Alinhamento Múltiplo de Sequências Utilizando Algoritmos Genéticos

Sergio Jeferson Rafael Ordine

Agosto 2011

Resumo

O Alinhamento Múltiplo de Sequências é uma das principais ferramentas nos campos da bioinformática e biologia computacional. A resolução deste problema, contudo, envolve grandes dificuldades computacionais e biológicas, o que levou ao surgimento de diversas aproximações e heurísticas para sua resolução.

A proposta de projeto aqui apresentada consiste em estudar uma destas heurísticas, o uso de Algoritmos Genéticos para a resolução deste problema, procurando possíveis pontos de melhoria nos métodos hoje existentes.

1 Introdução

Um campo de estudo muito explorado nas áreas de Biologia Computacional e Bioinformática é o Alinhamento Múltiplo de Sequências (MSA, do inglês *Multiple Sequence Alignment*). Sua utilização é muito vasta, sendo uma ferramenta de grande importância para a construção de árvores filogenéticas, a identificação de *motifs* conservados, a predição de funções de proteínas e de suas estruturas secundárias e terciárias [25, 37]. Além disto é útil também na definição de primers para PCR (*Polimerase Chain Reaction*) e validação de dados [25].

O problema do Alinhamento Múltiplo de Sequências pode ser descrito como o arranjo de três ou mais sequências de "resíduos" (nucleotídeos ou aminoácidos), no formato de colunas, de forma a evidenciar sua origem evolucionária comum [12].

A solução do problema de MSA é altamente complexa, envolvendo três dificuldades técnicas distintas: a escolha de sequências a serem alinhadas, a escolha de uma função de comparação adequada entre estas e a otimização desta função [37]. A resolução destes problemas é de natureza multidisciplinar, envolvendo os campos de Estatística, Biologia e Ciência da Computação.

Devemos considerar além da natureza multidisciplinar do problema outros fatores que o tornam complexo: em termos biológicos é difícil definir precisamente a qualidade de um alinhamento de sequências [25, 36]; em termos computacionais, este problema é reconhecidamente MAX-SNP-Difícil [49], o que torna o esforço computacional proibitivo para solucionar, de forma ótima, a maior parte das instâncias práticas deste problema.

Devido a tal impossibilidade, diversas heurísticas foram propostas para a obtenção de bons resultados para MSA. Entre estas destacam-se duas categorias principais: algoritmos progressivos e iterativos.

Algoritmos progressivos [16] ordenam as sequências alvo de acordo com um certo critério e de forma incremental constroem o alinhamento destas através do alinhamento de um novo

par de sequências ou de um sub-alinhamento à solução. São as soluções mais utilizadas atualmente, sendo que uma das aplicações mais conhecidas é o ClustalW [43].

A grande vantagem dos algoritmos progressivos é seu desempenho. Devido a sua natureza, porém, este tipo de algoritmo não consegue recuperar-se de decisões prévias, podendo causar uma degeneração no resultado final (em especial se a decisão ruim for tomada em um passo inicial do processo). Reside neste fato sua principal desvantagem.

Algoritmos iterativos são estruturados na forma de passos nos quais o alinhamento é gradualmente refinado. Existem diversas soluções iterativas para o problema MSA, baseadas em abordagens distintas tais como: modelos ocultos de Markov, arrefecimento simulado (*simulated annealing*) [26], amostragem de Gibbs (*Gibbs sampling*) [27], algoritmos genéticos [19, 34], entre outras.

Os algoritmos iterativos demandam, em geral, mais tempo de processamento que algoritmos progressivos. Contudo, são capazes de efetuar buscas mais amplas no espaço de soluções. Isto faz com que tais algoritmos possam sobrepujar problemas que prejudicam algoritmos progressivos, devendo ser vistos como alternativas para estes ou, pelo menos, como possíveis refinadores de suas soluções.

O trabalho aqui descrito propõe-se a estudar os métodos de alinhamento múltiplo de sequências, com foco principal nas soluções utilizando algoritmos genéticos, buscando sugerir melhorias nestes métodos que possam ser úteis à comunidade.

O restante deste documento delimita o escopo e objetivos do trabalho proposto e está estruturado da seguinte forma. A Seção 2 apresenta a base conceitual necessária ao entendimento e desenvolvimento do projeto. Os trabalhos correlatos a esta proposta são descritos na Seção 3. A seguir, na Seção 4, apresenta-se os primeiros trabalhos desenvolvidos no âmbito deste projeto. Por fim, na Seção 5, é apresentado o cronograma de atividades a serem desenvolvidas.

2 Conceitos Básicos

Esta seção apresenta os conceitos necessários para a compreensão e desenvolvimento do trabalho proposto. A Seção 2.1 descreve a base biológica associada a este trabalho. Em seguida, a Seção 2.2 apresenta as soluções existentes para este problema. Por fim, a Seção 2.3 descreve brevemente os algoritmos genéticos.

2.1 Conceitos Biológicos

Esta seção descreve os conceitos biológicos associados ao trabalho proposto. São apresentados aqui os conceitos de DNA e RNA, proteínas e Seleção Natural.

2.1.1 DNA e RNA

Todo ser vivo possui a capacidade de se reproduzir, seja de forma sexuada ou assexuada, gerando uma prole que conserva parcialmente suas características. O mecanismo responsável por este fenômeno baseia-se, principalmente, em moléculas capazes de autorreplicação existentes no interior das células, chamadas DNA (do inglês *Deoxyribonucleic acid*, ácido desoxirribonucléico) e RNA (do inglês *Ribonucleic acid*, ácido ribonucléico) [8]. A estrutura destas moléculas foi descoberta em 1953 por Watson e Crick [50].

As moléculas de DNA são compostas por cadeias de moléculas mais simples chamadas de nucleotídeos. Estas tem em sua estrutura uma pentose, uma base nitrogenada e um

grupo fosfato. Sua construção permite que estas se conectem a dois outros nucleotídeos, formando assim longas cadeias (polinucleotídeos).

Nas moléculas de DNA os nucleotídeos são compostos sempre pela mesma pentose (desoxiribose). Como o grupo fosfato também é único, a distinção dos nucleotídeos está na base nitrogenada que a forma. Sendo assim, no DNA encontram-se quatro tipos de nucleotídeos distintos: as purinas Adenina (representada pela letra A) e Guanina (G), e as pirimidinas Timina (T) e Citosina (C).

Os nucleotídeos conectam-se também através de pontes de hidrogênio. Estas conexões ocorrem respeitando afinidades químicas e elétricas entre os nucleotídeos: uma molécula de Adenina pode conectar-se apenas a moléculas de Timina enquanto moléculas de Citosina conectam-se apenas a moléculas de Guanina e vice-versa.

Este conjunto de características faz com que a molécula de DNA apresente-se com um formato espiralado como uma cadeia dupla de nucleotídeos, unidas por pontes de hidrogênio, sendo que cada cadeia contém os nucleotídeos complementares à outra. Tal estrutura permite que, através de processos químicos, uma molécula de DNA seja dividida em 2 fitas e que, dados suficientes nucleotídeos, cada uma das fitas seja recomposta como uma nova molécula de DNA idêntica a molécula original. A estrutura da molécula de DNA e dos nucleotídeos que a formam podem ser vistas na Figura 1.

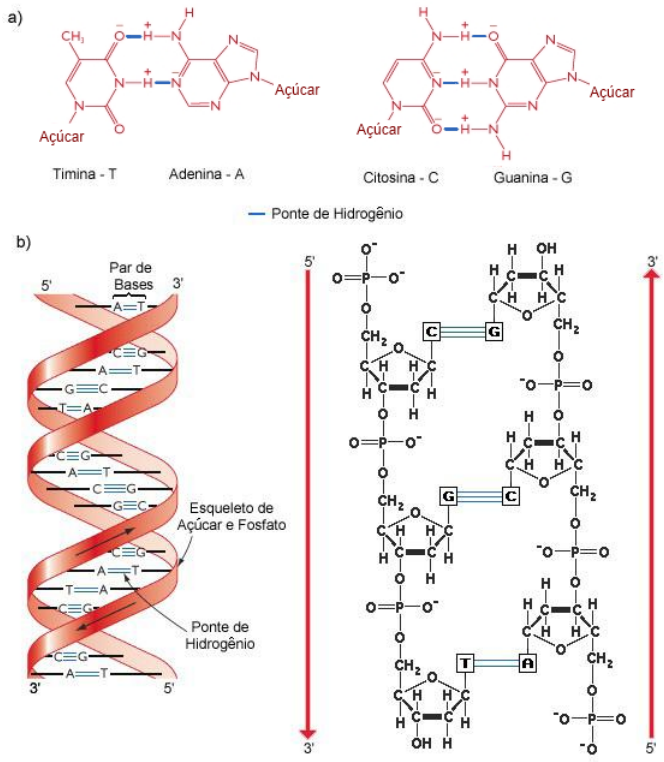

Figura 1: a) Bases que compõem o DNA e ligações de bases complementares por pontes de hidrogênio. b) Estrutura de dupla hélice do DNA com terminações 5' (fosfato livre) e 3' (hidroxil livre). **Fonte:** Souza [42] traduzido de Brown [8].

As molécula de RNA tem certas similaridades com as moléculas de DNA, embora sejam muito mais frágeis e normalmente estruturadas como uma fita simples (embora também

existam fitas duplas de RNA). Esta molécula é formado por nucleotídeos com uma pentose diferente do DNA (ribose) e possui a base nitrogenada Uracila, que assume o papel da base nitrogenada Timina (T) que não está presente no RNA. Da mesma forma que esta, a Uracila conecta-se através de uma ponte de hidrogênio com as moléculas de Adenina(A).

2.1.2 Proteínas

Proteínas [7] são os blocos fundamentais que formam todos os seres vivos, responsáveis por diversas funções biológicas. Estas moléculas são estruturas complexas, formadas por unidades mais simples conhecidas como aminoácidos.

Os aminoácidos são moléculas orgânicas, formadas por uma molécula de carbono central (C_α), conectada a um átomo de hidrogênio, a um grupo amina (NH_2), a um grupo carboxila ($COOH$) e a uma cadeia lateral. Esta cadeia lateral é o que diferencia os diversos aminoácidos. Devido a esta estrutura, as moléculas de aminoácido, com exceção da glicina, podem ter dois isômeros, com os grupos carboxila, a cadeia lateral e o grupo amina organizados de forma dextrógira (CO-R-N) ou levógira (N-R-CO). Na natureza, a forma dextrógira (horária) é a única presente [7].

Os aminoácidos se ligam através de uma reação onde ocorre a condensação do grupo carboxila de um aminoácido com o grupo amina de outro, liberando uma molécula de água (H_2O) e formando uma ligação chamada peptídica entre os dois resíduos dos aminoácidos. É possível que as extremidades amino-terminal e corboxi-terminal sofram o mesmo processo, permitindo encadear inúmeros aminoácidos e assim, formar as cadeias protéicas. Uma visão esquemática dos aminoácidos e ligações pepidíticas pode ser vista na Figura 2 .

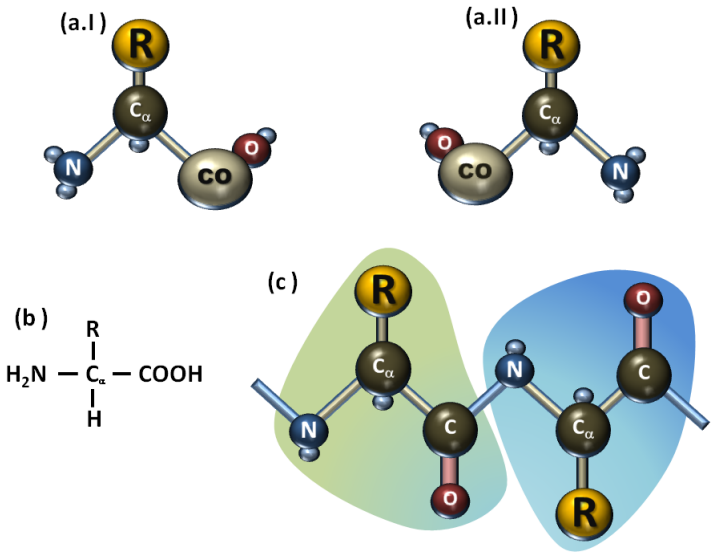

Figura 2: a) Representação de uma molécula de aminoácido em suas duas formas quirais: levógira (I) e destrógira (II). b) Fórmula química da molécula de aminoácido. c) Representação de ligações pepetídicas (linhas azuis) e dos resíduos de aminoácidos (regiões sombreadas) em uma molécula de proteína. Imagens baseadas no livro de Branden e Tooze [7].

Devido as características físico-químicas diversas dos aminoácidos, várias interações ocorrem entre os resíduos de uma molécula de proteína. Tais interações fazem com que

estas moléculas possuam estruturas tridimensionais muito distintas e são estas estruturas que definem as funções desta proteína.

A estrutura de uma proteína é dividida em diversos níveis: a chamada estrutura primária, definida pela sequência de resíduos que a forma; a estrutura secundária, definida principalmente pelas iterações entre resíduos próximos, sendo que as principais estruturas apresentadas são as α-hélices e as folhas-β; a estrutura terciária, resultante das dobras nas estruturas secundárias que definem o que são chamado de domínios; por fim, as estruturas quaternárias, presentes nas proteínas mais complexas, são formadas pela união de diversos domínios distintos.

2.1.3 Seleção Natural

A Seleção Natural é um modelo que descreve a maneira como as espécies adaptam-se ao meio ambiente e evoluem. Esta teoria foi proposta pelos naturalistas ingleses Charles Darwin e Alfred Russel Wallace em 1844, sendo que ambos chegaram a ela de forma independente. O trabalho inicial neste campo foi o livro "A Origem das Espécies por meio da Seleção Natural" de Charles Darwin [9].

Pode-se dizer que o que é chamado usualmente de "teoria darwiniana da evolução" é formada, na verdade por um conjunto de cinco teorias [17]:

1. Linhagens de organismos modificam-se, de forma natural, ao longo do tempo (o conceito de *"evolution as such"*);

2. Todas as espécies existentes divergiram de um ancestral remoto comum;

3. Todas as modificações em organismos vivos ocorrem de forma gradual, causando mudanças maiores apenas após um longo período de tempo (gradualismo);

4. A evolução ocorre através da mudança da proporção de dada característica na população ao longo do tempo;

5. A existência de uma característica favorável em dado indivíduo torna-o mais apto a sobreviver e reproduzir-se, passando esta característica a sua progênie, aumentando assim sua proporção na população geral. Ao longo de um período suficientemente grande de tempo isto pode causar a dominância de uma dada característica e desaparecimento de outras. Esta idéia, a seleção natural, é a tese mais revolucionária vislumbrada por Darwin e Wallace.

Trabalhos posteriores, chamados de Síntese Evolucionária, fizeram a união da teoria da Seleção Natural e das descobertas genéticas, sendo a fundamentação teórica para a biologia evolucionária moderna.

A Teoria da Seleção Natural pode ser vista como um dos trabalhos científicos com maior impacto na história moderna. Em parte isto deve-se a maneira ao mesmo tempo revolucionária e simples como foi proposta; porém, parte de seu impacto deve-se também às profundas implicações filosóficas e religiosas associadas a ela, sendo que até hoje é alvo de controvérsias dentro e fora do meio científico.

2.2 Alinhamento Múltiplo de Sequências - MSA

Esta seção apresenta o problema alvo deste trabalho, o Alinhamento Múltiplo de Sequências (MSA - *Multiple Sequence Alignment*). A Seção 2.2.1 descreve os algoritmos para alinhamento de um par de sequências. A formulação matemática para o problema de MSA é apresendada na Seção 2.2.2. A Seção 2.2.3 apresenta os métodos utilizados para a medição da qualidade de um MSA. Por fim, as Seções 2.2.4 e 2.2.5 descrevem, respectivamente, as heurísticas progressivas e iterativas, apresentando também algumas das principais aplicações que utilizam estas técnicas.

2.2.1 Alinhamentos de Pares de Sequências

Alinhamento de pares de sequências (*pairwise alignment*), como seu nome indica, é o alinhamento de duas sequências buscando maximizar as similaridades entre eles. Needleman e Wunsch [33] propuseram em 1970 uma solução para este problema utilizando programação dinâmica, com complexidade $\Theta(mn)$ onde m e n representam o tamanho das sequências a serem alinhadas.

Muitas variações deste algoritmo foram criadas, sendo importante citar aqui o trabalho de Smith e Waterman [41], que é utilizado para buscar a região mais bem conservada entre duas sequências. Tais algoritmos constroem uma matriz que representa o alinhamento de todos os prefixos de uma sequência α com os da sequência β, aceitando neste alinhamento a inclusão de espaços (*gaps*) para ajustá-lo. O valor de cada posição i, j da matriz é dada pelo melhor valor que pode ser obtido no alinhamento dos prefixos α_i e β_j. Cada posição deste alinhamento pode ser valorado distintamente se contiver letras iguais (*match*), letras distintas (*mismatch*) ou o alinhamento de uma letra com um espaço (*gap*).

A extrapolação deste algoritmo para um número maior de sequências provou-se um problema NP-Completo [49]. Não obstante, algoritmos de alinhamento de pares são muitas vezes utilizados como ferramenta na construção de alinhamentos múltiplos, ao permitir o alinhamento e análise par a par das sequências que o compõe.

2.2.2 Alinhamentos Múltiplos de Sequências

Alinhamentos múltiplos de sequência (MSA) podem ser vistos como um modelo que visa representar os eventos de mutação que ocorreram no conjunto de sequências dado à partir de um ancestral comum. Estas mutações são representadas pelos desalinhamentos encontrados (sejam letras distintas ou *gaps* presentes em dada coluna).

De maneira mais formal, pode-se definir que uma sequência s é uma cadeia de n símbolos de um alfabeto Σ. Dado então um conjunto de k sequências $M = \langle s^1, ..., s^k \rangle$ formadas apenas por símbolos de um mesmo alfabeto Σ e um alfabeto $\Sigma' = \Sigma \cup \{-\}$, onde este caracter adicional representa um *gap*, um alinhamento de M é um conjunto $M' = \langle as^1, ..., as^k \rangle$ onde as^i é uma sequência de símbolos de Σ' que possui todos os caracteres de s^i na mesma ordem em que se encontram nesta, diferindo apenas pela inclusão de gaps. Define-se também que todas as sequências de M' possuem o mesmo tamanho e que não existe uma coluna em M'_j que contenha apenas *gaps* ($\exists\ as^i \in M' : as^i_j \neq -, \forall j : 1 \leq j \leq comprimento(M')$).

2.2.3 Qualidade de um MSA

Um dos desafios e dificuldades associados ao alinhamento de sequências é a necessidade de mensurar a qualidade deste. Isto é feito, via de regra, pela definição de forma explícita

ou implícita de uma função objetivo que quantifica esta qualidade. Esta função deve ser passível de ser utilizada em um algoritmo eficiente para encontrar um alinhamento ótimo – ou próximo do ótimo – de acordo com o critério definido por ela [11].

Uma das formas mais utilizadas na avaliação de alinhamentos de um par de sequências é a chamada soma de pares. Utilizando as definições dadas na Seção 2.2.2 podemos dizer que esta função objetivo é representada como:

$$SP = \sum_{k=1}^{n} \sum_{i=1}^{m-1} \sum_{j=i}^{m} \sigma(as_k^i, as_k^j) \tag{1}$$

Onde m representa o número de sequências, n o comprimento do alinhamento (número de colunas) e σ uma função que define a pontuação entre dois caracteres do alfabeto Σ'. Esta pontuação modela e valora a existência de homologias, mutações – o que ocorre quando duas letras iguais ou diferentes, respectivamente, estão presentes em as_k^i e as_k^j; ou de inserções e remoções – o que ocorre quando do alinhamento de uma letra com um *gap* – nas sequências comparadas. Tal função é utilizada nos algoritmos clássicos de alinhamento de pares, como Needleman e Wunsch [33] e Smith e Waterman [41].

Para homologias e mutações, a pontuação é dada por valores para cada uma destas situações se as sequências são de DNA ou RNA (ao que chamamos de *matches* e *mismatches*). Quando as sequências verificadas são proteínas, é utilizada uma pontuação mais refinada devido as similaridades e diferenças bio-químicas existentes entre os diversos aminoácidos. Esta pontuação é dada em geral na forma de uma matriz de pontuação contendo todas as combinações de aminoácidos possíveis, sendo que as mais utilizadas são as matrizes do tipo PAM [10] e BLOSUM [22].

As matrizes de ambas as famílias são nomeadas como o nome da família seguido de um número. No caso da família PAM, este número isto indica a distância evolutiva entre as duas sequências, ou seja, número maiores indicam matrizes para tratamento de sequências menos similares. Já no caso das matrizes BLOSUM, onde o número indica o grau de similaridade das proteínas utilizadas em sua construção, a escala cresce no sentido inverso e, quanto maior sua numeração, melhor esta é para o tratamento de sequências com maior similaridade.

A pontuação para eventos de remoção e inserção é dada pela pontuação do alinhamento entre um aminoácido e um gap. Esta pontuação em geral é definida de forma empírica [48].

Algumas variações sobre esta função objetivo são utilizadas, como a definição de pesos para sequências (de forma a não supervalorizar a existência de sequências muito similares) e o uso de pontuações afim e semi-afins para *gaps* [2], visto que em geral inserções e remoções ocorrem em blocos na natureza, devendo ser tratadas também no modelo como um único evento evolutivo.

O maior problema no uso deste tipo de função objetivo, para alinhamento de proteínas, está no fato que as pontuações são dadas por matrizes de substituição genéricas, calculadas estatisticamente para que atendam uma ampla gama de alinhamentos. Apesar da flexibilidade dada por esta abordagem, a função pode não ser a mais adaptada para o conjunto de sequências de interesse ao não utilizar similaridades que podem ser naturais a este grupo especificamente [36]. Como alternativas, foram propostos os métodos baseados em Modelos Ocultos de Markov (HMM, do inglês *Hidden Markov Models*) e métodos baseados em consistência (*consistency-based schemes*).

Algoritmos que implementam HMMs utilizam as sequências de entrada para gerar modelos estatísticos, que depois servem de referência para o alinhamento das sequências. O

maior problema com este tipo de método é a necessidade de um número grande de sequências para a geração de um modelo acurado, o que pode ser parcialmente resolvido com o uso de informações extras, como misturas de Dirichlet [36].

Como alternativa os métodos baseados em consistência – ou em perfil – parte do princípio que é melhor alinhar todas as sequências simultaneamente, de forma a utilizar toda a informação disponível para melhorar a qualidade do resultado, ao invés de adotar uma estratégia gulosa para progressivamente construir o alinhamento. Este conceito foi inicialmente introduzido por Gotoh [20] para identificar pontos âncora que diminuissem o espaço de busca de um MSA. Uma reformulação do conceito utilizando multiplicação de matrizes booleanas foi desenvolvida por Vingron e Argos [47].

Vários algoritmos distintos foram criados utilizando esta abordagem: a aplicação DIALIGN [31] utiliza alinhamentos locais sem gaps entre pares de sequência como base para definição de pesos e construção do alinhamento. A aplicação ProbCons [11] utiliza uma abordagem chamada consistência probabilística que combina o método de soma de pares com probabilidades obtidas no uso e treino de um HMM. São utilizados também métodos similares a soma de pares com pesos, ajustados através de algoritmos que refinam as pontuações utilizadas considerando as sequências alvos, como o COFFEE [36] e T-COFFEE [35], entre muitas outras.

2.2.4 Soluções baseadas em Algoritmos Progressivos

Algoritmos progressivos para MSA são heurísticas que tentam construir a solução em passos, incluindo em cada uma deles uma nova sequência ou sub-alinhamento computado à solução. Estas soluções são altamente utilizadas pois apresentam uma utilização muito eficiente de memória e processamento. Todavia, estes algoritmos possuem uma natureza gulosa, o que pode aprisioná-los em um mínimo local devido a uma má decisão em alguma passo do processo (em especial nos estágios iniciais).

Esta abordagem foi descrita por Hogeweg e Harper em 1984 [23], sendo reformulada posteriormente por Feng e Dolittle [16]. Pode-se em linhas gerais dividir este algoritmo em algumas etapas: alinhamento das sequências de entrada par a par, construção de uma árvore guia para orientar o processo e a integração das sequências ao alinhamento de acordo com esta árvore. A possibilidade de utilização de diversas estratégias em cada um destes passos permitiu o surgimento de diversas aplicações para alinhamento múltiplo baseadas nesta estratégia.

A solução ClustalW [43], uma das aplicações mais utilizadas e conhecidas para MSA, segue uma abordagem progressiva. Esta solução permite o uso de programação dinâmica ou de uma heurística de aproximação rápida para efetuar os alinhamentos par a par de sequências, organizando a partir destes alinhamentos uma árvore filogenética utilizando o algoritmo Neighbour-Joining [39] (em versões anteriores da aplicação o algoritmo UPGMA era utlizado). Por fim, a árvore gerada é utilizada para a definição de pesos para as sequências (evitando assim a super-valorização de padrões existentes em sequências muito próximas) e para a construção do alinhamento final. Esta construção é sujeita a uma série de parametrizações como o uso de matrizes de pontuação distintas dependendo do estágio do algoritmo (considerando assim a distância evolucionária das sequências envolvidas) e uma série de ajustes nas penalidades dadas a gaps (de acordo com a diferença de tamanho entre as sequências, de suas similaridades, da posição dos gaps entre outros fatores).

A aplicação MUSCLE [14, 15] é uma outra aplicação muito rápida e eficiente baseada numa abordagem, em linhas gerais, progressiva. Esta aplicação inicia a construção do

algoritmo em etapas similares a da abordagem progressiva (análise par a par, construção de árvore guia e do alinhamento), porém executa refinamentos posteriores para melhorar o alinhamento inicialmente criado. Esta variação o aproxima, de certa forma, dos algoritmos iterativos descritos na Seção 2.2.5.

Outra aplicação que podemos citar é o T-COFFEE [35]. Tal aplicação utiliza a idéia de consenso para melhorar a qualidade do alinhamento gerado e minimizar o efeito que decisões erradas podem ter na estrutura gulosa dos algoritmos progressivos. Esta aplicação é uma implementação do algoritmo de mesmo nome (sigla para *Tree-based Consistency Objective Function for alignment Evaluation*) que por sua vez é um aprimoramento do algoritmo COFFEE (*Consistency Objective Function for alignment Evaluation*) [36]. O grande diferencial deste algoritmo é a criação de uma biblioteca que armazena informações sobre as relações dos resíduos das $\frac{N(N-1)}{2}$ combinações par a par de sequências de entrada. Esta informação é utilizada para mensurar os pesos das sequências e na pontuação do alinhamento.

2.2.5 Soluções baseadas em Algoritmos Iterativos

Uma outra grande classe de algoritmos para MSA são os algoritmos iterativos. Estes algoritmos partem de alinhamentos previamente construídos e gradualmente o refinam utilizando alguma heurística. Tal abordagem reduz sensivelmente o problema encontrado em algoritmos gulosos quanto ao aprisionamento em mínimos locais ou a degradação de qualidade devido a desalinhamentos. Por outro lado, estes algoritmos demandam mais tempo e processamento que algoritmos progressivos, o que torna as soluções baseadas nesta abordagem menos utilizadas que as soluções baseadas em algoritmos progressivos.

Uma solução sofisticada e com resultados muito bons é o PRRP, construído a partir de um algoritmo iterativo proposto por Gotoh [21]. Esta solução baseia-se na otimização simultânea do valor do alinhamento e dos pesos utilizados. O algoritmo termina quando os pesos convergem.

As soluções baseadas em algoritmos estocásticos utilizam-se de diversas abordagens como: arrefecimento simulado (*simulated annealing*) [26], amostragem de Gibbs (*Gibbs sampling*) [27], modelos ocultos de Markov e algoritmos genéticos.

Devido ao foco deste trabalho temos interesse especial em algoritmos bio-inspirados, em especial algoritmos genéticos. Tais algoritmos tentam reproduzir comportamentos de entidades biológicas para resolver problemas de otimização. Entre estes algoritmos podemos incluir, fora os já citados algoritmos genéticos [51]: algoritmos de Sistema de Formigas (*Ant System*), algoritmos de Sistema de Colônias de Formigas (*Ant Colony System*), algoritmos de Abelhas (*Bee Algorithms*) e algortimos de Vaga-Lumes (*Firefly Algorithm*).

Alguns algoritmos para MSA foram criados baseados em algoritmos de colônia de formigas. Estes algoritmos tentam simular a maneira como formigas estabelecem um caminho entre seu formigueiro e alguma fonte de alimento, indicando este com seus feromônios. Outras formigas utilizam esta informação olfativa para encontrar o caminho e reforçá-lo depositando mais um pouco de feromônio. Esta abordagem é utilizada nos algoritmos para otimizar caminhos em grafos.

O algoritmo AntAlign [32] reforça regiões bem conservadas entre sequências através de uma "trilha de feromônios", que é utilizada posteriormente para remover *gaps* mal posicionados nestes trechos.

Existem ainda algoritmos que combinam o algoritmo de colônia de formigas com algoritmos genéticos [28]. Tais soluções, assim como as soluções desenvolvidas puramente sobre

algoritmos genéticos, são detalhados na Seção 3.1.

2.3 Algoritmos Genéticos - GA

Algoritmos genéticos são algoritmos inspirados no processo de seleção natural. Simulações computacionais do processo evolutivo foram inicialmente propostas por Barricelli [5], mas a idéia de algoritmos genéticos e sua popularização deve-se em muito ao trabalho de Holland [24], no início dos anos 70.

Algoritmos genéticos representam uma possível solução de um problema de otimização como sendo um indivíduo (representado por uma estrutura de dado chamada cromossomo). Estes indivíduos são agrupados em uma população, sujeita ao processo de seleção natural. A indicação de quais indivíduos são os mais aptos dentre a população é dada por uma função objetivo, que depende da natureza do problema tratado. Uma implementação possível deste tipo de algoritmo é apresentada no Algoritmo 1.

Algoritmo 1: Genetic Algorithm

```
Data: Set of Possible Solutions (S)
Result: chromosome c (the best result for S)
Population ← CreateInitialPopulation(S)
while Stop Criteria not reached do
    BreedingPopulation ← SelectForBreeding(Population)
    Population ← Population ∪ Offspring(BreedingPopulation)
    Population ← Select(Population)
c ← Max(Population)
return c
```

O algoritmo inicia pela criação de uma população inicial, que pode ser aleatória ou baseada em algum critério que produz soluções possíveis para o problema. Em seguida, o algoritmo iterativamente seleciona indivíduos para reprodução.

A reprodução é produzido por operadores que criam novos indivíduos a partir dos indivíduos selecionados. Podem ser utilizados operadores de mutação, que alteram um indivíduo em algum ponto, gerando uma nova possível solução; ou operadores de *crossover*, que mesclam partes de dois indivíduos, tentando combinar pontos fortes de ambos.

Estes indivíduos são combinados a população existentes e submetidos novamente a um processo de seleção. Os indivíduos que sobrevivem a este processo são mantidos na população, sendo os outros descartados.

Por fim, este processo pode ser repetido até que uma condição de parada seja alcançada, podendo ser esta o número de gerações, número de gerações sem que a população apresente melhoria, encontro de um valor alvo ou qualquer outro critério. Quando a condição é finalmente alcançada, o algoritmo retorna o indivíduo (solução) com o melhor resultado.

Para executar a seleção de indivíduos existem diversos algoritmos. Um deles é o algoritmo de amostragem estocástica, ou algoritmo "de roleta", onde é criada uma roleta desigualmente dividida, onde cada indivíduo recebe uma área proporcional a sua qualidade em relação a qualidade geral da população, isto torna um indivíduo mais apto propenso a ser selecionado, porém não exclui a possibilidade de um indivíduo menos apto também o ser.

Outro possível algoritmo é o torneio: neste método dois indivíduos são sorteados através

do mecanismo de roleta e o mais apto entre eles é selecionado. O indivíduo "derrotado" é mantido e um novo indivíduo é sorteado da mesma forma para então ocorrer novamente a seleção. Este processo é repetido até que o conjunto a ser selecionado esteja completo.

3 Detalhamento do Problema

Esta seção apresenta o problema específico que será tratado pelo projeto, a descrição dos trabalhos anteriores utilizando algoritmos genéticos para a geração de alinhamentos múltiplos de sequências e as ferramentas que podem ser utilizadas para análise dos resultados obtidos.

3.1 Soluções para MSA utilizando Algoritmos Genéticos

Nesta seção apresenta-se os trabalhos encontrados na literatura relacionados ao projeto proposto. Estão descritas aqui a aplicação SAGA, o primeiro alinhador utilizando GA desenvolvido, e também a solução mais conhecida, assim como os trabalhos posteriormente desenvolvidos.

3.1.1 SAGA

O trabalho pioneiro no uso de algoritmos genéticos para a resolução de MSA é a aplicação SAGA, proposto e desenvolvido por Notredame e Higgins [34]. Esta aplicação utiliza um grande número de operadores (totalizando 22) para *crossover* e mutação. O uso de cada um deles é dada através de *dynamic scheduling*, onde a probabilidade de dado operador ser aplicado é proporcional ao sucesso das operações realizadas por eles nas últimas gerações (inicialmente esta probabilidade é igual para todos). O critério de parada utilizado pelo SAGA é a execução de um determinado número de gerações sem que alguma melhoria no resultado seja apresentada (em geral 100 gerações).

Posteriormente, esta aplicação foi utilizada como base para o teste da função objetivo baseada em consenso COFFEE [36]. Esta função busca criar pontuações específicas para o conjunto de sequências de entrada, respeitando as similaridades existentes especificamente neste conjunto. Para tal, é criada uma biblioteca contendo os pares de resíduos encontrados nos alinhamentos par a par entre as sequências e pesos para estes alinhamentos, sendo estes maiores quanto mais similar for o par envolvido. Os pesos utilizados neste algoritmo seguem uma abordagem oposta a muitos outros (que diminuem os pesos quanto maior a similaridade apresentada) pois visam forçar que sequências próximas sejam alinhadas corretamente; não impedir que informações de uma sub-família prevaleçam no alinhamento final, o que é garantido pela própria biblioteca.

Uma pesquisa posterior de Thomsen e Boomsma [46] testou a qualidade de alinhamentos gerados pela aplicação SAGA com diversos parâmetros. Vários resultados interessantes foram mostrados. Primeiramente o mecanismo de *dynamic scheduling* mostrou não melhorar significativamente os resultados gerais, o que pode ser efeito direto da natureza estocástica dos operadores. Além disto o uso dos operadores de *crossover* também não afetou o resultado significativamente.

Tais fatores indicam que algoritmos genéticos baseados em uma estrutura mais simples podem ter resultados tão bons ou superiores aos obtidos pelo SAGA, fato que pode ser visto nos resultados obtidos por Botta e Negro na aplicação PWMAligner [6]. Estes resultados são um fator motivador para o projeto proposto.

3.1.2 Em Busca da Simplicidade

Vários trabalhos posteriores apresentam versões mais simples de algoritmos genéticos do que o proposto pela solução SAGA para a resolução do problema de MSA.

Zhang e Wong [52] propuseram em 1997 uma solução baseada no alinhamento exato de colunas, o que mostrou bons resultados de forma muito eficiente para a época, porém devido a sua própria estrutura, este algoritmo limita-se ao tratamento de sequências com alto grau de identidade.

Gondro e Kinghorn [19] desenvolveram a solução MSA-GA (2007), que utiliza os alinhamentos como cromossomos (da mesma forma que o SAGA). Esta solução utiliza um conjunto reduzido de operadores para *crossover* (2, um para combinações horizontais e outro para combinações verticais entre alinhamentos) e mutações (4, todos operando sobre gaps). Tal solução obteve resultados melhores que o ClustalW [43] para diversos conjuntos do BAliBASE 2 [4], porém, um fator que chama a atenção é o tamanho excessivo da população e número de gerações utilizado (1000 indivíduos e 20.000 gerações, respectivamente).

Lee *et al.* [28] propuseram em 2008 o algoritmo GA-ACO, que utiliza o algoritmo de colônia de formigas como um método local para otimizar o espaço de busca de um algoritmo genético para MSA. Tal algoritmo é executado sobre o melhor indivíduo a cada geração.

Meshoul *et al.* [29] e Abdesselem *et al.* [1], propuseram, respectivamente em 2005 e 2006, algoritmos utilizando algoritmos genéticos quânticos como alternativa. Tais algoritmos mesclam conceitos de computação quântica à algoritmos evolutivos para obter resultados melhores que algoritmos genéticos comuns [1].

Botta e Negro [6] apresentaram a solução PWMAligner (2010). Esta solução representa um indivíduo como uma matriz com pesos posicionais (PWM - *Positional Weight Matrix*) contendo a probabilidade de um dado resíduo aparecer naquela coluna do alinhamento. Através de um algoritmo determinístico é possível reconstruir de forma única o alinhamento representado, dado a matriz e o conjunto de sequências do alinhamento.

Além da representação utilizada, outro ponto original deste trabalho é a possibilidade de utilização de um *template* para o cálculo da função objetivo, sendo este uma ou mais sequência sendo alinhadas, o consenso ou mesmo uma sequência externa fornecida. A solução PWMAligner, quando comparada a diversos alinhadores conhecidos (utilizando o BAliBASE 2 como benchmark), apresentou resultados comparáveis ou superiores (em especial no caso das referências Ref2 e Ref3). Os resultados são especialmente significativos quando comparados aos obtidos pelo SAGA. Para tais testes, foram utilizadas uma população de 200 indivíduos e 500 gerações.

3.2 Ferramentas para Análise de MSA

Esta seção apresenta o ferramental que pode ser utilizado na análise dos resultados obtidos no âmbito do projeto. Esta subdivide-se em: Seção 3.2.1, que descreve o ferramental para comparação de alinhamentos contra uma base de referência, com foco no BAliBASE, e Seção 3.2.2, que descreve a SuiteMSA, um conjunto de ferramentas para visualização e análise de alinhamentos múltiplos.

3.2.1 Benchmarks para MSA

Devido ao grande número de soluções e funções objetivo existentes para MSA tornou-se necessário a criação de mecanismos sistemáticos de comparação entre eles, para identifi-

car quais obtinham resultados mais significativos e em que situações isto ocorria. Esta necessidade motivou a criação de diversos mecanismos e bases de dados para *benchmark*.

Entre as bases de dados para *benchmark* de alinhamentos de proteínas podemos citar o BAliBASE [45], HOMSTRAD [30] e PREFAB [15]. Entre os benchmarks para alinhamentos de sequências de RNA podemos citar o BRAliBASE [18].

O *benchmark* BAliBASE, atualmente em sua terceira versão [44], foi o primeiro benchmark criado para comparação de alinhadores de proteínas. Em suas versões anteriores [4,45] a base de dados era formada por conjuntos de alinhamentos criados manualmente o que limitava o número destes conjuntos e o seu tamanho. Para contornar esta desvantagem a última versão foi construída sobre alinhamentos assistidos por algoritmos computacionais, posteriormente refinados manualmente.

Esta base agrupa os alinhamentos em conjuntos de referência (*reference sets*) com características em comum, que permitem validar a qualidade de um alinhador em diversos cenários. Uma breve descrição destes conjuntos, conforme proposto por Thompson [44], é apresentado na Tabela 1.

Conjunto	Descrição
RV11	Sequências com menos de 20% de identidade entre qualquer par
RV12	Sequências que compartilham de 20 a 40% de identidade
RV20	Sequências com alta similaridade contendo uma sequência "orfã"
RV30	Sub-famílias com mais de 40% de similaridade entre si, mas que compartilham menos de 20% de identidade com outras sub-famílias
RV40	Sequências com grandes extensões nas extermidades
RV50	Sequências com grandes inserções internas

Tabela 1: Conjuntos de referência do BAliBASE 3.0

A aplicação **bali_score** permite comparar um alinhamento contra o alinhamento de referência da base de dados do BAliBASE. Esta ferramenta permite a comparação do alinhamento como um todo ou apenas das regiões altamente conservadas, chamadas *core blocks*, utilizando duas métricas: SP, que indica a razão entre a soma de pares do alinhamento dado e do alinhamento de referência, e TC, que indica a razão de colunas do alinhamento que são idênticas em ambos os alinhamentos. Estas duas métricas tem valores entre 0.0 e 1.0, sendo que quanto mais alto o valor, melhor o alinhamento.

3.2.2 SuiteMSA

Para auxiliar a análise dos resultados obtidos sobre um MSA, percebemos a necessidade de uma aplicação que permitisse identificar, de forma simples e rápida, em que pontos o alinhamento realizado divergiu do alinhamento de referência. Uma ferramenta visual seria especialmente útil para tal.

Apesar do grande número de visualizadores de alinhamentos existentes, o único trabalho que parcialmente atendeu esta necessidade foi o pacote SuiteMSA [3].

Este pacote, desenvolvida por Anderson *et al.* utilizando a linguagem JAVA, é uma suite de aplicações para auxilio na análise de MSAs. Incluem-se neste pacote as aplicações: *MSA Viewer*, um visualizador simples de alinhamentos múltiplos; *MSA Comparator*, um visualizador de alinhamentos múltiplos que permite a visualização de um alinhamento contra um alinhamento de referência; *Pixel Plot*, um visualizador de alinhamentos múltiplos que representa cada resíduo como apenas um pixel, permitindo uma visão global (*eagle eye*) do alinhamento; e a aplicação *Phylogeny Viewer*, que apresenta a árvore filogenética associada a um dado alinhamento.

Dentre este ferramental, é de interesse principal do projeto o aplicativo *MSA Comparator*. Tal ferramenta pode auxiliar a análise dos alinhamentos gerados pelos métodos testados no projeto contra uma referência reconhecidamente boa, no caso, os conjuntos do benchmark BAliBASE 3.0 [44].

Um ponto negativo, porém, é que esta ferramenta apenas evidencia colunas idênticas entre ambos os alinhamentos. Este tipo de ferramental, apesar de útil dificilmente será suficiente para uma análise mais detalhada do alinhamento gerado em busca de pontos a melhorar. Como parte deste projeto deseja-se explorar a visualização de outras informações, gerando um ferramental que pode ser útil tanto ao projeto quanto à comunidade em geral.

4 Trabalho em Andamento

Esta seção apresenta os trabalhos já iniciados como parte do projeto aqui descrito. Aqui são descritas a solução ALGAe e alguns experimentos para a geração de populações iniciais.

4.1 ALGAe

A aplicação ALGAe (*ALigning by Genetic Algorithm environment*) foi a primeira aplicação desenvolvida no escopo deste projeto. Esta consiste de um ambiente parametrizável para a execução de um algoritmo genético para MSA. Um resumo estendido deste trabalho foi apresentado no Brazilian Symposium of Bioinformatics de 2011 (BSB 2011) [38].

O ambiente ALGAe, desenvolvido em linguagem JAVA, permite que através da implementação de certas interfaces e do uso de Reflexão, sejam definidos quais algoritmos para seleção, criação de população inicial, função objetivo e operadores serão utilizados em dada execução. Além destes, parâmetros como o tamanho da população e probablidade de execução dos operadores podem ser definidos.

Foram criados, até o presente momento, os seguintes operadores de mutação:

- ***Single Point Mutation*** - Este operador seleciona aleatoriamente uma sequência e uma posição, incluindo um novo *gap* nesta, assim como nas últimas posições de todas as outras sequências, de forma a manter o alinhamento consistente.
- ***Shift Gap Block Mutation*** - Este operador seleciona um bloco de *gaps* e o desloca uma posição para a direita ou esquerda.
- ***Change Gap Block Mutation*** - Este operador seleciona parte de um bloco de *gaps* e um bloco de caracteres vizinhos (a direita ou a esquerda) e os troca de posição.

Também foram criados os seguintes operadores de *crossover*:

- ***Single Point Crossover*** - Este operador executa um corte vertical em um dos alinhamentos pai e um corte compatível com este no outro, que contém os mesmo

resíduos. Eventualmente, é necessário completar algum dos sub-alinhamentos gerados com *gaps* para manter a consistência dos alinhamentos. Os sub-alinhamentos gerados são então recombinados: sendo que o sub-alinhamento esquerdo do primeiro pai é combinado com o sub-alinhamento direito do segundo, e vice-versa, gerando dois novos indivíduos.

- ***Best Partial Alignment Crossover*** - Este operador seleciona, alternadamente, as sequências melhor alinhadas com todas as outras em cada um dos pais, formando dois grupos de sequências. Estes grupos são então concatenados, através do alinhamento dos consensos destes grupos.

- ***Sequence Similarity Crossover*** - Este operador escolhe um nó de forma aleatória em uma árvore filogenética (criada à partir das distâncias par a par entre as sequências de entrada, utilizando UPGMA) e divide os alinhamentos pais baseado nas sequências que se encontram acima e abaixo deste. Os grupos resultantes são concatenados utilizando o alinhamento dos consensos destes dois grupos.

Como teste inicial, foi considerada a execução do ALGAe – com os operadores desenvolvidos até o momento – contra a do alinhador GAADT, baseado em tipos abstratos de dados e tema da dissertação de mestrado de Santos [40]. Foram obtidos os resultados apresentados na Tabela 2 para o conjunto de testes apresentado para o GAADT, utilizando a métrica SP do benchmark BAliBASE versão 2 [4].

	1aab	1fjlA	1hpi	1csy	1tgxA	média
`GAADT` (avg)	88.1	81.4	70.8	70.3	69.2	76.0
`ALGAe` (avg)	88.4	93.8	88.3	76.2	68.5	83.0
`ALGAe` (max)	89.6	100.0	96.2	80.7	77.2	88.7

Tabela 2: Resultados dos alinhadores ALGAe e GAADT para a família de proteínas 1aab e grupos correlatos (em porcentagem).

Em 80% dos cenários testados, ALGAe obteve melhores resultados. Tais testes foram realizados com 20 baterias para cada cenário, com uma população de 250 indivíduos em 2000 gerações para o alinhador ALGAe, sendo que não foi possível obter os valores destes parâmetros para o algoritmo GAADT.

O mesmo conjunto de operadores foi testado também contra o primeiro conjunto de referência do BAliBASE versão 3 (RV11 e RV12) [44], utilizando funções objetivo de soma de pares (SP) e soma de pares com afinidade de gaps (GASP) com diversas matrizes de pontuação. Os resultados obtidos para 20 baterias de teste para cada cenário, com populações de 100 indivíduos em 650 gerações são apresentados na Tabela 3.

Reference	BLOSUM62	BLOSUM80	PAM100	PAM250
RV11 (SP)	33.6	34.9	28.9	28.9
RV11 (GASP)	35.6	38.1	30.5	32.7
RV12 (SP)	73.9	75.7	74.2	75.2
RV12 (GASP)	73.2	75.9	72.8	76.1

Tabela 3: Resultados do ALGAe contra as referência RV11 e RV12 do BAliBASE 3 (em porcentagem).

Os resultados ainda estão aquém dos apresentados pelos alinhadores mais conhecidos. Além disto, o tempo de execução do algoritmo precisa ser melhorado, em especial devido aos operadores de *crossover* que realizam alinhamentos sobre consensos.

Dois pontos a serem investigados nas próximas etapas do projeto são a melhoria dos parâmetros utilizados no ALGAe e a análise da viabilidade de remoção dos operadores de *crossover* sobre consenso (considerando os resultados do trabalho de Thomsen e Boosman [46] sobre o SAGA), visando tornar o algoritmo mais eficiente.

4.2 Criação da População Inicial

A primeira abordagem adotada neste projeto para a criação da população inicial foi a utilização dos alinhamentos par a par de sequências, considerando tanto alinhamentos globais quanto semi-globais gerados pelo algoritmo de Needleman e Wunsch [33]. A criação de cada indivíduo é feito pela seleção aleatória de um par de sequências e um tipo de alinhamento. Em seguida um par de sequências é sorteado, sendo uma que ainda não pertence ao alinhamento e outra pertencente a ele, que funciona como âncora. É selecionado então um tipo de alinhamento e a sequência escolhida é adicionada ao alinhamento múltiplo, utilizando como restrições o alinhamento entre esta e a âncora e os gaps já existentes (respeitando o princípio "*once a gap, always a gap*"). Este processo é repetido até que o alinhamento esteja completo, sendo então incluído na população.

Foram posteriormente desenvolvidos dois algoritmos, baseados em alinhamentos de blocos, que serão incorporados futuramente a população inicial na tentativa de aumentar a diversidade desta e melhorar o resultado final.

O primeiro algoritmo (A1) é baseado em alinhamentos locais. Para cada par de sequências é executado o algoritmo de Smith e Waterman [41] buscando a região de maior similaridade entre elas. Este bloco é armazenado e o processo repetido recursivamente para os trechos das sequências que precedem e sucedem este trecho até que o melhor alinhamento local tenha um tamanho inferior a um limite dado.

Os trechos entre blocos (ou nas extremidades) são então alinhados de forma global, utilizando o algoritmo de Needleman e Wunsch. Baseado na pontuação destes alinhamentos uma árvore guia é criada (utilizando UPGMA) e utilizada na construção da solução.

O segundo algoritmo desenvolvido (A2) busca, utilizando uma janela deslizante de tamanho fixo, blocos contínuos (sem *gaps*) e idênticos entre cada par de sequências a e b. É criado então um grafo onde cada nó é representado por um bloco β encontrado no passo anterior, contendo as posições iniciais β_i e β_j e finais $\beta_i + k$ e $\beta_j + k$ do bloco, onde k representa o tamanho da janela deslizante. As arestas (orientadas) conectam blocos consistentes. Por blocos consistentes entende-se dois blocos β^1 e β^2 de forma que:

$$\mathscr{C}(\beta^1, \beta^2) = \begin{cases} \beta_i^1 + k < \beta_i^2 \wedge \beta_j^1 + k < \beta_j^2 \\ \\ \vee \\ \\ (\beta_i^1 < \beta_i^2 \wedge \beta_i^1 + k \geq \beta_i^2) \wedge \\ (\beta_j^1 < \beta_j^2 \wedge \beta_j^1 + k \geq \beta_j^2) \wedge \\ (\beta_i^2 - \beta_i^1 = \beta_j^2 - \beta_j^1) \end{cases}$$

A primeira condição é satisfeita quando um bloco antecede o outro sem qualquer sobreposição. A segunda condição trata do caso de sobreposição, garantindo que esta seja condizente nas duas sequências ($\beta_i^2 - \beta_i^1 = \beta_j^2 - \beta_j^1$).

Após este passo são criados dois nós especiais representando o início e fim das sequências. O nó inicial é conectado a todos os outros nós e todos os nós são conectados ao nó final. Procura-se então o caminho máximo neste grafo, o que, sendo este orientado e acíclico, pode ser feito em tempo polinomial utilizando-se ordenação topológica. Este caminho representa o maior conjunto consistente de blocos conservados para este par de sequências.

Em seguida, as regiões entre os blocos são alinhadas utilizando-se o algoritmo de alinhamento global de Needleman e Wunsch [33] para cada par de sequências. Uma árvore guia é então construída, considerando que a distância entre as sequências é inversamente proporcional ao número de blocos conservados entre elas. Finalmente, o alinhamento é construído baseado nesta árvore.

Uma variação deste algoritmo (A2a) foi testada, utilizando o alfabetos comprimido Dayhoff6 [13] e janelas de tamanho maior, o que pode facilitar a busca de similaridades entre sequências distantes.

Os resultados obtidos contra os conjuntos de referência do BAliBASE 3 para estes algoritmos podem ser vistos na Tabela 4.

	RV11	RV12	RV20	RV30	RV40	RV50	Média
A1	30.2	67.5	73.1	58.4	58.7	58.0	57.7
A2 (k=5)	20.3	60.7	66.5	49.4	49.4	44.6	48.5
A2a (k=12)	26.9	67.6	73.3	50.8	50.1	51.1	53.3

Tabela 4: Resultados dos algoritmos de alinhamento progressivo testados utilizando o BAliBASE 3.0 (em porcentagem).

5 Proposta e Cronograma

Este trabalho propõe-se a estudar novas abordagens para a utilização de algoritmos genéticos para a resolução do problema de alinhamento múltiplo de sequências. Para realizar tal objetivo, serão estudadas alternativas para os principais parâmetros de algoritmos genéticos: criação de uma população inicial, operadores (mutação e crossover) e métodos de seleção.

As alternativas estudadas serão refinadas utilizando o ambiente ALGAe [38]. Este projeto propõe-se também ao desenvolvimento de ferramentas que auxiliem na análise dos alinhamentos obtidos em busca de pontos de melhoria. Uma visão do cronograma planejado pode ser visto na Tabela 5.

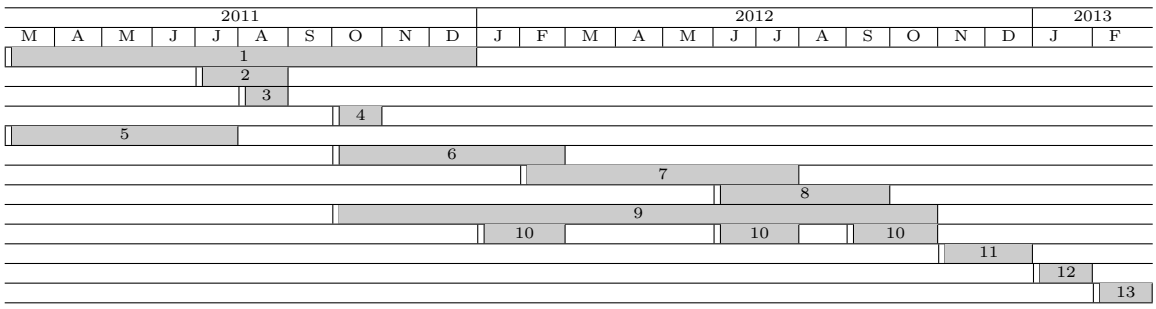

Tabela 5: Cronograma proposto - 2011 à 2013

A descrição destas tarefas é dada abaixo:

1. Cumprimento dos créditos obrigatórios
2. Estudo Dirigido/Preparação do EQM
3. Entrega do texto do EQM
4. Apresentação do EQM
5. Avaliações iniciais - ALGAe e geração da população inicial
6. Geração de População Inicial
7. Operadores (Mutação e Crossover)
8. Métodos de Seleção
9. Refinamento dos resultados obtidos/desenvolvimento de ferramental para análise dos resultados
10. Escrita da Dissertação
11. Finalização e revisão da dissertação
12. Entrega da dissertação
13. Defesa do mestrado

Referências

[1] L. Abdesslem, M. Soham, and B. Mohamed. Multiple sequence alignment by quantum genetic algorithm. In *Proceedings of the 20th international conference on Parallel and distributed processing*, IPDPS'06, Washington, DC, USA, 2006. IEEE Computer Society.

[2] S. Altschul and B. Erickson. Optimal sequence alignment using affine gap costs. *Bulletin of Mathematical Biology*, 48(5):603–616, September 1986.

[3] C. Anderson, C. Strope, and E. Moriyama. SuiteMSA: visual tools for multiple sequence alignment comparison and molecular sequence simulation. *BMC Bioinformatics*, 12(1):184, 2011.

[4] A. Bahr, J. D. Thompson, J. C. Thierry, and O. Poch. BAliBASE (Benchmark Alignment dataBASE): enhancements for repeats, transmembrane sequences and circular permutations. *Nucl. Acids Res.*, 29(1):323–326, January 2001.

[5] N. Barricelli. Esempi numerici di processi di evoluzione. *Methodos*, pages 45–68, 1954.

[6] M. Botta and G. Negro. Multiple sequence alignment with genetic algorithms. In *Computational Intelligence Methods for Bioinformatics and Biostatistics*, volume 6160 of *Lecture Notes in Computer Science*, pages 206–214. Springer, 2010.

[7] C. Branden and J. Tooze. *Introduction to Protein Structure*. Garland Publishing, Inc., 2nd edition, 1999.

[8] T. A Brown. *Genomes*. Oxford: Wiley-Liss, 2nd edition, 2002.

[9] C. Darwin. *The Origin of Species By Means Of Natural Selecion, or the Preservation of Favoured Races in the Struggle for Life*. John Murray, 1st edition, 1859.

[10] M. O. Dayhoff and R. M. Schwartz. Chapter 22: A model of evolutionary change in proteins. In *Atlas of Protein Sequence and Structure*, 1978.

[11] C. B. Do, M. S. P. Mahabhashyam, M. Brudno, and S. Batzoglou. ProbCons: Probabilistic consistency-based multiple sequence alignment. *Genome Res*, 15:330–340, 2005.

[12] L. Duret and S. Abdeddaim. *Bioinformatics: sequence, structure and databanks*, chapter Multiple alignment for structural functional or phylogenetic analyses of homologous sequences. Oxford University Press, Oxford, UK, 2000.

[13] R. C. Edgar. Local homology recognition and distance measures in linear time using compressed amino acid alphabets. *Nucleic Acids Research*, 32(1):380–385, 2004.

[14] R. C. Edgar. MUSCLE: a multiple sequence alignment method with reduced time and space complexity. *BMC bioinformatics*, 5(1):113+, August 2004.

[15] R. C. Edgar. MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic acids research*, 32(5):1792–1797, March 2004.

[16] D. F. Feng and R. F. Doolittle. Progressive sequence alignment as a prerequisite to correct phylogenetic trees. *Journal of molecular evolution*, 25(4):351–360, 1987.

[17] D. J. Futuyma. *Evolution*. Sinauer Associates, Inc., 2nd edition, 2009.

[18] P. P. Gardner, A. Wilm, and S. Washietl. A benchmark of multiple sequence alignment programs upon structural RNAs. *Nucleic Acids Research*, 33(8):2433–2439, 2005.

[19] C. Gondro and B. P. Kinghorn. A simple genetic algorithm for multiple sequence alignment. *Genetics and molecular research : GMR*, 6(4):964–982, 2007.

[20] O. Gotoh. Consistency of optimal sequence alignments. *Bull Math Biol*, 52(4):509–525, 1990.

[21] O. Gotoh. Significant improvement in accuracy of multiple protein sequence alignments by iterative refinement as assessed by reference to structural alignments. *Journal of molecular biology*, 264(4):823–838, December 1996.

[22] S. Henikoff and J. G. Henikoff. Amino acid substitution matrices from protein blocks. *Proc Natl Acad Sci U S A*, 89(22):10915–10919, November 1992.

[23] P. Hogeweg and B. Hesper. The alignment of sets of sequences and the construction of phyletic trees: An integrated method. *Journal of Molecular Evolution*, 20(2):175–186, June 1984.

[24] J. Holland. *Adaptation in natural and artificial systems*. University of Michigan Press, 1975.

[25] C. Kemena and C. Notredame. Upcoming challenges for multiple sequence alignment methods in the high-throughput era. *Bioinformatics*, 25(19):2455–2465, 2009.

[26] J. Kim, S. Pramanik, and M. J. Chung. Multiple sequence alignment using simulated annealing. *Computer applications in the biosciences : CABIOS*, 10(4):419–426, 1994.

[27] C. E. Lawrence, S. F. Altschul, M. S. Boguski, J. S. Liu, A. F. Neuwald, and J. C. Wootton. Detecting subtle sequence signals: a gibbs sampling strategy for multiple alignment. *Science*, 262(5131):208–214, 1993.

[28] Z. Lee, S. Su, C. Chuang, and K. Liu. Genetic algorithm with ant colony optimization (ga-aco) for multiple sequence alignment. *Applied Soft Computing*, 8(1):55 – 78, 2008.

[29] S. Meshoul, A. Layeb, and M. Batouche. A quantum evolutionary algorithm for effective multiple sequence alignment. In *Progress in Artificial Intelligence*, volume 3808 of *Lecture Notes in Computer Science*, pages 260–271. Springer Berlin / Heidelberg, 2005.

[30] K. Mizuguchi, C. M. Deane, T. L. Blundell, and J. P. Overington. HOMSTRAD: a database of protein structure alignments for homologous families. *Protein Sci*, 7(11):2469–2471, November 1998.

[31] B. Morgenstern, K. Frech, A. Dress, and T. Werner. DIALIGN: finding local similarities by multiple sequence alignment. *Bioinformatics*, 14(3):290–294, April 1998.

[32] J. D. Moss and C. G. Johnson. An ant colony algorithm for multiple sequence alignment in bioinformatics. *Artificial neural networks and genetic algorithms*, pages 182–186, 2003.

[33] S. B. Needleman and C. D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3):443–453, March 1970.

[34] C. Notredame and D. G. Higgins. SAGA: sequence alignment by genetic algorithm. *Nucleic acids research*, 24(8):1515–1524, April 1996.

[35] C. Notredame, D. G. Higgins, and J. Heringa. T-Coffee: A novel method for fast and accurate multiple sequence alignment. *Journal of molecular biology*, 302(1):205–217, September 2000.

[36] C. Notredame, L. Holm, and D. G. Higgins. COFFEE: an objective function for multiple sequence alignments. *Bioinformatics*, 14(5):407–422, June 1998.

[37] Cédric Notredame. Recent progress in multiple sequence alignment: a survey. *Pharmacogenomics*, 3(1):131–144, 2002.

[38] S. Ordine, A. Grilo, A. Almeida, and Z. Dias. ALGAe: A test-bench environment for a genetic algorithm-based multiple sequence aligner. In *Brazilian Symposium on Bioinformatics 2011 Digital Proceedings*, pages 57–60, 2011.

[39] N. Saitou and M. Nei. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular Biology and Evolution*, 4(4):406–425, July 1987.

[40] D. Santos. Alinhamento múltiplo de proteínas via algoritmo genético baseado em tipos abstratos de dados. Master's thesis, Universidade Federal de Alagoas, 2008. In Portuguese.

[41] T. F. Smith and M. S. Waterman. Identification of common molecular subsequences. *Journal of molecular biology*, 147(1):195–197, March 1981.

[42] M. A. L. Souza. Alinhamento múltiplo progressivo de sequências de proteínas. Master's thesis, University of Campinas, Brazil, 2010. In Portuguese.

[43] J. D. Thompson, D. G. Higgins, and T. J. Gibson. CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic acids research*, 22(22):4673–4680, November 1994.

[44] J. D. Thompson, P. Koehl, R. Ripp, and O. Poch. BAliBASE 3.0: latest developments of the multiple sequence alignment benchmark. *Proteins*, 61(1):127–136, October 2005.

[45] J. D. Thompson, F. Plewniak, and O. Poch. BAliBASE: a benchmark alignment database for the evaluation of multiple alignment programs. *Bioinformatics (Oxford, England)*, 15(1):87–88, January 1999.

[46] R. Thomsen and W. Boomsma. Multiple Sequence Alignment Using SAGA: Investigating the effects of operator scheduling, population seeding, and crossover operators. In *Applications of Evolutionary Computing*, volume 3005 of *Lecture Notes in Computer Science*, pages 113–122. Springer, 2004.

[47] M. Vingron and P. Argos. A fast and sensitive multiple sequence alignment algorithm. *Comput. Appl. Biosci.*, 5(2):115–121, April 1989.

[48] M. Vingron and M. Waterman. Sequence alignment and penalty choice: Review of concepts, case studies and implications. *Journal of Molecular Biology*, 235(1):1–12, January 1994.

[49] L. Wang and T. Jiang. On the complexity of multiple sequence alignment. *Journal of Computational Biology*, 1(4):337–348, 1994.

[50] J. D. Watson and F. H. C. Crick. Molecular structure of nucleic acids: A structure for deoxyribose nucleic acid. *Nature*, 171:737–738, 1953.

[51] H. Zang, S. Zhang, and K. Hapeshi. A review of nature-inspired algorithms. *Journal of Bionic Engineering*, 7(Supplement 1):S232 – S237, 2010.

[52] C. Zhang and A. K. C. Wong. A genetic algorithm for multiple molecular sequence alignment. *Computer applications in the biosciences : CABIOS*, 13(6):565–581, 1997.