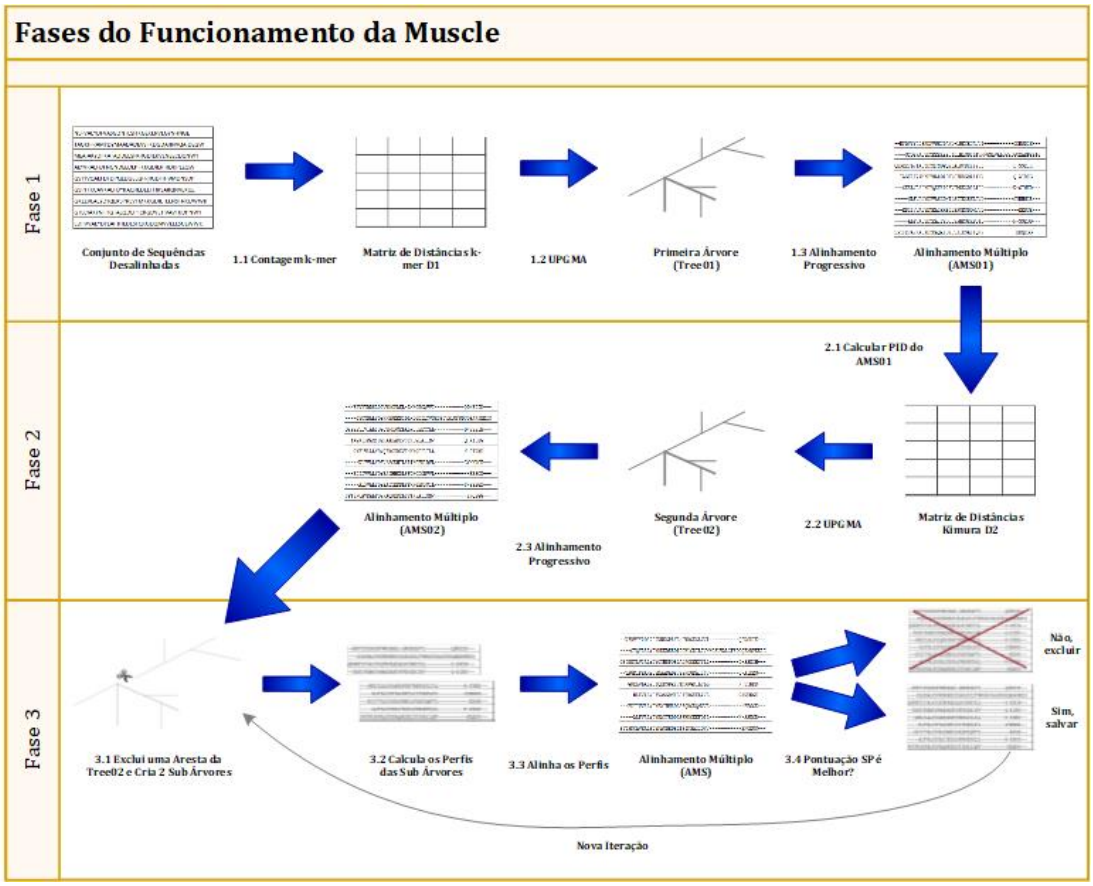
Fonte: Adaptado de (NOTREDAME; HIGGINS, 1996).

2.6.5 MUSCLE

A MUSCLE *(Multiple Sequence Comparison by Log-Expectation)* é uma ferramenta para Alinhamento Múltiplo de Sequências que implementa a heurística do Alinhamento Progressivo e que possui três estágios de execução esquematizados na Figura 2.20. Ao final da execução de cada estágio, o algoritmo disponibiliza um alinhamento múl-

tiplo. A sua estratégia básica é semelhante à utilizada pela MAFFT, no qual refinamentos horizontais são aplicados ao Alinhamento Progressivo, fornecendo melhorias significativas na precisão e velocidade (EDGAR, 2004b).

Figura 2.20: Esquema de funcionamento do algoritmo MUSCLE.



Fonte: Adaptado de (EDGAR, 2004b).

O primeiro estágio cria um rascunho de Alinhamento Progressivo que será melhorado nos próximas estágios. Utiliza-se de uma medida de similaridade computada entre todos os possíveis pares de sequências do conjunto de entrada, por meio de um método probabilístico de contagem *k-mer* (GUERRA; BUCKLER, 2017), ou construindo o alinhamento global dos pares por meio da medida de identidade fracionária.

A partir da medida de similaridade entre todas as sequências do conjunto, uma matriz de distância triangular é calculada considerando-se as semelhanças entre os pares. Por fim, uma árvore guia é construída a partir da matriz de distância, adotando-

se um algoritmo baseado na *UPGMA* ou *Neighbor-Joining* e uma raiz para a árvore é identificada. Finalmente, utilizando-se da árvore guia, o Alinhamento Progressivo é construído seguindo a ordem de ramificação da árvore, produzindo um alinhamento múltiplo de todas as sequências de entrada até a raiz da árvore.

O segundo estágio é uma fase que pode ser iterativa e é responsável por produzir uma melhoria progressiva do alinhamento, por meio da evolução da árvore guia. Com isso, é possível produzir um novo Alinhamento Progressivo de acordo com essa árvore. Nesse estágio, a medida de similaridade entre cada par de sequências é calculada usando a identidade fracionária computada no alinhamento múltiplo atual. Uma nova árvore é produzida por meio do cálculo de uma matriz de distâncias, utilizando-se da distância de *Kimura* e aplicando-se um método de agrupamento na matriz de distâncias (EDGAR; BATZOGLOU, 2006).

A árvore obtida é comparada com a árvore obtida na etapa anterior, que pode ser a árvore do primeiro estágio ou a árvore construída na iteração anterior do segundo estágio. A comparação identifica o conjunto de nós internos para os quais a ordem de ramificação foi alterada. O novo Alinhamento Progressivo é construído, alterando-se apenas as partes do alinhamento para os quais as ramificações internas da árvore sofreram alterações, até que alinhamento encontre a raiz. Neste ponto, o algoritmo pode terminar ou retornar para mais uma iteração do segundo estágio ou seguir para o terceiro estágio. Desta forma, se o segundo estágio for executado mais de uma vez e o número de alterações não diminuir, o processo de melhoria da árvore será considerado convergente e a iteração será encerrada.

O terceiro estágio é uma fase de refinamento iterativo que se utiliza de uma variante do particionamento restrito dependente de árvore(SAITOU; NEI, 1987). Neste, cada ramificação da árvore guia é excluída, gerando dois grupos de sequências disjuntos, para quais todos os nós serão visitados em ordem de distância decrescente a partir da raiz de cada subárvore. Com isso, pode-se extrair um perfil de cada subconjunto

do alinhamento atual para produzir um novo alinhamento perfil-a-perfil. Descartando as colunas sem resíduos *(gaps)*, os perfis são realinhados e calcula-se a pontuação da soma de pares *(SP - Sum-of-Pairs*) do novo alinhamento. Se a pontuação aumentar, o novo alinhamento é mantido, caso contrário é descartado. Se todas as arestas foram visitadas sem que uma mudança seja feita ou que um número máximo de iterações definido pelo usuário foi atingido, o algoritmo é finalizado. Caso contrário, ele retorna para uma nova iteração do terceiro estágio. As arestas visitadas em ordem decrescente de distância da raiz têm o efeito de primeiro realinhar sequências individuais, depois grupos intimamente relacionados (EDGAR, 2004a; WANG et al., 2015).

2.7 Considerações

Neste capítulo, para um bom entendimento do trabalho, apresentou-se os conceitos básicos da biologia molecular, das heurísticas para Alinhamento Múltiplo de Sequências e para reconstrução da árvore guia, bem como uma revisão bibliográfica das ferramentas mais utilizadas para o problema de rearranjo de árvores e da reconstrução filogenética.

Apresentaram-se também os resultados obtidos pelo método TRMLE, o qual será uma base para desenvolvimento do método proposto pelo presente trabalho, a fim de melhorar a acurácia das ferramentas que utilizam o Alinhamento Progressivo. Desta forma, pretende-se contribuir com a produção de alinhamentos finais com mais significado biológico, por meio de uma árvore guia com tamanho minimizado e próxima da filogenia verdadeira. Além disso, pretende-se contribuir com um método que mantenha o tempo de execução em níveis aceitáveis e que permita operar com grande volumes de dados. Isto destaca-se como de suma importância para os trabalhos dos pesquisadores, principalmente devido ao grande volume de dados biológicos que estão sendo produzidos nos últimos anos, devido ao advento dos sequenciadores NGS. No próximo capítulo, será apresentado o desenvolvimento do trabalho proposto.

Capítulo 3

Desenvolvimento do Trabalho

Este capítulo tem por objetivo apresentar os detalhes das atividades executadas para implementação do método proposto no presente trabalho.

3.1 O Método Proposto

O método proposto no presente trabalho foi nomeado de TreeRMEP, que é abreviação do termo *Rearranjo de Árvores Com o Princípio da Evolução Mínima (**T**ree **R**earrangement with **M**inimum **E**volution **P**rinciple)*.

O método utiliza-se os conceitos de refinamento de árvores encontrados em (GIRIBET, 2007), combinados, especificamente, com a estratégia TBR descrita em (HSIEH et al., 2015), que contribui para redução do espaço de busca e aceleração da procura pela melhor árvore, baseando-se no conceito do Princípio da Evolução Mínima (ME) (KIDD; SGARAMELLA-ZONTA, 1971), no qual procura-se pela topologia de árvore que fornece uma soma mínima dos comprimentos de suas ramificações (RZHETSKY; NEI, 1993) e sua premissa básica é de que a evolução das espécies ocorre de maneira lenta e progressiva. Portanto, uma árvore com distâncias mínimas entre seus nós tem maior probabilidade de ser a árvore filogenética verdadeira e representar com maior proximidade a evolução verdadeira das espécies (BASTKOWSKI et al., 2015).

Com o TreeRMEP promovem-se trocas na posição dos nós da árvore guia e assim altera-se sua topologia, com objetivo de encontrar uma árvore com tamanhos mínimos, menor que a árvore guia anterior. Isto é uma tarefa com grande consumo de tempo computacional e não são todas as mudanças que trazem melhorias.

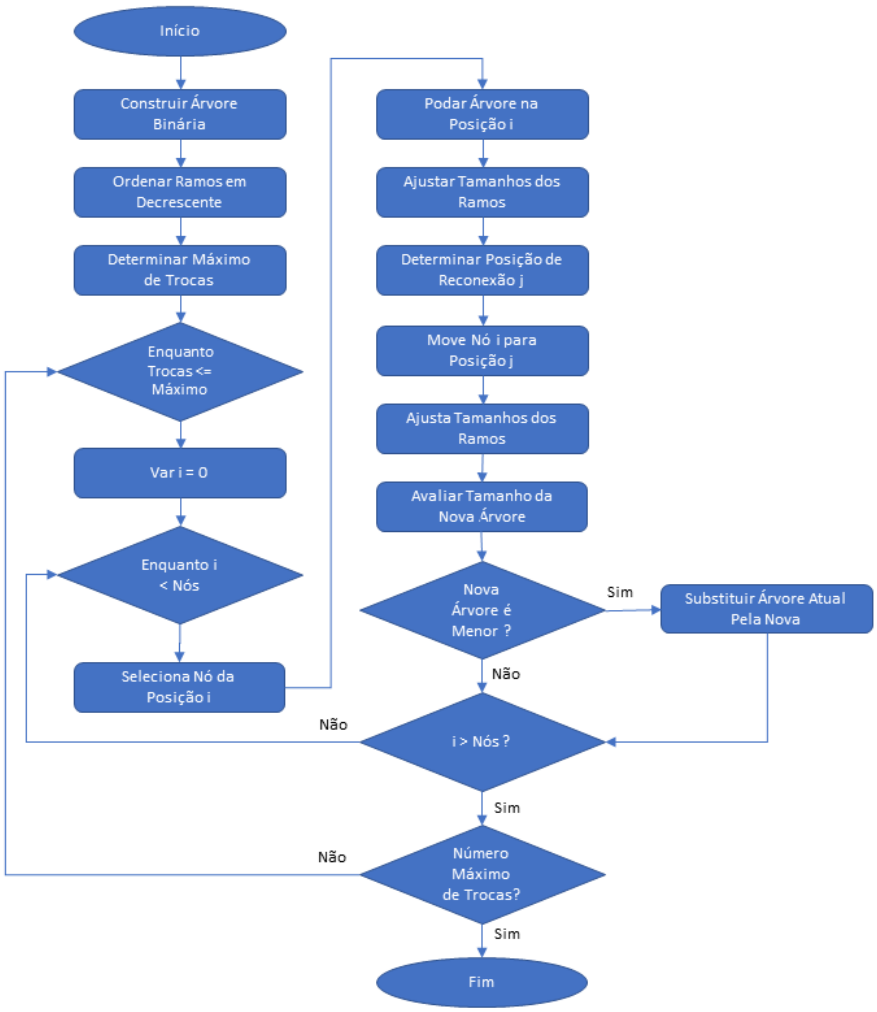
Neste ponto, o ME é empregado para selecionar primeiro os nós com maior probabilidade de trazer melhorias e, desta forma, o método deve evitar as mudanças que não produzem bons resultados, diminuindo o espaço de busca e acelerando a execução do método, reduzindo o consumo do tempo computacional.

As mudanças na topologia da árvore requerem ajustes nos tamanhos das arestas modificadas e das novas arestas criadas e, assim, é necessário o desenvolvimento de uma função para recalcular esses tamanhos. Por fim, a nova árvore produzida precisa ser avaliada e comparada com o tamanho da árvore guia inicial para determinar se houve uma melhora, ou seja, se o tamanho da nova árvore é melhor do que a árvore inicial. Caso a nova árvore seja menor que a anterior, essa é descarta e substituída pela nova árvore.

Na Figura 3.1 encontra-se representado o fluxo de execução do algoritmo 1 do método TreeRMEP, que recebe como parâmetros de entrada uma árvore gerada por uma ferramenta de AMS e a sua matriz de distâncias, retornando na saída do processo, uma nova árvore.

Inicialmente o método converte uma árvore dada como entrada para uma estrutura de árvore binária e que será utilizada pelo método. Dessa forma, torna-se o método independente da estrutura de árvore padrão usado pelas ferramentas de AMS, permitindo a sua portabilidade para outras ferramentas. Para isso, é necessário apenas a construção de um método auxiliar que interprete a estrutura padrão da ferramenta AMS e transforme na estrutura de árvore binária usada pelo TreeRMEP.

Figura 3.1: Fluxograma de execução do método.



Fonte: Elaborado pelo autor.

O uso de uma árvore binária como uma estrutura de dados capaz de armazenar todos os dados de um nó em um único local, mostrou-se uma estratégia mais segura e rápida para o desenvolvimento do trabalho. A estrutura da árvore binária implementada e adequada para o acoplamento está representada na Figura 3.2. Pode-se observar que todos os dados necessários para construção e manipulação da nova árvore guia estão reunidos em uma única estrutura e, desta forma, o processo de leitura e transferência das estruturas da ferramenta AMS para a estrutura ***btree*** do TreeRMEP mostrou-se seguro e estável.

Algorithm 1 Algoritmo de rearranjo proposto

procedure TREERMEP($tree, m\_dist$) ▷ ***tree***: árvore inicial
▷ ***m_dist***: matriz de distâncias
$btree \leftarrow new\ btree(tree,\ m\_dist)$ ▷ Construir árvore binária
$m\_ord \leftarrow new\ array()$ ▷ Matriz de ordenação
$nos \leftarrow nos(btree)$ ▷ Número de nós da árvore binária
$cont \leftarrow 0$
while $cont$ <= nos*0,8 **do** ▷ Executar em 80% dos nós,
▷ evitando mudanças ruins
$m\_ord \leftarrow ordenar(btree)$ ▷ Ordenar nós da árvore principal
$ind \leftarrow 0$
while (ind < nos) **do**
$no\_sel \leftarrow btree(m\_ord[\text{ind}])$ ▷ Selecionar o nó de posição ind
$S \leftarrow$ sub-árvore em T induzida por {u, v}
$dist \leftarrow$ determinar posição para movimento de S em T
$no\_dis \leftarrow btree(m\_ord[\text{dist}])$ ▷ Selecionar o nó de posição dist
if $no\_dist.parenth <> no\_sel.parenth$ **then** ▷ Mesmo pai, ignora
$new_btree \leftarrow$mover S para posição $dist$ em T
if $new\_btree < btree$ **then**
$btree \leftarrow new\_btree$
$ind \leftarrow ind + 1$
$cont \leftarrow cont + 1$
$tree \leftarrow$ criar nova árvore usando $btree$ **return** $tree$

Figura 3.2: A Estrutura de dados da árvore binária.

```
18  class Node{
19      public : int index;
20               int id;
21               char *name;
22               float length;
23               Node *parenth, *left, *right;
24               Node ();
25               Node (int, int, char*, float);
26               Node (Node*, int, int, char*, float);
27  };
28
29  class BTree {
30      protected : Node    *root;
31                  int     find (Node*, int);
32                  void    preOrder (Node*);
33                  void    posOrdem (Node*);
34                  int     remove (Node*,int);
35                  void    move(Node*, Node*);
36                  int     height (Node*);
37                  int     level (Node*, int, int);
38      public:     BTree();
39                  Node*   getRoot();
40                  bool    empty ();
41                  int     find (int);
42                  Node*   findID (Node*, int);
43                  void    insert (Node*, int, int, char*, float);
44                  int     remove (int);
45                  void    preOrder ();
46                  void    inOrder (Node*);
47                  void    posOrder ();
48                  int     countLeaf(Node*);
49  };
```

Fonte: Elaborada pelo autor.

Com a árvore binária construída, ordena-se as suas arestas pelos tamanhos em ordem decrescente, aplicando-se as técnicas descritas em (HSIEH et al., 2015). Iniciam-se os testes pelas arestas de maior tamanho e mais distantes da raiz, para as quais a probabilidade de uma troca gerar melhorias na árvore são maiores. Outro ponto a ser considerado e que ajuda o método executar de forma mais eficiente, é a determinação no número máximo de arestas que serão testados. Uma vez que o método ordena as arestas pelo maior tamanho e inicia as trocas pelas de maior tamanho, esse valor deve ser escolhido afim de evitar trocas de arestas que possuem os menores tamanhos e para quais as trocas não produzirão bons resultados. Nos testes executados com o BAliBase 3.0 e que serão apresentados no capítulo 4, determinou-se o número máximo em 80% (0,8) das arestas da árvore. Esse percentual pode ser ajustado de forma a balancear a qualidade do resultado e o consumo do tempo computacional, levando em consideração do tamanho dos conjuntos de sequências. Quanto maior o número de sequências no conjunto de entrada e maior o percentual escolhido, isto causa um consumo de tempo computacional maior devido aos aumento no espaço de busca.

A execução do método pode ser dividida em duas partes:

- na primeira parte, garante-se que somente o número máximo de trocas desejadas seja executado;
- na segunda parte, garante-se que para todas as trocas desejadas, todos os pontos possíveis de reconexão na árvore principal sejam testados e avaliados.

Iterativamente, no TreeRMEP seleciona-se e separa-se da árvore principal T os nós mais distantes, criando-se duas subárvores. A subárvore que foi separada da árvore T e que possui a aresta selecionada na iteração, é reconectada em outro ponto selecionado na árvore T, conservando e assegurando suas propriedades. Desta forma, T é podada gerando duas subárvores A e B e todos os possíveis pontos de reconexão são testados e avaliados para gerar uma nova árvore T'. Para cada ponto de reconexão, as distâncias

médias das arestas reconectadas são calculados para *T'* usando a equação 3.1 (DESPER; GASCUEL, 2002), em que Δ representa a matriz de distâncias evolucionárias estimadas para todos os pares possíveis de sequências do conjunto, $\Delta_{i,j}$ representa a distância entre as sequências i e j, A e B representa as duas subárvores sem interseção da árvore T e $\Delta_{A|B}$ representa a distância média entre as subárvores A e B.

$$\Delta_{A|B} = \frac{1}{|A||B|} \sum_{i \in A, j \in B} \Delta_{i,j} \tag{3.1}$$

Após a operação de troca ser realizada, o tamanho de *T'* é estimado e comparado com o tamanho de T, sendo que T pode ser a árvore inicial ou uma árvore gerada em uma iteração anterior do TreeRMEP. O tamanho de *T'* é estimando com o uso da equação 3.2 (DESPER; GASCUEL, 2002), em que *L(T')* representa o tamanho de da árvore *T'*, *L(T)* representa o tamanho a árvore T, Δ^{τ} representa a matriz de distâncias de uma dada topologia de árvore τ, A, B, C e k representam as subárvores envolvidas no processo de troca de ramos da árvore T e $\Delta^{\tau}_{A|C}$, $\Delta^{\tau}_{k|B}$, $\Delta^{\tau}_{A|B}$ e $\Delta^{\tau}_{k|C}$ representa as distâncias médias entre as subárvore, obtidas pela equação 3.1.

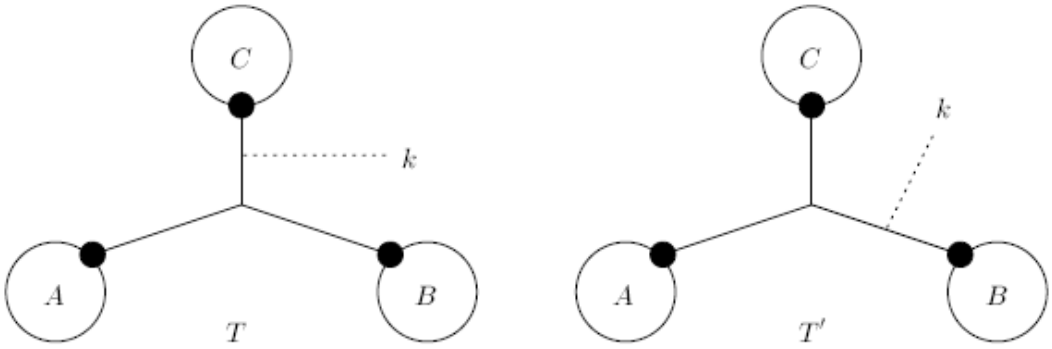
O tamanho de T é estimando com o uso da equação 3.3, em que *(u, v)* representa o ramo que liga o nó u ao nó v, *E(T)* representa o conjunto de todos os arestas da árvore T e *l(u, v)* representa o tamanho da aresta *(u, v)*. A Figura 3.3 esquematiza uma operação de troca da subárvore k em uma árvore T, que gera, após a reconexão das subárvores, uma árvore *T'*. Se o tamanho de *T'* for menor que o tamanho de T, então *T'* tem maior probabilidade de ser a árvore verdadeira, ou seja, aquela com maior fidelidade de representar a verdadeira evolução das espécies. Sendo assim, T é substituída por *T'* e a execução do algorítimo continua até que todas as trocas sejam testadas em todos os possíveis pontos de reconexão. A métrica usada para selecionar a melhor árvore, baseia no princípio da evolução mínima, descrito em (DESPER; GASCUEL, 2002;

BORDEWICH et al., 2009) e usado no método descrito em (HSIEH et al., 2015).

$$L(T') = L(T) + \frac{1}{4}\left[\left(\Delta^{\tau}_{A|C} + \Delta^{\tau}_{k|B}\right) - \left(\Delta^{\tau}_{A|B} + \Delta^{\tau}_{k|C}\right)\right] \tag{3.2}$$

$$L(T) = \sum_{(u,v)\in E(T)} l(u,v) \tag{3.3}$$

Figura 3.3: Árvore *T'* obtida pela mudança de aresta de *T*.



Fonte: (DESPER; GASCUEL, 2002).

Esse processo é repetido até o momento em que todos os nós que contribuam com melhorias na precisão da árvore e que, consequentemente melhorem o resultado do alinhamento final, sejam testados. Após realizado o refinamento da árvore por meio do método TreeRMEP, a ferramenta hospedeira recebe a árvore resultante deste processo e realiza o alinhamento das sequências do conjunto de entrada, utilizando-se de uma das técnicas que a ferramenta tem disponível na sua implementação padrão.

3.2 Desenvolvimento do Método

Para o desenvolvimento do TreeRMEP, adotou-se o uso de uma *IDE (Integrated Development Environment)* para facilitar a depuração do código da ferramenta hospedeira. No caso, no presente trabalho, a ferramenta hospedeira foi a MUSCLE (EDGAR; BATZOGLOU, 2006), na versão 3.8.31, que é a sua versão mais recente e encontra-se

disponível em www.drive5.com. A IDE selecionada foi Eclipse para C/C++ na versão Oxygen, disponível em www.eclipse.org, devido a mesma ser gratuita. Além disso, esta permite a reconstrução do projeto da MUSCLE de forma automática, sendo uma ferramenta amplamente usada e com uma comunidade altamente ativa, o que é de grande ajuda para resolver dúvidas relativas ao bom uso da IDE.

A reconstrução do projeto C++ da MUSCLE em uma IDE executando no Windows 10 foi um ponto que permitiu acelerar o desenvolvimento do trabalho. A execução da ferramenta em uma ambiente gráfico e amigável possibilitou uma análise dos detalhes da sua execução, além de se obter informações valiosas sobre a organização da ferramenta. Este fato é relevante, pois a maioria dos pesquisadores da área de ciências biológicas utilizam-se de computadores com sistema operacional Windows. Assim, uma compilação feita para Windows é mais adequada para o contexto.

Para testar a solução, utilizou-se conjuntos de sequências do BAliBase 3.0 disponível em www.ncbi.nlm.nih.gov, que é um banco de dados bem conhecido e possui conjuntos para as quais o alinhamento é conhecido. Para validar os resultados da proposta, utilizou-se a ferramenta QScore disponível em www.drive5.com/qscore, que permitiu medidas e validações confiáveis para corroborar o sucesso da proposta.

3.3 Análise da Muscle

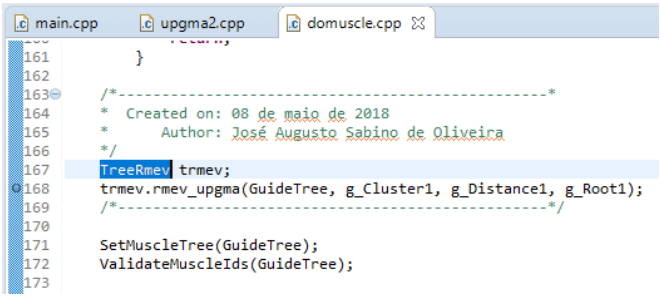
O primeiro passo para inserir a proposta na Muscle, foi acompanhar a sua execução até a localização dos pontos de interesse. Para identificar esses pontos, foi necessário analisar as diversas estruturas de dados e métodos utilizados para calcular as distâncias entra as sequências do conjunto e para a construção da árvore guia. Na construção da primeira árvore, a MUSCLE usa uma função baseada no método de contagem $k-mers$ (GUERRA; BUCKLER, 2017) para calcular as distâncias e montar a matriz de distâncias.

Nas analises da execução da MUSCLE, observou-se que a posição ideal para inse-

rir a implementação da proposta de melhoria da árvore guia é no módulo ***domuscle*** a partir da linha 163. Esta posição foi escolhida pois é nesse ponto que a primeira árvore guia gerada pela MUSCLE está pronta, ou seja, a árvore sem refinamentos.

Com o objetivo de preservar os arquivos fontes originais da MUSCLE e realizar apenas as alterações estritamente necessárias em seus códigos, optou-se por construir um novo módulo separado no qual implementou-se toda a lógica do método proposto. Isso é importante pelo fato de criar maior independência para o TreeRMEP e facilitar a sua incorporação em outras ferramentas de AMS. Desta forma, como observa-se na Figura 3.4, somente duas linhas de códigos foram acrescentadas ao código original da MUSCLE no módulo ***domuscle***.

Figura 3.4: Ponto de alteração da Muscle.

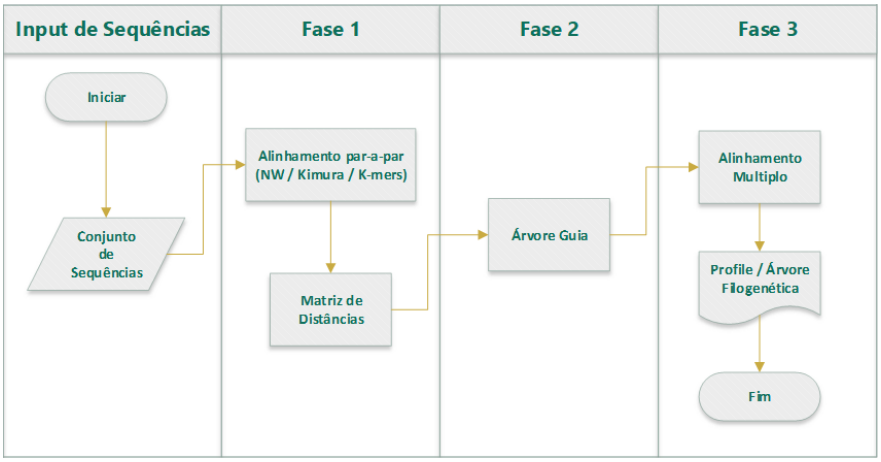


Fonte: Elaborada pelo autor.

A MUSCLE utiliza-se da estratégia do Alinhamento Progressivo para construir o resultado e sua sua execução ocorre basicamente em três fases, que estão representadas na Figura 3.5.

O Alinhamento Progressivo recebe um conjunto de sequências como entrada e como mencionado anteriormente, utilizou-se dos conjuntos de sequências do BaliBase 3.0, sendo que um dos conjuntos usados como entrada está representado na Figura 3.6.

Figura 3.5: Fases de Execução do Alinhamento Progressivo.



Fonte: Elaborada pelo autor.

Figura 3.6: Conjunto de sequências do BAliBase.

>1aboA	NLFVALYDFVASGDNTLSITKGEKLRVLGYNHNGEWCEAQTKNGQGWVPSNYITPVN
>1ark	TAGKIFRAMYDYMAADADEVSFKDGDAIINVQAIDEGWMYGTVQRTGRTGMLPANYVEAI
>1gbq	MEAIAKYDFKATADDELSFKRGDILKVLNEECDQNWYKAELNGKDGFIPKNYIEMKP
>1ckb	AEYVRALFDFNGNDEEDLPFKKGDILRIRDKPEEQWWNAEDSEGKRGMIPVPYVEKY
>1gfc	GSTYVQALFDFDPQEDGELGFRRGDFIHVMDNSDPNWWKGACHGQTGMFPRNYVTPV
>1hsp	GSPTFKCAVKALFDYKAQREDELTFIKSAIIQNVEKQEGGWWRGDYGGKKQLWFPSNYVEEMV
>1aey	GKELVLALYDYQEKSPREVTMKKGDILTLLNSTNKDWWKVEVNDRQGFVPAAYVKKL
>1csk	GTECIAKYNFHGTAEQDLPFCKGDVLTIVAVTKDPNWYKAKNKVGREGIIPANYVQKR
>1ad5	EDIIVVALYDYEAIHHEDLSFQKGDQMVVLEESGEWWKARSLATRKEGYIPSNYVARVD
>1awj	RRSFQEPEETLVIALYDYQTNDPQELALRCDEEYYLLDSSEIHWWRVQDKNGHEGYAPSSYLVEKS
>1efn	ALFVALYDYEAITEDDLSFHKGEKFQILNSSEGDWWEARSLTTGETGYIPSNYVAPV
>1sem	ETKFVQALFDFNPQESGELAFKRGDVITLINKDDPNWWEGQLNNRRGIFPSNYVCPY
>1ycsB	KGVIYALWDYEPQNDDELPMKEGDCMTIIHREDEDEIEWWWARLNDKEGYVPRNLLGLYP
>1pht	GYQYRALYDYKKEREEDIDLHLGDILTVNKGSLVALGFSDGQEARPEEIGWLNGYNETTGERGDFPGTYVEYIGRKKISP
>1ihvA	NFRVYYRDSRDPVWKGPAKLLWKGEGAVVIQDNSDIKVVPRRKAKIIRD

Fonte: Elaborada pelo autor.

Na primeira fase de execução, a MUSCLE realiza o alinhamento par-a-par de todos os pares possíveis de sequências e constrói uma árvore guia, utilizando-se de diversas matrizes para armazenar os dados que compõe a estrutura da árvore guia, como verifica-se na representação da Figura 3.7. A função de cada matriz está descrita na Tabela 3.1 e como qualquer árvore guia, as sequências do conjunto que a mesma representa estão todas armazenadas em seus nós externos, que são os nós folhas.

Tabela 3.1: Estrutura da árvore guia da Muscle.

Parâmetro	**Descrição**
Index	Índice da matriz.
	continua na próxima página

Tabela 3.1 – continua na próxima página

Parâmetro	Descrição
m_uNeighbor1	Nós da árvore em ordem decrescente pelo tamanho das arestas. A última posição representa a raiz.
m_uNeighbor2	Nó filho esquerdo do nó de mesmo índice da estrutura ***m_uNeighbor1***. O valor ***4294967295*** indica posição vazia.
m_uNeighbor3	Nó filho direito do nó de mesmo índice da estrutura ***m_uNeighbor1***. O valor ***4294967295*** indica posição vazia.
m_bHasEdgeLength1	O valor ***"true"*** indica que a matriz ***m_dEdgeLength1*** armazena um tamanho válido para aresta.
m_bHasEdgeLength2	O valor ***"true"*** indica que a matriz ***m_dEdgeLength2*** armazena um tamanho válido para aresta.
m_bHasEdgeLength3	O valor ***"true"*** indica que a matriz ***m_dEdgeLength3*** armazena um tamanho válido para aresta.
m_dEdgeLength1	Tamanho das arestas e o valor ***900000000000*** indica uma posição vazia ou raiz da árvore. Com o uso dessa matriz e da matriz ***m_uNeighbor1*** é possível construir a representação da árvore da Figura 3.8.
m_dEdgeLength2 [left]	Tamanho da aresta do nó filho esquerdo do nó de mesmo índice representado na estrutura ***m_uNeighbor1***. Parte do tamanho da aresta dos nós folhas é cedida para as arestas que se ligam aos nós internos e que não possuem tamanho por não representarem uma sequência. O valor ***900000000000*** indica uma posição vazia.
m_dEdgeLength3 [right]	Tamanho da aresta do nó filho direito do nó de mesmo índice representado na estrutura ***m_uNeighbor1***. Parte do tamanho da aresta dos nós folhas é cedida para as arestas que se ligam aos nós internos e que não possuem tamanho por não representarem uma sequência. O valor ***900000000000*** indica uma posição vazia.
m_Ids	Identificadores das sequências (nós folhas). O valor ***8888888*** índica um nó interno.
m_ptrName	Nomes das sequências (nós folhas).

A matriz de distâncias é dada como entrada na segunda fase, na qual a árvore representada na Figura 3.8 é produzida. A árvore, trata-se de um Cladograma (SLATKIN; MADDISON, 1989) que demonstra as relações filogenéticas ou genealógicas entre as sequências analisadas. Com estas relações demonstra-se uma história comum entre as mesmas e apesar de serem obtidas por caracteres morfológicos, as sequências de DNA e RNA e a filogenética computacional são normalmente usados para gerar tais

Figura 3.7: Estrutura da árvore guia da Muscle.

Index	m_uNeighbor1	m_uNeighbor2	m_uNeighbor3	m_Ids	m_ptrName	m_bHasEdge Length1	m_bHasEdge Length2	m_bHasEdge Length3	m_dEdgeLength1	m_dEdgeLength2 [left]	m_dEdgeLength3 [right]
0	18	4294967295	4294967295	0	1aboA	true	false	false	1,240384579	900000000000	900000000000
1	25	4294967295	4294967295	1	1ark	true	false	false	1,363636374	900000000000	900000000000
2	15	4294967295	4294967295	2	1gbq	true	false	false	1,182692289	900000000000	900000000000
3	16	4294967295	4294967295	3	1ckb	true	false	false	1,182692289	900000000000	900000000000
4	17	4294967295	4294967295	4	1gfc	true	false	false	1,187019229	900000000000	900000000000
5	26	4294967295	4294967295	5	1hsp	true	false	false	1,374427795	900000000000	900000000000
6	24	4294967295	4294967295	6	1aey	true	false	false	1,327961206	900000000000	900000000000
7	15	4294967295	4294967295	7	1csk	true	false	false	1,182692289	900000000000	900000000000
8	19	4294967295	4294967295	8	1ad5	true	false	false	1,244711518	900000000000	900000000000
9	22	4294967295	4294967295	9	1awj	true	false	false	1,316944957	900000000000	900000000000
10	18	4294967295	4294967295	10	1efn	true	false	false	1,240384579	900000000000	900000000000
11	16	4294967295	4294967295	11	1sem	true	false	false	1,182692289	900000000000	900000000000
12	20	4294967295	4294967295	12	1ycsB	true	false	false	1,253004789	900000000000	900000000000
13	25	4294967295	4294967295	13	1pht	true	false	false	1,363636374	900000000000	900000000000
14	28	4294967295	4294967295	14	1ihvA	true	false	false	1,450924516	900000000000	900000000000
15	21	2	7	8888888		true	true	true	0,087543368	1,182692289	1,182692289
16	17	3	11	8888888		true	true	true	0,004326940	1,182692289	1,182692289
17	20	4	16	8888888		true	true	true	0,065985560	1,187019229	0,00432694
18	19	0	10	8888888		true	true	true	0,004326940	1,240384579	1,240384579
19	23	8	18	8888888		true	true	true	0,076218009	1,244711518	0,00432694
20	21	12	17	8888888		true	true	true	0,017230868	1,253004789	0,06598556
21	22	15	20	8888888		true	true	true	0,046709299	0,087543368	0,017230868
22	23	9	21	8888888		true	true	true	0,003984571	1,316944957	0,046709299
23	24	19	22	8888888		true	true	true	0,007031679	0,076218009	0,003984571
24	27	6	23	8888888		true	true	true	0,073021293	1,327961206	0,007031679
25	26	1	13	8888888		true	true	true	0,010791421	1,363636374	1,363636374
26	27	5	25	8888888		true	true	true	0,026554704	1,374427795	0,010791421
27	28	24	26	8888888		true	true	true	0,049942017	0,073021293	0,026554704
28		14	27	8888888		false	true	true	900000000000	1,450924516	0,049942017

Fonte: Elaborada pelo autor.

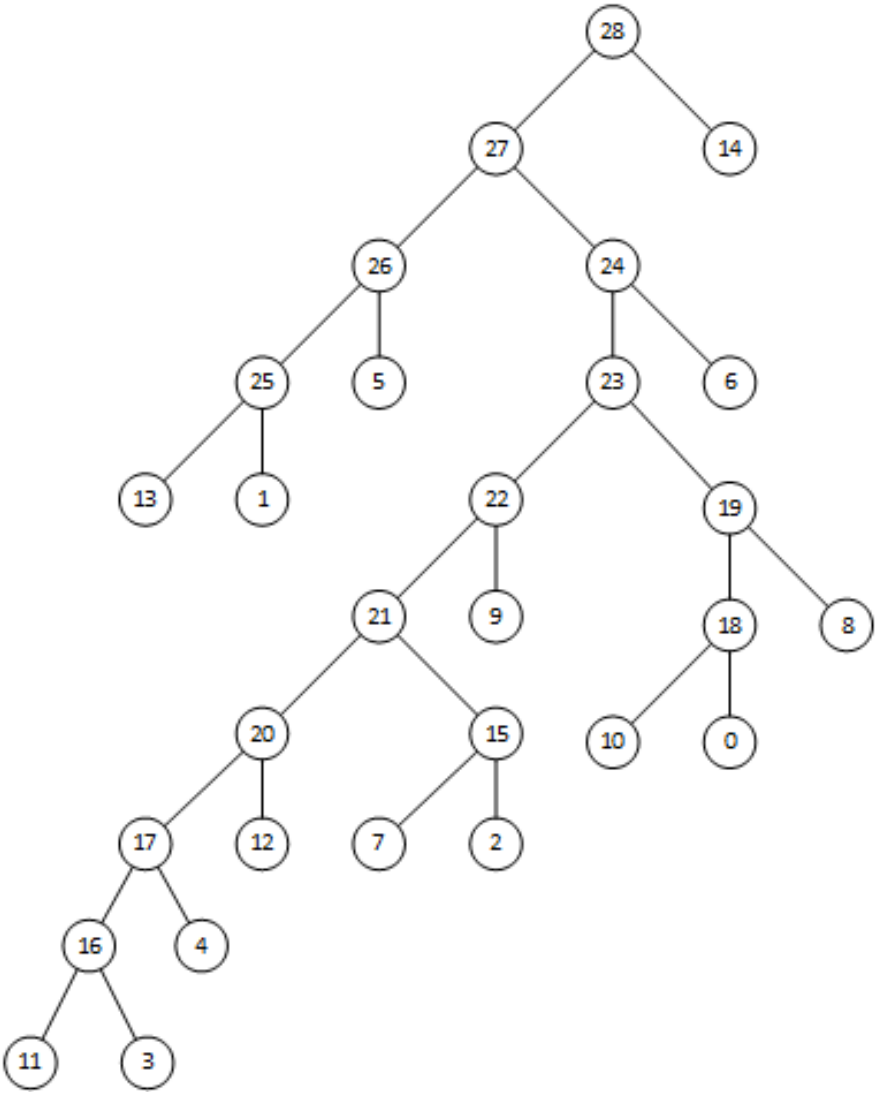
cladogramas. Desta forma, por representar as relações de evolução das espécies geram árvores que nem sempre são balanceadas.

A árvore guia é submetida a fase 3 da MUSCLE para finalmente, produzir como resultado, o conjunto de sequências alinhadas demostrado na Figura 3.9.

3.4 Considerações

Nesse capítulo apresentou-se o trabalho realizado para o desenvolvimento do método de rearranjo de árvores TreeRMEP, aplicado as ferramentas de AMS na etapa pós construção da árvore guia. O método foi desenvolvido com base no ME, em que se aplicam técnicas que visam reduzir o espaço de busca e evitar testes desnecessários e, desta forma, otimizar o tempo computacional. Assim, caso o tamanho de uma árvore seja minimizado, a TreeRMEP irá selecionar e devolver para a ferramenta de AMS a nova árvore. Caso contrário, a árvore original será mantida, de modo a não degradar o resultado do alinhamento de sequências produzido.

Figura 3.8: Árvore guia da Muscle.



Fonte: Elaborada pelo autor.

Figura 3.9: Conjunto de sequências alinhadas.

>1ihvA	------NFRVYYRDSRDPVWKGPAKL-LWKGEGAVVI----------QDNSDIK---------------VVPRRKAKIIRD-----
>1pht	--------GYQYRALYDYKKEREEDIDLHLGDILTVNKGSLVALGFSDGQEARPEEIGWLNGYNETTGERGDFPGTYVEYIGRKKISP
>1awj	RRSFQEPEETLVIALYDYQTNDPQELALRCDEEYYLL----------D-SSEIH---WWRVQ-DKNGHEGYAPSSYLVEKS------
>1ark	------TAGKIFRAMYDYMAADADEVSFKDGDAIINV----------Q-AIDEG---WMYGTVQRTGRTGMLPANYVEAI-------
>1aey	-------GKELVLALYDYQEKSPREVTMKKGDILTLL----------N-STNKD---WWKVE--VNDRQGFVPAAYVKKL-------
>1aboA	---------NLFVALYDFVASGDNTLSITKGEKLRVL----------GYNHNGE---WCEAQ--TKNGQGWVPSNYITPVN------
>1ad5	-------EDIIVVALYDYEAIHHEDLSFQKGDQMVVL-------------EESGE---WWKARSLATRKEGYIPSNYVARVD------
>1efn	---------ALFVALYDYEAITEDDLSFHKGEKFQIL----------N-SSEGD---WWEARSLTTGETGYIPSNYVAPV-------
>1hsp	---GSPTFKCAVKALFDYKAQREDELTFIKSAIIQNV-----------EKQEGG---WWRGD-YGGKKQLWFPSNYVEEMV------
>1ckb	--------AEYVRALFDFNGNDEEDLPFKKGDILRI-----------RDKPEEQ---WWNAE-DSEGKRGMIPVPYVEKY-------
>1gbq	---------MEAIAKYDFKATADDELSFKRGDILKVL----------NEECDQN---WYKAE--LNGKDGFIPKNYIEMKP------
>1csk	--------GTECIAKYNFHGTAEQDLPFCKGDVLTIV----------AVTKDPN---WYKAK-NKVGREGIIPANYVQKR-------
>1ycsB	--------KGVIYALWDYEPQNDDELPMKEGDCMTII----------H-REDEDEIEWWWAR--LNDKEGYVPRNLLGLYP------
>1gfc	-------GSTYVQALFDFDPQEDGELGFRRGDFIHVM----------D-NSDPN---WWKGA--CHGQTGMFPRNYVTPV-------
>1sem	-------ETKFVQALFDFNPQESGELAFKRGDVITLI----------N-KDDPN---WWEGQ--LNNRRGIFPSNYVCPY-------

Fonte: Elaborada pelo autor.

Capítulo 4

Testes e Resultados

Neste capítulo são apresentados os testes e resultados obtidos a partir da implementação da abordagem proposta no presente trabalho para melhoria no resultado das ferramentas de AMS. Além disso, especifica-se a plataforma de testes e o Benchmark utilizado para aferir a qualidade dos resultados obtidos.

4.1 Execução dos Testes

Todos os casos de teste do presente trabalho foram executados no Laboratório de Bioinformática da UNESP, Campus de São José do Rio Preto e executados em um computador Dell Inspiron 5558, com processador Intel Core i5-5200U, com ciclo de 2,20GHz, de arquitetura 64 bits, com 8GB de memória RAM, executando sistema operacional Windows 10 64 Bits.

Na execução da MUSCLE utilizou-se os parâmetros descritos na Tabela 4.1. No entanto a ferramenta oferece uma gama maior de opções de parâmetros, como pode ser visto em (EDGAR et al., 2005). Como exemplo, pode-se citar o parâmetro ***distance1***, que determina qual medida de distância será usada na primeira interação da MUSCLE. Quando omitido seu valor, são assumidos os valores ***Kmer6-6*** para aminoácidos e ***Kmer6-4*** para nucleotídeos.

Tabela 4.1: Parâmetros da MUSCLE usados.

Parâmetro	**Valor**	**Descrição**
-in	nome do arquivo	Arquivo de entrada, contendo o conjunto de sequências a ser alinhadas.
-out	nome do arquivo	Arquivo de saída, contendo o conjunto de sequências alinhadas.
-msfout	nome do arquivo	Arquivo de saída no formato MSF.
-tree1	nome do arquivo	Argumento opcional. Arquivo de saída contendo o esquema da primeira árvore gerada.
-tree2	nome do arquivo	Argumento opcional. Arquivo de saída contendo o esquema da segunda árvore gerada, resultante do refinamento da primeira árvore.
-log	nome do arquivo	Argumento opcional. Arquivo de saída contendo os logs gerados nas etapas do processo de alinhamento e construção das árvores.
-cluster1	upgma upgmb neighborjoining	Método de agrupamento usado na primeira iteração.
-cluster2	upgma upgmb neighborjoining	Método de agrupamento usado na segunda iteração.

Sendo assim, salvo os parâmetros descrito na Tabela 4.1, os testes do trabalho proposto baseiam-se na execução da MUSCLE utilizando os valores padrão para os demais parâmetros. Os parâmetros *-cluster1* e *-cluster2* determinam quais funções de agrupamento serão utilizadas pela MUSCLE nos dois estágios de execução, o estágio de construção da árvore guia e o estágio de refinamento do alinhamento produzido. Para os casos de testes executados, utilizou-se o valor *neighborjoining*.

Para avaliar os resultados propostos pelo TreeRMEP, foram executados testes para todos os conjuntos de sequências disponíveis no BAliBase 3.0 e os alinhamentos resultantes foram avaliados com a ferramenta QScore (MOORE; YOUNG; LEE, 2002), pontuando as medidas Q Score (Q) e Total Column Score (TC) . Devido à abordagem estocástica da AMS, que utiliza métodos probabilísticos para encontrar um resultado ótimo, diferente da PD que é uma abordagem determinística, cada caso de teste foi executado cinco vezes e a pontuação considerada foi a média aritmética das execu-

ções, de modo a garantir uma avaliação mais robusta em termos estatísticos.

As pontuações Q e TC representam métricas de significância biológica em AMS. A pontuação *TC* é uma métrica de similaridade para alinhamentos, que relacionada ao número de colunas compartilhadas pelos dois perfis alinhados(THOMPSON et al., 2005). A *Q* é uma métrica usada para avaliar a qualidade do alinhamento quando comparado a um alinhamento de referência, que normalmente são alinhados manualmente e que, portanto, representam o resultado exato. Assim, foi possível comparar os resultados obtidos pela abordagem proposta com uma das ferramentas do estado da arte. Além disso, foram coletados os tempos de execução da ferramenta com o uso e sem o uso da abordagem de rearranjo proposta, afim de avaliar também o acréscimo do tempo de execução exigido pela abordagem proposta.

4.2 Resultados

Os resultados obtidos pelo método TreeRMEP foram comparados com os resultados da MUSCLE original e de outras ferramentas bem conhecidas de AMS. A comparação dos resultados esta demonstrada na Tabela 4.2, na qual observa-se as métricas adotadas para medir a eficiência com os conjuntos do BAliBase 3.0, que é composto por 386 conjuntos, totalizando 11.082 sequências com tamanho médio de 702 símbolos, sendo a menor sequência com 52 símbolos e a maior com 8.495, separadas em seis famílias de sequencias, sendo elas: RV11, RV12, RV20, RV30, RV40 e RV50. Pode-se observar que o método obteve bons resultados, mas as diferenças observadas entre a TreeRMEP e a MUSCLE são pequenas e isso se deve ao fato da MUSCLE já ser uma ferramenta que produz bons resultados.

Tabela 4.2: Comparação dos resultados obtidos.

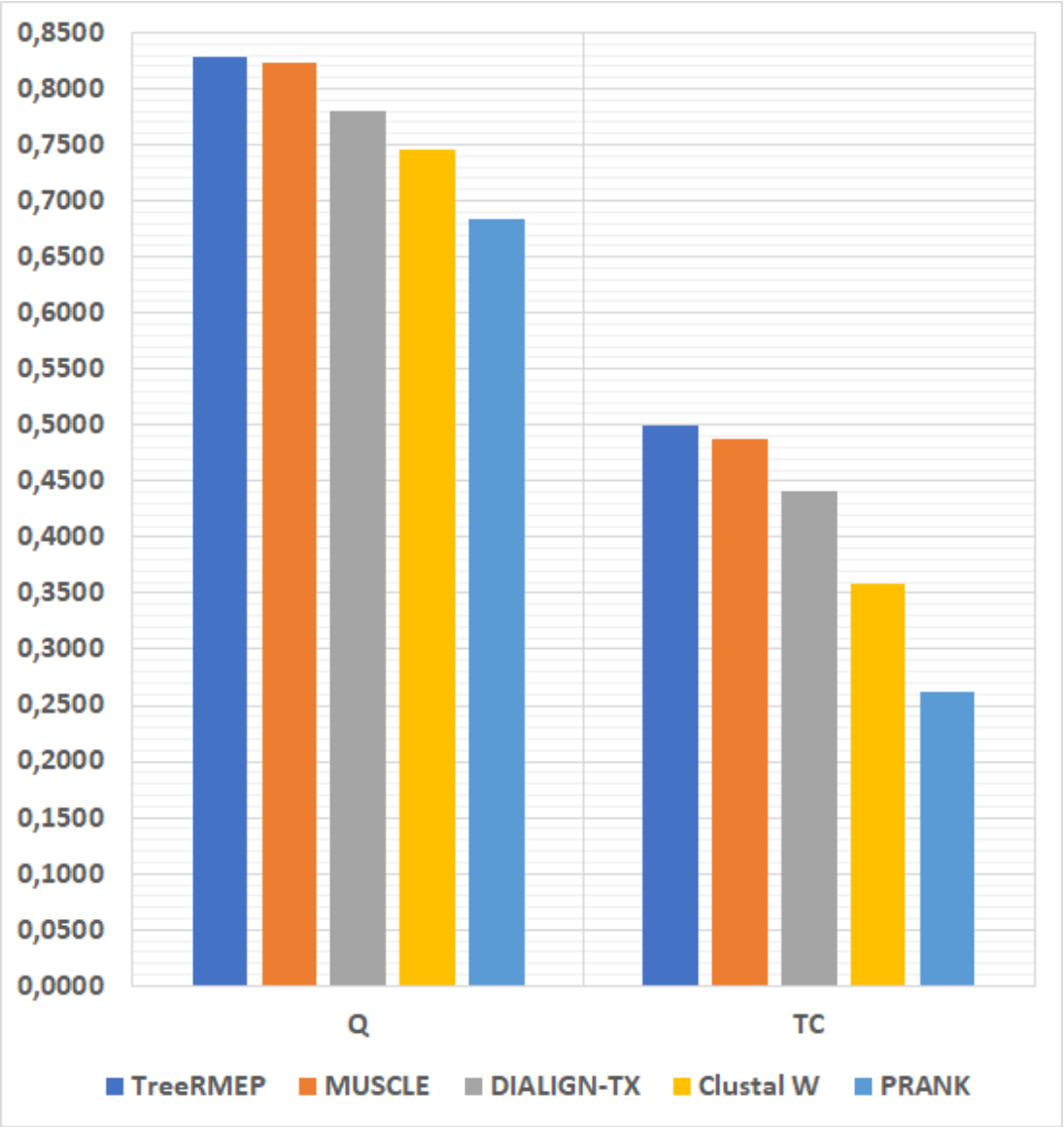
Ferramentas	Total		RV11		RV12		RV20		RV30		RV40		RV50	
	Q	TC	Q	TC	Q	TC	Q	TC	Q	TC	Q	TC	Q	TC
TreeRMEP	0,8281	0,4997	0,6467	0,4064	0,9124	0,7947	0,9058	0,3897	0,8190	0,4490	*0,8338*	0,3955	0,8368	*0,4444*
MUSCLE	0,8238	0,4879	0,6408	0,4050	0,9108	0,7875	0,9045	0,3745	0,8108	0,4243	0,8247	0,3823	0,8355	0,4307
DIALIGN-TX	0,7803	0,4407	0,5047	0,2681	0,8821	0,7569	0,8781	0,3078	0,7614	0,3890	0,8340	0,4517	0,8215	0,4705
Clustal W	0,7457	0,3590	0,5006	0,2299	0,8649	0,7170	0,8520	0,2216	0,7250	0,2759	0,7894	*0,3982*	0,7424	0,3116
PRANK	0,6834	0,2629	0,4618	0,2162	0,8377	0,6208	0,8014	0,1242	0,5784	0,0638	0,7477	0,3422	0,6735	0,2099

Na Tabela 4.2, para melhor observação dos resultados, as células com os melhores resultados estão com a cor de fundo em destaque, onde observa-se que no total geral, o TreeRMEP obteve resultados melhores que todas as ferramentas comparadas. Quando realizada uma comparação com a MUSCLE original, que é uma das ferramentas de AMS mais utilizadas em Bioinformática, para todas as famílias do BAliBase, o TreeRMEP foi capaz de promover melhorias nos resultados. Comparando-se os resultados das famílias com as demais ferramentas, pode-se observar que para a pontuação Q, o TreeRMEP só não obteve o melhor resultado para a família RV40, no entanto obteve o segundo lugar. Para a pontuação *TC*, nas famílias RV40 e RV50 o TreeRMEP não obteve os melhores resultados, porém para a família RV50 obteve o segundo lugar. Nestes últimos resultados discutidos, o TreeRMEP só não obteve os melhores resultados devido às características das sequências dessas famílias, as quais apresentam um menor índice de similaridade entre elas. A ferramenta que ficou em primeiro lugar, que é a DIALIGN-TX, possui alguns ajustes em seu método para tratar sequências com similaridade menor. Porém, isto impacta na qualidade, que é reduzida, dos seus resultados com sequências mais similares, bem como no seu tempo de execução, que é mais elevado, pois necessita de mais iterações para extrair informações dos conjuntos menos similares. Na Figura 4.1, demonstra-se de forma gráfica os resultados obtidos e que implicam diretamente em melhores alinhamentos.

Com relação ao consumo de tempo computacional, o objetivo da proposta é não gerar uma degradação que torne sua execução impraticável. Na Tabela 4.3 observa-se o tempo para executar todas as famílias do BAliBase. Nota-se que houve um sútil aumento no tempo, mas o acréscimo ficou em níveis aceitáveis, totalizando 1 minuto e 6 segundos de acréscimo, o que totaliza 0,81% de acréscimo quando o TreeRMEP é comparado com a MUSCLE original. Desta forma, o TreeRMEP cumpriu o objetivo proposto de não inviabilizar o seu uso, como pode-se observar na Figura 4.2. O aumento no tempo de execução pode ser ajustado através da determinação no número

máximo de arestas que serão testados, como discutido na seção 3.1 e nos nossos testes adotou-se um fator de 80%.

Figura 4.1: Comparação dos resultados obtidos.

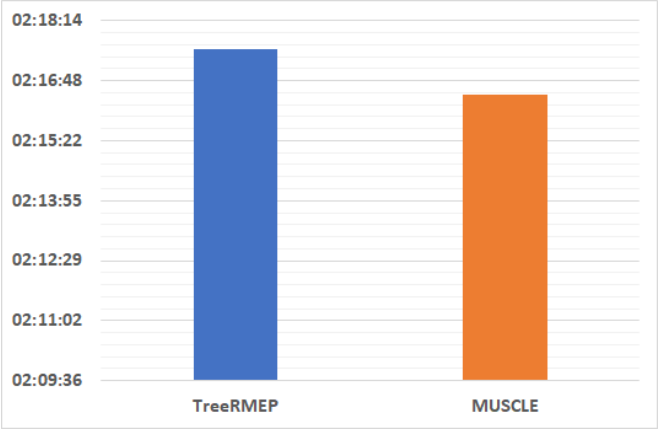


Fonte: Elaborada pelo autor.

Tabela 4.3: Tempos de execução.

	Tempo de Execução
TreeRMEP	02:17:34
MUSCLE	02:16:28
Minutos de Acréscimo	**00:01:06**
Percentual	0,81%

Figura 4.2: Tempos de execução.



Fonte: Elaborada pelo autor.

Na Tabela 4.4, observa-se a efetividade com às métricas adotadas para analisar os resultados obtidos com o BAliBase 3.0, que é composto por 386 conjuntos de sequências distribuídos em 6 famílias. Para ambas as métricas, a efetividade foi alcançada, o que significa que para a maioria dos conjuntos, o objetivo proposto foi alcançado. Para a métrica *Q*, em 269 ou 70% dos conjuntos e para a métrica *TC*, em 307 ou 80% dos conjuntos, o método melhorou a acurácia do alinhamento produzido.

Tabela 4.4: Efetividade do TreeRMEP.

Métricas	**Conjuntos Totais**	**Conjuntos com Ganho**	**Conjuntos com Perda**	**Efetividade**
Q	386	269	117	70%
TC	386	307	79	80%

4.3 Considerações

A partir da análise dos resultados, observa-se a contribuição da estratégia implementada no trabalho. Por meio do modelo adotado para avaliar os resultados das mudanças na topologia da árvore, encontrou-se árvores melhores e sem grandes acréscimos no tempo de execução e possibilitou-se a obtenção de melhores pontuações de Q e TC. Consequentemente, alinhamentos de sequências mais precisos são obtidos. Com isso, as ferramentas que utilizarem o método tornam-se mais robusta e produzirão resultados com mais significância biológica.

Capítulo 5

Conclusões

5.1 Considerações Finais

No presente trabalho, para o bom entendimento do contexto no qual a Bioinformática está inserida, apresentamos vários conceitos fundamentais e uma breve descrição das principais ferramentas e heurísticas para o Alinhamento Múltiplo de Sequências, com maior atenção para a heurística do Alinhamento Progressivo. Também apresentamos os conceitos básicos para o entendimento do contexto do problema da reconstrução filogenética e rearranjo de árvores. Destacou-se várias técnicas aplicadas à solução do referido problema, com objetivo de obter-se melhores resultados no Alinhamento Múltiplo de Sequência. Uma atenção especial foi dada à técnica TBR, a fim de apresentar um resultado final do alinhamento com maior significância biológica.

Sendo assim, buscou-se apresentar os bons resultados obtidos pelo MUSCLE, uma das principais e mais utilizadas ferramentas na área de Bioinformática e o funcionamento da TBR, bem como os bons resultados obtidos da sua utilização. Além disso, destacou-se os pontos em que se aplicou a TBR na MUSCLE, a fim de melhorar os resultados dos alinhamentos de sequencias produzidos.

A escolha da TBR deu-se pelos excelentes resultados apresentados em sua aplicação por Hsieh et al. (HSIEH et al., 2015). A escolha do MUSCLE deu-se pela ampla

utilização da ferramenta em Bioinformática, por sua abordagem computacional bem estruturada e sua capacidade de produzir bons resultados com custo computacional baixo (EDGAR; BATZOGLOU, 2006). Além do que, os códigos-fonte da MUSCLE são abertos e encontram-se muito bem fundamentados no paradigma de programação Orientado a Objetos, tornando o desenvolvimento da proposta mais robusto.

A aplicação do TreeRMEP para reconstrução filogenética produzida pela função padrão do MUSCLE, logo após a geração da primeira árvore guia, sem refinamentos, foi capaz de obter uma árvore guia melhor, com distâncias menores entre suas ramificações, de acordo com o Principio da Evolução Miníma. Isto resultou em uma árvore filogenética mais próxima da árvore verdadeira, resultando em contribuição para produzir alinhamentos de sequências com maior significância biológica.

Os resultados obtidos comprovaram a contribuição do método, com pequeno aumento no tempo computacional, o que era esperado, mas ficou dentro de níveis que não inviabilizam o método, o que é muito importante, uma vez que existe a necessidade de evolução na capacidade da MUSCLE de operar com grandes conjuntos de dados, especialmente os produzidos por NGS, o que é um fator de limitação da ferramenta.

5.2 Trabalhos Futuros

Apesar do tempo de execução ter ficado em limites aceitáveis, existe a possibilidade do desenvolvimento de uma versão do método para ambientes paralelizados, como *grids* ou *clusters*. Atualmente muitas aplicações tem usufruído do poder de processamento em GPU *(Graphics Processing Unit)* e uma versão do TreeRMEP paralelizado será capaz de compensar o acréscimo no tempo de execução.

Outro ponto é que para algumas sequências o método não foi capaz de encontrar uma árvore melhorada, o que abre a possibilidade para se investigar outras funções de medidas capazes de avaliar com maior precisão as topologias geradas e, como isso, aumentar a qualidade dos resultados obtidos pelas ferramentas de AMS.

REFERÊNCIAS

ADAMS, R. L. P. et al. *The biochemistry of the nucleic acids*. [S.l.]: Chapman and Hall, 1992.

ALTSCHUL, S. F.; CARROLL, R. J.; LIPMAN, D. J. Weights for data related by a tree. *Journal of molecular biology*, Elsevier, v. 207, n. 4, p. 647–653, 1989.

AMARAN, S. et al. Simulation optimization: a review of algorithms and applications. *Annals of Operations Research*, Springer, v. 240, n. 1, p. 351–380, 2016.

BAJORATH, J.; STENKAMP, R.; ARUFFO, A. Knowledge-based model building of proteins: concepts and examples. *Protein science: a publication of the Protein Society*, Blackwell Publishing, v. 2, n. 11, p. 1798, 1993.

BASTKOWSKI, S. et al. The minimum evolution problem is hard: a link between tree inference and graph clustering problems. *Bioinformatics*, Oxford University Press, v. 32, n. 4, p. 518–522, 2015.

BEHJATI, S.; TARPEY, P. S. What is next generation sequencing? *Archives of Disease in Childhood-Education and Practice*, Royal College of Paediatrics and Child Health, v. 98, n. 6, p. 236–238, 2013.

BLACKSHIELDS, G. et al. Sequence embedding for fast construction of guide trees for multiple sequence alignment. *Algorithms for Molecular Biology*, BioMed Central, v. 5, n. 1, p. 21, 2010.

BORDEWICH, M. et al. Consistency of topological moves based on the balanced minimum evolution principle of phylogenetic inference. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, IEEE Computer Society Press, v. 6, n. 1, p. 110–117, 2009.

BORDEWICH, M.; SEMPLE, C. On the computational complexity of the rooted subtree prune and regraft distance. *Annals of combinatorics*, Springer, v. 8, n. 4, p. 409–423, 2005.

BOYCE, K.; SIEVERS, F.; HIGGINS, D. G. Instability in progressive multiple sequence alignment algorithms. *Algorithms for molecular biology*, BioMed Central, v. 10, n. 1, p. 26, 2015.

BRAZMA, A. et al. Approaches to the automatic discovery of patterns in biosequences. *Journal of computational biology*, v. 5, n. 2, p. 279–305, 1998.

BRYANT, D. The splits in the neighborhood of a tree. *Annals of Combinatorics*, Springer, v. 8, n. 1, p. 1–11, 2004.

BUSIA, A. et al. A deep learning approach to pattern recognition for short dna sequences. *bioRxiv*, Cold Spring Harbor Laboratory, p. 353474, 2018.

CAPELLA-GUTIÉRREZ, S.; GABALDÓN, T. Measuring guide-tree dependency of inferred gaps in progressive aligners. *Bioinformatics*, Oxford University Press, v. 29, n. 8, p. 1011–1017, 2013.

CHAPIN, E. A.; CHAPIN, C. C. Darwin and the galapagos islands. *Bull. Pan Am. Union*, HeinOnline, v. 69, p. 655, 1935.

CHATZOU, M. et al. Multiple sequence alignment modeling: methods and applications. *Briefings in bioinformatics*, Oxford University Press, v. 17, n. 6, p. 1009–1023, 2015.

COHEN, J. Bioinformatics—an introduction for computer scientists. *ACM Computing Surveys (CSUR)*, ACM, v. 36, n. 2, p. 122–158, 2004.

DAHM, R. Discovering dna: Friedrich miescher and the early years of nucleic acid research. *Human genetics*, Springer, v. 122, n. 6, p. 565–581, 2008.

DAY, W. H. Properties of the nearest neighbor interchange metric for trees of small size. *Journal of Theoretical Biology*, Elsevier, v. 101, n. 2, p. 275–288, 1983.

DENIS, F.; GASCUEL, O. On the consistency of the minimum evolution principle of phylogenetic inference. *Discrete Applied Mathematics*, Elsevier, v. 127, n. 1, p. 63–77, 2003.

DESPER, R.; GASCUEL, O. Fast and accurate phylogeny reconstruction algorithms based on the minimum-evolution principle. *Journal of computational biology*, Mary Ann Liebert, Inc., v. 9, n. 5, p. 687–705, 2002.

DORIGO, M.; BLUM, C. Ant colony optimization theory: A survey. *Theoretical computer science*, Elsevier, v. 344, n. 2, p. 243–278, 2005.

EDGAR, R. C. Muscle: a multiple sequence alignment method with reduced time and space complexity. *BMC bioinformatics*, BioMed Central, v. 5, n. 1, p. 113, 2004.

EDGAR, R. C. Muscle: multiple sequence alignment with high accuracy and high throughput. *Nucleic acids research*, Oxford University Press, v. 32, n. 5, p. 1792–1797, 2004.

EDGAR, R. C.; BATZOGLOU, S. Multiple sequence alignment. *Current opinion in structural biology*, Elsevier, v. 16, n. 3, p. 368–373, 2006.

EDGAR, R. C. et al. *MUSCLE user guide*. [S.l.], 2005.

EIDHAMMER, I.; JONASSEN, I.; TAYLOR, W. R. Structure comparison and structure patterns. *Journal of Computational Biology*, Mary Ann Liebert, Inc., v. 7, n. 5, p. 685–716, 2000.

FENG, D.-F.; DOOLITTLE, R. F. Progressive sequence alignment as a prerequisitetto correct phylogenetic trees. *Journal of molecular evolution*, Springer, v. 25, n. 4, p. 351–360, 1987.

FILIPSKI, A. et al. Phylogenetic placement of metagenomic reads using the minimum evolution principle. *BMC genomics*, BioMed Central, v. 16, n. 1, p. S13, 2015.

GAO, X.; ZHANG, J.; WEI, Z. Deep learning for sequence pattern recognition. In: IEEE. *Networking, Sensing and Control (ICNSC), 2018 IEEE 15th International Conference on*. [S.l.], 2018. p. 1–6.

GEST, H. The discovery of microorganisms by robert hooke and antoni van leeuwenhoek, fellows of the royal society. *Notes and records of the Royal Society of London*, The Royal Society, v. 58, n. 2, p. 187–201, 2004.

GIRIBET, G. Efficient tree searches with available algorithms. *Evolutionary bioinformatics online*, SAGE Publications, v. 3, p. 341, 2007.

GLENN, T. C. Field guide to next-generation dna sequencers. *Molecular ecology resources*, Wiley Online Library, v. 11, n. 5, p. 759–769, 2011.

GLOVER, F.; LAGUNA, M. *Tabu search*. [S.l.]: Springer, 1999.

GOLDBERG, D. E.; HOLLAND, J. H. Genetic algorithms and machine learning. *Machine learning*, Springer, v. 3, n. 2, p. 95–99, 1988.

GONDRO, C.; KINGHORN, B. A simple genetic algorithm for multiple sequence alignment. *Genetics and Molecular Research*, v. 6, n. 4, p. 964–982, 2007.

GOTOH, O. An improved algorithm for matching biological sequences. *Journal of molecular biology*, Elsevier, v. 162, n. 3, p. 705–708, 1982.

GOULD, S. J. *Ontogeny and phylogeny*. [S.l.]: Harvard University Press, 1977.

GUERRA, M. K. M.; BUCKLER, E. S. k-mer grammar uncovers maize regulatory architecture. *bioRxiv*, Cold Spring Harbor Laboratory, p. 222927, 2017.

GUINDON, S.; GASCUEL, O. A simple, fast, and accurate algorithm to estimate large phylogenies by maximum likelihood. *Systematic biology*, Society of Systematic Zoology, v. 52, n. 5, p. 696–704, 2003.

GUSFIELD, D. *Algorithms on strings, trees and sequences: computer science and computational biology*. [S.l.]: Cambridge university press, 1997.

HIGGINS, D. G.; SHARP, P. M. Clustal: a package for performing multiple sequence alignment on a microcomputer. *Gene*, Elsevier, v. 73, n. 1, p. 237–244, 1988.

HIROSAWA, M. et al. Comprehensive study on iterative algorithms of multiple sequence alignment. *Bioinformatics*, Oxford University Press, v. 11, n. 1, p. 13–18, 1995.

HSIEH, S.-Y. et al. An enhanced algorithm for reconstructing a phylogenetic tree based on the tree rearrangement and maximum likelihood method. In: SPRINGER. *International Conference on Intelligent Computing*. [S.l.], 2015. p. 530–541.

KATOH, K. et al. Mafft version 5: improvement in accuracy of multiple sequence alignment. *Nucleic acids research*, Oxford University Press, v. 33, n. 2, p. 511–518, 2005.

KATOH, K. et al. Mafft: a novel method for rapid multiple sequence alignment based on fast fourier transform. *Nucleic acids research*, Oxford Univ Press, v. 30, n. 14, p. 3059–3066, 2002.

KATOH, K.; ROZEWICKI, J.; YAMADA, K. D. Mafft online service: multiple sequence alignment, interactive sequence choice and visualization. *Briefings in bioinformatics*, 2017.

KATOH, K.; STANDLEY, D. M. Mafft multiple sequence alignment software version 7: improvements in performance and usability. *Molecular biology and evolution*, SMBE, v. 30, n. 4, p. 772–780, 2013.

KAUR, H.; CHAND, L. Biological sequence alignment using varied optimization algorithms. In: IEEE. *Inventive Computation Technologies (ICICT), International Conference on*. [S.l.], 2016. v. 1, p. 1–5.

KHURI, S. A bioinformatics track in computer science. In: ACM. *ACM SIGCSE Bulletin*. [S.l.], 2008. v. 40, n. 1, p. 508–512.

KIDD, K. K.; SGARAMELLA-ZONTA, L. A. Phylogenetic analysis: concepts and methods. *American journal of human genetics*, Elsevier, v. 23, n. 3, p. 235, 1971.

KIRKPATRICK, S. et al. Optimization by simulated annealing. *science*, World Scientific, v. 220, n. 4598, p. 671–680, 1983.

KOCEV, D. et al. Tree ensembles for predicting structured outputs. *Pattern Recognition*, Elsevier, v. 46, n. 3, p. 817–833, 2013.

LASSMANN, T.; SONNHAMMER, E. L. Kalign–an accurate and fast multiple sequence alignment algorithm. *BMC bioinformatics*, BioMed Central, v. 6, n. 1, p. 298, 2005.

LEMOS, M.; ARAGAO, M. V. S. P.; CASANOVA, M. A. *Padrões em Biossequências*. [S.l.]: PUC, 2003.

LEMOS, M.; BASÍLIO, A.; CASANOVA, M. A. *Um estudo dos algoritmos de montagen de fragmentos de DNA*. [S.l.]: PUC, 2003.

LEMOS, M.; CASANOVA, M. A. *Algoritmos para análise de sequências*. [S.l.]: PUC, 2000.

LI, M.; TROMP, J.; ZHANG, L. On the nearest neighbour interchange distance between evolutionary trees. *Journal of Theoretical Biology*, Elsevier, v. 182, n. 4, p. 463–467, 1996.

LIEW, A. W.-C.; YAN, H.; YANG, M. Pattern recognition techniques for the emerging field of bioinformatics: A review. *Pattern Recognition*, Elsevier, v. 38, n. 11, p. 2055–2073, 2005.

LIU, L. et al. Comparison of next-generation sequencing systems. *BioMed Research International*, Hindawi Publishing Corporation, v. 2012, 2012.

LIU, Y.; SCHMIDT, B.; MASKELL, D. L. Msaprobs: multiple sequence alignment based on pair hidden markov models and partition function posterior probabilities. *Bioinformatics*, Oxford University Press, v. 26, n. 16, p. 1958–1964, 2010.

LÓPEZ-IBÁÑEZ, M.; STÜTZLE, T.; DORIGO, M. Ant colony optimization: A component-wise overview. *Handbook of Heuristics*, Springer, p. 1–37, 2016.

MOORE, R. E.; YOUNG, M. K.; LEE, T. D. Qscore: an algorithm for evaluating sequest database search results. *Journal of the American Society for Mass Spectrometry*, Springer, v. 13, n. 4, p. 378–386, 2002.

MORGENSTERN, B. Multiple sequence alignment with dialign. In: *Multiple Sequence Alignment Methods*. [S.l.]: Springer, 2014. p. 191–202.

MORGENSTERN, B. et al. Dialign: finding local similarities by multiple sequence alignment. *Bioinformatics*, Oxford Univ Press, v. 14, n. 3, p. 290–294, 1998.

MORRISON, D.; MORGAN, M.; KELCHNER, S. Molecular homology and multiple sequence alignment: an analysis of concepts and practice. *Australian Systematic Botany*, CSIRO, 2015.

MOSS, J.; JOHNSON, C. G. An ant colony algorithm for multiple sequence alignment in bioinformatics. In: SPRINGER. *Artificial Neural Nets and Genetic Algorithms*. [S.l.], 2003. p. 182–186.

NAIDU, V.; NARAYANAN, A. Needleman-wunsch and smith-waterman algorithms for identifying viral polymorphic malware variants. In: IEEE. *2016 IEEE 14th Intl Conf on Dependable, Autonomic and Secure Computing, 14th Intl Conf on Pervasive Intelligence and Computing, 2nd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech)*. [S.l.], 2016. p. 326–333.

NEEDLEMAN, S. B.; WUNSCH, C. D. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, Elsevier, v. 48, n. 3, p. 443–453, 1970.

NELESEN, S. M. et al. The effect of the guide tree on multiple sequence alignments and subsequent phylogenetic analysis. In: *Pacific Symposium on Biocomputing*. [S.l.: s.n.], 2008. v. 13, n. 2008, p. 25–36.

NOTREDAME, C.; HIGGINS, D. G. Saga: sequence alignment by genetic algorithm. *Nucleic acids research*, Oxford Univ Press, v. 24, n. 8, p. 1515–1524, 1996.

NOTREDAME, C.; HIGGINS, D. G.; HERINGA, J. T-coffee: A novel method for fast and accurate multiple sequence alignment. *Journal of molecular biology*, Elsevier, v. 302, n. 1, p. 205–217, 2000.

NOTREDAME, C.; HOLM, L.; HIGGINS, D. G. Coffee: an objective function for multiple sequence alignments. *Bioinformatics*, Oxford Univ Press, v. 14, n. 5, p. 407–422, 1998.

OGDEN, T. H.; ROSENBERG, M. S. Multiple sequence alignment accuracy and phylogenetic inference. *Systematic biology*, Society of Systematic Zoology, v. 55, n. 2, p. 314–328, 2006.

PAL, S. K.; WANG, P. P. *Genetic algorithms for pattern recognition*. [S.l.]: CRC press, 2017.

PAPADIMITRIOU, C. H.; STEIGLITZ, K. *Combinatorial optimization: algorithms and complexity*. [S.l.]: Courier Corporation, 1998.

PENN, O. et al. An alignment confidence score capturing robustness to guide tree uncertainty. *Molecular Biology and Evolution*, Oxford University Press, v. 27, n. 8, p. 1759–1767, 2010.

PROSDOCIMI, F. et al. Bioinformática: manual do usuário. *Biotecnologia Ciência & Desenvolvimento*, v. 29, p. 12–25, 2002.

REEVES, C. R. Genetic algorithms for the operations researcher. *INFORMS journal on computing*, INFORMS, v. 9, n. 3, p. 231–250, 1997.

RIAZ, T.; WANG, Y.; LI, K.-B. Multiple sequence alignment using tabu search. In: AUSTRALIAN COMPUTER SOCIETY, INC. *Proceedings of the second conference on Asia-Pacific bioinformatics-Volume 29*. [S.l.], 2004. p. 223–232.

RIDDER, D. de; RIDDER, J. de; REINDERS, M. J. Pattern recognition in bioinformatics. *Briefings in bioinformatics*, Oxford University Press, v. 14, n. 5, p. 633–647, 2013.

RUBIN, G. M. et al. Comparative genomics of the eukaryotes. *Science*, American Association for the Advancement of Science, v. 287, n. 5461, p. 2204–2215, 2000.

RZHETSKY, A.; NEI, M. Theoretical foundation of the minimum-evolution method of phylogenetic inference. *Molecular biology and evolution*, v. 10, n. 5, p. 1073–1095, 1993.

SAITOU, N.; NEI, M. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular biology and evolution*, v. 4, n. 4, p. 406–425, 1987.

SCHWEFEL, H.-P.; RUDOLPH, G. *Contemporary evolution strategies*. [S.l.]: Springer, 1995.

SIEVERS, F.; HIGGINS, D. G. Clustal omega, accurate alignment of very large numbers of sequences. In: *Multiple sequence alignment methods*. [S.l.]: Springer, 2014. p. 105–116.

SIEVERS, F.; HIGGINS, D. G. Clustal omega for making accurate alignments of many protein sequences. *Protein Science*, Wiley Online Library, v. 27, n. 1, p. 135–145, 2018.

SIEVERS, F. et al. Fast, scalable generation of high-quality protein multiple sequence alignments using clustal omega. *Molecular systems biology*, EMBO Press, v. 7, n. 1, p. 539, 2011.

SLATKIN, M.; MADDISON, W. P. A cladistic measure of gene flow inferred from the phylogenies of alleles. *Genetics*, Genetics Soc America, v. 123, n. 3, p. 603–613, 1989.

SMITH, T. F.; WATERMAN, M. S. Identification of common molecular subsequences. *Journal of molecular biology*, Elsevier, v. 147, n. 1, p. 195–197, 1981.

SOKAL, R. R. A statistical method for evaluating systematic relationship. *University of Kansas science bulletin*, v. 28, p. 1409–1438, 1958.

STALLINGS, W. *Computer organization and architecture: designing for performance*. [S.l.]: Pearson Education India, 2003.

STAMATAKIS, A.; HOOVER, P.; ROUGEMONT, J. A rapid bootstrap algorithm for the raxml web servers. *Systematic biology*, Taylor & Francis, v. 57, n. 5, p. 758–771, 2008.

TAYLOR, W. R. A flexible method to align large numbers of biological sequences. *Journal of Molecular Evolution*, Springer, v. 28, n. 1-2, p. 161–169, 1988.

THOMPSON, J. D.; HIGGINS, D. G.; GIBSON, T. J. Clustal w: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic acids research*, Oxford Univ Press, v. 22, n. 22, p. 4673–4680, 1994.

THOMPSON, J. D. et al. Balibase 3.0: latest developments of the multiple sequence alignment benchmark. *Proteins: Structure, Function, and Bioinformatics*, Wiley Online Library, v. 61, n. 1, p. 127–136, 2005.

TURING, A. M. On computable numbers, with an application to the entscheidungsproblem. *Proceedings of the London mathematical society*, Wiley Online Library, v. 2, n. 1, p. 230–265, 1937.

WALLACE, I. M.; BLACKSHIELDS, G.; HIGGINS, D. G. Multiple sequence alignments. *Current opinion in structural biology*, Elsevier, v. 15, n. 3, p. 261–266, 2005.

WANG, L.; JIANG, T. On the complexity of multiple sequence alignment. *Journal of computational biology*, v. 1, n. 4, p. 337–348, 1994.

WANG, X.-D. et al. A survey of multiple sequence alignment techniques. In: SPRINGER. *International Conference on Intelligent Computing*. [S.l.], 2015. p. 529–538.

WATERMAN, M.; SMITH, T.; BEYER, W. Some biological sequence metrics. *Advances in Mathematics*, v. 20, n. 3, p. 367 – 387, 1976. ISSN 0001-8708. Disponível em: <http://www.sciencedirect.com/science/article/pii/0001870876902024>.

WATSON, J. D.; CRICK, F. H. et al. Molecular structure of nucleic acids. *Nature*, v. 171, n. 4356, p. 737–738, 1953.

WHITMAN, W. B.; COLEMAN, D. C.; WIEBE, W. J. Prokaryotes: the unseen majority. *Proceedings of the National Academy of Sciences*, National Acad Sciences, v. 95, n. 12, p. 6578–6583, 1998.

WU, Y. A practical method for exact computation of subtree prune and regraft distance. *Bioinformatics*, Oxford University Press, v. 25, n. 2, p. 190–196, 2008.

XU, D.; TIAN, Y. A comprehensive survey of clustering algorithms. *Annals of Data Science*, Springer, v. 2, n. 2, p. 165–193, 2015.

YE, Y. et al. Glprobs: Aligning multiple sequences adaptively. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, IEEE Computer Society Press, v. 12, n. 1, p. 67–78, 2015.

YE, Y.; LAM, T.-W.; TING, H.-F. Pnpprobs: a better multiple sequence alignment tool by better handling of guide trees. *BMC bioinformatics*, BioMed Central, v. 17, n. 8, p. 285, 2016.

ZAFALON, G. et al. A parallel approach of coffee objective function to multiple sequence alignment. In: IOP PUBLISHING. *Journal of Physics: Conference Series*. [S.l.], 2015. v. 633, n. 1, p. 012084.

ZAFALON, G. F. D. *Aplicação de estratégias híbridas em algoritmos de alinhamento múltiplo de sequências para ambientes de computação paralela e distribuída.* Tese (Doutorado) — Universidade de São Paulo, 2014.

ZHAN, Q. et al. Improving multiple sequence alignment by using better guide trees. *BMC bioinformatics*, BioMed Central, v. 16, n. 5, p. S4, 2015.

ZHU, X. et al. Parallel implementation of mafft on cuda-enabled graphics hardware. *Computational Biology and Bioinformatics, IEEE/ACM Transactions on*, IEEE, v. 12, n. 1, p. 205–218, 2015.