Universidade Federal do Amazonas
Instituto de Ciências Exatas
Departamento de Ciência da Computação
Programa de Pós-Graduação em Informática

Alinhamento Múltiplo de Seqüências Através de Técnicas de Agrupamento

Patrícia Silva Peres

Manaus – Amazonas
Fevereiro de 2006

Patrícia Silva Peres

Alinhamento Múltiplo de Seqüências Através de Técnicas de Agrupamento

Dissertação apresentada ao Programa de Pós-Graduação em Informática do Departamento de Ciência da Computação da Universidade Federal do Amazonas, como requisito parcial para obtenção do Título de Mestre em Informática. Área de concentração: Recuperação de Informação.

Orientador: Prof. Dr. Edleno Silva de Moura

Patrícia Silva Peres

Alinhamento Múltiplo de Seqüências Através de Técnicas de Agrupamento

Dissertação apresentada ao Programa de Pós-Graduação em Informática do Departamento de Ciência da Computação da Universidade Federal do Amazonas, como requisito parcial para obtenção do Título de Mestre em Informática. Área de concentração: Recuperação de Informação.

Banca Examinadora

Prof. Dr. Edleno Silva de Moura – Orientador
Departamento de Ciência da Computação – UFAM

Prof. Nívio Ziviani, Ph.D.
Departamento de Ciência da Computação – UFMG

Prof. João Marcos Bastos Cavalcanti, Ph.D.
Departamento de Ciência da Computação – UFAM

Manaus – Amazonas
Fevereiro de 2006

Dedico este trabalho à minha mãe, Vera Alice.

Agradecimentos

Agradeço a Deus, acima de tudo. À minha mãe que sempre me apoiou em todos os momentos. Ao meu orientador, Edleno Moura, por ter acreditado na minha capacidade. Ao meu namorado, Allan e aos meus amigos Célia Santos, Márcio Vidal e Ruan Belém, companheiros de todas as horas. Ao PPGI, pela oportunidade. E a todos aqueles que ajudaram direta ou indiretamente na realização deste trabalho, o meu mais profundo agradecimento.

"Somos o que repetidamente fazemos; a excelência, portanto, não é um feito, mas sim um hábito."

Aristóteles

Resumo

O alinhamento simultâneo entre várias seqüências de DNA ou proteína é um dos principais problemas em biologia molecular computacional. Alinhamentos múltiplos são importantes em muitas aplicações, tais como, predição da estrutura de novas seqüências, demonstração do relacionamento entre novas seqüências e famílias de seqüências já existentes, inferência da história evolutiva de uma família de seqüências, descobrimento de padrões que sejam compartilhados entre seqüências, montagem de fragmentos de DNA, entre outras.

Atualmente, a estratégia mais popular utilizada na resolução do problema do alinhamento múltiplo é o alinhamento progressivo. Cada etapa desta estratégia pode gerar uma taxa de erro que tenderá a ser baixa no caso de seqüências muito similares entre si, porém tenderá a ser alta na medida em que as seqüências divergirem. Portanto, a determinação da ordem de alinhamento das seqüências constitui-se em um passo fundamental na estratégia de alinhamento progressivo.

Estratégias tradicionais levam em consideração, a cada iteração do alinhamento progressivo, apenas o par ou grupo de seqüências mais próximo a ser alinhado. Tal estratégia minimiza a taxa de erro introduzida em cada etapa, porém pode não ser a melhor forma para minimizar a taxa de erro final.

Baseado nesta hipótese, este trabalho tem por objetivo o estudo e aplicação de uma técnica de agrupamento global para executar uma análise prévia de todas as seqüências de forma a separá-las em grupos de acordo com suas similaridades. Estes grupos, então, guiarão o alinhamento progressivo tradicional, numa tentativa de minimizar a taxa de erro global introduzida pelas etapas do alinhamento progressivo e melhorar o resultado final.

Para avaliar a confiabilidade desta nova estratégia, três métodos conhecidos foram modificados com o objetivo de agregar a nova etapa de agrupamento de seqüências. A acurácia das novas versões dos métodos foi testada utilizando três diferentes coleções de referências. Além disso, os métodos modificados foram comparadas com suas respectivas versões originais.

Os resultados dos experimentos mostram que as novas versões dos métodos com a etapa de agrupamento global realmente obtiveram alinhamentos melhores do que suas versões originais nas três coleções de referência e alcançando melhorias sobre os principais métodos encontrados na literatura, com um aumento de apenas 3% em média no tempo de execução.

Palavras-chave: Alinhamento Múltiplo de Seqüências, Estratégia de Alinhamento Progressivo, Técnicas de Agrupamento.

Abstract

The simultaneous alignment of many DNA or protein sequences is one of the commonest tasks in computational molecular biology. Multiple alignments are important in many applications, such as, predicting the structure of new sequences, demonstrating the relationship between new sequences and existing families of sequences, inferring the evolutionary history of a family of sequences, finding the characteristic motifs (core blocks) between biological sequences, assembling fragments in DNA sequencing, and many others.

Currently, the most popular strategy used for solving the multiple sequence alignment problem is the progressive alignment. Each step of this strategy might generate an error which is expected to be low for closely related sequences but increases as sequences diverge. Therefore, determining the order in which the sequences will be aligned is a key step in the progressive alignment strategy.

Traditional approaches take into account, in each iteration of the progressive alignment, only the closest pair or groups of sequences to be aligned. Such strategy minimizes the error introduced in each step, but may not be the best option to minimize the final error.

Based on that hypothesis, this work aims the study and the application of a global clustering technique to perform a previous analysis of all sequences in order to separate them into groups according to their similarities. These groups, then, guide the traditional progressive alignment, as an attempt to minimize the overall error introduced by the steps of the progressive alignment and improve the final result.

To assess the reliability of this new strategy, three well-known methods were modified for the purpose of introducing the new sequence clustering stage. The accuracy of new versions of the methods was tested using three different reference collections. Besides, the modified methods were compared with their original versions.

Results of the conducted experiments depict that the new versions of the methods with the global clustering stage really obtained better alignments than their original versions in the three reference collections and achieving improvement over the main methods found in literature, with an increase of only 3% on average in the running time.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

O alinhamento simultâneo entre várias seqüências de DNA ou proteína é um dos principais problemas em biologia computacional molecular. Alinhamentos múltiplos são importantes em muitas aplicações, tais como, predição de estruturas secundária e terciária de novas seqüências, demonstração do relacionamento entre novas seqüências e famílias de seqüências já existentes, inferência da história evolutiva de uma família de seqüências, descobrimento de padrões que sejam compartilhados entre seqüências, montagem de fragmentos de DNA, entre outras.

As técnicas de alinhamento múltiplo são geralmente aplicadas a seqüências de proteínas, pois tais tipos de seqüências fornecem mais informação sobre suas estruturas, as funções que as mesmas exercem em um organismo e a evolução entre seqüências. Como a determinação da estrutura e função é uma tarefa não-trivial mesmo para uma única proteína, a melhor forma de compreender as características de um grupo de seqüências é encontrar o relacionamento existente entre estas e outras proteínas com propriedades já conhecidas. Identificar que uma seqüência pertence a uma família, e alinhá-la com os demais membros, em geral permite inferências sobre suas características estruturais, funcionais e evolutivas [6, 11].

O maior desafio em gerar automaticamente um alinhamento é a dificuldade de definir exatamente qual é o melhor alinhamento múltiplo, e é impossível definir um padrão para um único alinhamento múltiplo correto. Em teoria, existe um alinhamento evolutivo correto a ser gerado a partir de qualquer grupo de seqüências. Entretanto, as diferenças entre as seqüências podem ser tão grandes em algumas partes de um alinhamento que não há uma solução única evidente a ser encontrada [10].

Apesar de não existir um padrão para se definir o melhor alinhamento múltiplo, existem métodos para pontuar alinhamentos na tentativa de mensurar a qualidade dos mesmos. Entretanto, também não existe nenhum método de pontuação que seja utilizado como padrão. Na verdade, existem diversos algoritmos de alinhamento que não utilizam sistemas de pontuação para qualificar os resultados gerados. A qualidade dos alinhamentos é medida através do significado biológico dos mesmos, sendo de máxima importância a experiência do avaliador [11, 22].

Assim como existe um método de Programação Dinâmica para alinhar otimamente duas seqüências conforme um sistema de pontuação [17, 24], também existe a possibilidade de adaptar este método para alinhar múltiplas seqüências, porém o mesmo se torna impraticável, uma vez que é necessário um tempo de CPU proporcional a n^k, onde n é o tamanho médio das seqüências e k o número de seqüências a serem alinhadas [11].

Algumas heurísticas para alinhar múltiplas seqüências foram desenvolvidas nos últimos anos com a finalidade de realizar comparações mais rapidamente, mesmo sem a precisão dos algoritmos de Programação Dinâmica. Entretanto, é importante ressaltar que ainda que estes métodos heurísticos forneçam com sucesso alinhamentos ótimos, dada uma função-objetivo, resta o problema da possibilidade do alinhamento ótimo não corresponder ao biologicamente correto.

Os métodos heurísticos mais populares são baseados na abordagem de alinhamento progressivo, onde primeiramente são alinhadas as seqüências mais fortemente relacionadas e depois as seqüências mais distantes são gradativamente adicionadas ao alinhamento. A técnica do alinhamento progressivo é baseada em uma abordagem gulosa e o erro introduzido uma vez durante o alinhamento não pode ser corrigido subseqüentemente. Tal erro tenderá a ser baixo no caso do alinhamento de seqüências muito similares entre si, porém tenderá a ser alto na medida em que as seqüências divergirem [9]. Por este motivo, a determinação da ordem de alinhamento das seqüências constitui-se em um passo fundamental no processo de resolução do alinhamento múltiplo, onde o ideal é que as seqüências mais similares sejam alinhadas primeiramente para que haja uma baixa taxa de erro no início do processo.

Baseado nesta hipótese, este trabalho tem por objetivo o estudo e aplicação de técnicas de agrupamento no auxílio à determinação da ordem em que as seqüências serão alinhadas, com o propósito de obter alinhamentos melhores ao final do processo.

1.1 Trabalhos relacionados

Até então muitos métodos para a resolução do alinhamento múltiplo de seqüências já foram propostos. A grande maioria consiste em heurísticas baseadas na abordagem de alinhamento progressivo, principalmente a abordagem proposta por Feng e Doolittle [9]. Esta consiste na construção de uma árvore-guia utilizada para estimar a ordem do alinhamento, onde primeiramente as seqüências mais relacionadas são alinhadas e depois as mais distantes são adicionadas gradativamente ao alinhamento.

Exemplos de métodos baseados na estratégia de alinhamento progressivo proposta por Feng e Doolittle são: Muscle [7, 8], MAFFT [13, 14], T-Coffee [19] e ClustalW [26]. Tais métodos e suas principais características são apresentados a seguir.

ClustalW foi proposto em 1994 por Julie D. Thompson et al. [26] e até hoje continua sendo amplamente utilizado na resolução do alinhamento múltiplo. De modo geral, seu funcionamento constitui-se nos seguintes passos: (i) todos os pares de seqüências são alinhados separadamente; (ii) uma matriz de distâncias é calculada a partir dos pares de seqüências alinhados; (iii) uma árvore-guia é construída a partir da matriz de distâncias utilizando o método *Neighbor-Joining* [21]; (iv) as seqüências são progressivamente alinhadas conforme a árvore-guia. Além disso, nesta última etapa de alinhamento progressivo são utilizadas características que se constituem no diferencial do método. Tais características são: utilização de pesos para as seqüências, penalidades de *gaps*[1] baseadas em posições específicas e escolha de matrizes de pesos, conhecidas como matrizes de substituição ou pontuação.

T-Coffee (*Tree-based Consistency Objective Function For alignmEnt Evaluation*) foi proposto em 2000 por Cédric Notredame et al. [19] e é muito utilizado quando há necessidade de se alinhar seqüências com pouca similaridade entre si, fazendo com que o mesmo se torne mais lento do que os demais métodos existentes. Seu funcionamento consiste basicamente nas seguintes etapas: (i) todos os pares de seqüências são alinhados separadamente gerando uma biblioteca primária de alinhamentos; (ii) extensão da biblioteca primária gerando uma nova biblioteca estendida; (iii) uma matriz de distâncias é calculada a partir dos alinhamentos contidos na biblioteca estendida; (iv) uma árvore-guia é construída a partir da matriz de distâncias utilizando o método *Neighbor-Joining* [21]; (v) as seqüências são progressivamente alinhadas

[1] Um *gap* consiste em uma subseqüência contínua de espaços "-" no alinhamento.

conforme a árvore-guia. O diferencial deste método corresponde à geração de uma biblioteca onde o alinhamento entre cada par de seqüências leva em consideração não apenas as duas seqüências em questão, mas todas as demais seqüências. Isto faz com que o alinhamento gerado para um par de seqüências contenha características do inter-relacionamento deste com as seqüências restantes.

Muscle (*MUltiple Sequence Comparison by Log-Expectation*) foi proposto em 2004 por Robert C. Edgar [7] [8]. Apesar de não ser tão popular quanto o ClustalW, o Muscle obtém um dos melhores resultados encontrados na literatura, sendo superior ao ClustalW e ao T-Coffee, conforme experimentos utilizando a coleção de referência BAliBASE [1, 27, 28]. O Muscle possui como principal diferencial a execução de vários refinamentos iterativos na tentativa de melhorar o alinhamento final. As principais etapas deste método são: (i) uma matriz de distâncias é gerada a partir da distância *k-mer* [30] entre cada par de seqüências (não é necessária a utilização de alinhamentos nesta etapa); (ii) uma árvore-guia é construída a partir da matriz de distâncias utilizando o método UPGMA [25]; (iii) as seqüências são progressivamente alinhadas conforme a árvore-guia gerando um alinhamento múltiplo chamado de *draft*; (iv) uma nova matriz de distâncias é calculada a partir do alinhamento *draft*; (v) uma nova árvore-guia é construída a partir da matriz de distâncias utilizando novamente o método UPGMA; (vi) as seqüências são progressivamente alinhadas conforme a árvore-guia gerando um novo e melhorado alinhamento múltiplo; (vii) refinamentos iterativos são realizados baseados na bipartição das árvores-guia geradas pelos alinhamentos progressivos (nesta etapa é utilizada uma métrica de pontuação para comparar os alinhamentos gerados em cada iteração e retornar aquele de maior *score*); (viii) a etapa de refinamento anterior é repetida até que não seja mais possível melhorar o alinhamento múltiplo gerado, ou seja, a função-objetivo (métrica de pontuação) foi maximizada.

MAFFT (*Multiple sequence Alignment based on the Fast Fourier Transform*) foi proposto em 2002 por Kazutaka Katoh et al. [13, 14] e se baseia na transformada rápida de Fourier que permite a detecção eficiente de segmentos similares entre as seqüências. O MAFFT pode ser considerado, conforme a literatura, um dos métodos mais eficazes de geração de alinhamentos múltiplos. Além disso, possui etapas similares ao Muscle, que envolvem respectivamente a criação de um alinhamento múltiplo *draft* enfatizando a rapidez e não a qualidade do alinhamento, em seguida a geração de um novo e melhorado alinhamento múltiplo e, por último, a

execução iterativa de refinamentos no alinhamento até que a função-objetivo seja maximizada. A principal diferença entre os dois métodos está na construção dos alinhamentos progressivos. O MAFFT utiliza a transformada rápida de Fourier para construir alinhamentos entre pares ou grupos de seqüências.

1.2 Contribuições

O alinhamento múltiplo de seqüências é um dos principais problemas existentes na Biologia Molecular Computacional. Por este motivo, diversos métodos foram e continuam sendo desenvolvidos com o objetivo de melhorar cada vez mais a eficiência deste processo e a qualidade dos alinhamentos múltiplos gerados.

Sendo assim, este trabalho propõe a aplicação de uma técnica de agrupamento de seqüências em métodos de alinhamento múltiplo já existentes na tentativa de melhorar os seus resultados gerados sem comprometer a eficiência dos mesmos [20].

É importante notar que, a técnica de agrupamento aqui proposta tem como pré-requisito a determinação de um limiar de distância a ser utilizado como critério na etapa de agrupamento das seqüências. A determinação manual ou automática de bons limiares não é uma tarefa simples, pois a mesma depende de vários fatores, dentre os quais está a matriz de distâncias utilizada por cada método de alinhamento.

Entretanto, com a realização de experimentos preliminares utilizando uma base de treinamento, foi possível estabelecer estes limiares automaticamente, de acordo com o grupo de seqüências a serem alinhadas e sua respectiva matriz de distâncias gerada.

Como será visto no decorrer desta dissertação, principalmente no capítulo de experimentos, a estratégia aqui proposta atinge seus objetivos, ou seja, realmente melhora os alinhamentos finais gerados com um aumento ínfimo no tempo de execução dos métodos modificados.

1.3 Organização da dissertação

Esta dissertação está organizada em cinco capítulos. No capítulo 2 são apresentados os principais conceitos necessários à compreensão deste trabalho, como por exemplo, conceitos de alinhamentos múltiplos de seqüências e técnicas de agrupamento.

O capítulo 3 apresenta a proposta de utilizar uma técnica de agrupamento como forma de auxiliar a determinação da ordem de alinhamento das seqüências, através de uma análise global das similaridades entre as mesmas. Além disso, é apresentado o algoritmo de agrupamento proposto com suas principais características e como o mesmo pode ser inserido em métodos já existentes de alinhamento múltiplo de seqüências.

O capítulo 4 descreve detalhes sobre os experimentos realizados, tais como as coleções de referências utilizadas para testar a viabilidade da estratégia proposta, formas de avaliação dos resultados e a análise, através de tabelas e gráficos, de todos os resultados alcançados.

Finalmente, o capítulo 5 traz o resumo dos resultados e as conclusões obtidas, discute as contribuições do trabalho desenvolvido e apresenta sugestões de trabalhos futuros.

Capítulo 2

Conceitos Básicos

Neste capítulo são apresentados e discutidos alguns conceitos básicos necessários à compreensão deste trabalho, como por exemplo, o conceito de alinhamento múltiplo de seqüências e técnicas de agrupamento.

2.1 Alinhamento Múltiplo de Seqüências

Nas seções seguintes são apresentadas a definição do problema do alinhamento múltiplo de seqüências, uma alternativa para avaliar (pontuar) automaticamente um alinhamento múltiplo e, por último, a principal técnica de alinhamento múltiplo disponível na literatura, a qual serve de base para a grande maioria dos métodos existentes de alinhamento de seqüências.

2.1.1 O que é um Alinhamento Múltiplo de Seqüências

Para o melhor entendimento do problema do alinhamento múltiplo de seqüências, inicialmente, deve-se definir o que é um alinhamento entre apenas duas seqüências, uma vez que este corresponde ao passo base do alinhamento múltiplo.

As duas seqüências não precisam ter o mesmo tamanho. Assim, define-se o alinhamento entre duas seqüências como sendo a inserção de espaços ("-"), também conhecidos como *gaps*, em pontos arbitrários ao longo das duas seqüências de modo que estas fiquem com o mesmo comprimento ao final do processo. Além disso, não é permitido que um *gap* em uma das seqüências esteja alinhado com um *gap* na outra. *Gaps* podem ser inseridos inclusive no início ou no final das seqüências [22].

Um alinhamento múltiplo entre $k > 2$ seqüências é uma generalização natural do alinhamento entre duas seqüências. *Gaps* selecionados são inseridos ao longo de cada uma das k seqüências, fazendo com que ao final deste processo, todas tenham o mesmo tamanho l. O alinhamento múltiplo pode, então, ser visto como uma matriz de k linhas com l colunas cada, onde nenhuma coluna deverá ser constituída de apenas *gaps* [11, 22]. A Figura 2.1 abaixo mostra um exemplo contendo o alinhamento entre seqüências de proteína.

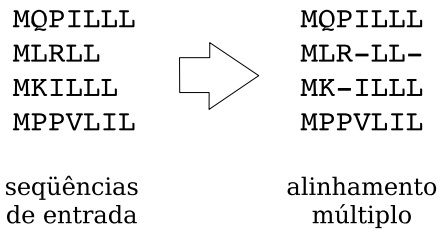

Figura 2.1: Exemplo de um alinhamento múltiplo entre seqüências de proteína.

Entretanto, esta é uma definição bastante simplista de alinhamento múltiplo. Não basta apenas inserir *gaps* aleatoriamente ao longo das seqüências. O importante é gerar um alinhamento múltiplo que possa expressar com um mínimo de clareza o relacionamento existente entre as seqüências alinhadas.

Com um conhecimento detalhado da bioquímica de uma seqüência, é possível criar um alinhamento manualmente. Contudo, este é um processo minucioso e cansativo. Por isso, a necessidade da geração de alinhamentos múltiplos automaticamente [10].

O maior desafio em gerar automaticamente um alinhamento é a dificuldade de definir exatamente qual é o melhor alinhamento múltiplo. Além disso, é impossível definir um padrão para um único alinhamento múltiplo de seqüências correto. Em teoria, existe um processo evolutivo subjacente e um alinhamento evolutivo correto a ser gerado a partir de qualquer grupo de seqüências. Entretanto, as diferenças entre as seqüências podem ser tão grandes em algumas partes de um alinhamento que não há uma solução única evidente a ser encontrada por algum algoritmo de alinhamento múltiplo [10].

2.1.2 Como pontuar um Alinhamento Múltiplo de Seqüências

Apesar de não existir um padrão para se definir o melhor alinhamento múltiplo, existem métodos para pontuar estes alinhamentos (utilizados como função-objetivo em algoritmos de alinhamento), como uma tentativa de mensurar a qualidade dos mesmos.

Contudo, também não existe nenhum método de pontuação que seja utilizado como padrão. Na verdade, existem diversos algoritmos de alinhamento que não utilizam nenhuma função-objetivo a ser otimizada. A qualidade dos resultados é medida através do significado biológico dos alinhamentos gerados, sendo de máxima importância a experiência do avaliador [11, 22].

Um modo idealizado de pontuar um alinhamento múltiplo poderia ser através da especificação de um modelo probabilístico completo da evolução das seqüências. Porém, infelizmente, não existem dados suficientes para parametrizar um modelo evolutivo complexo. Por esse motivo, simplificações devem ser feitas utilizando aproximações que parcial ou totalmente ignorem a árvore filogenética[1] correspondente a um grupo de seqüências [6].

Muitos métodos de pontuação assumem que as colunas de um alinhamento múltiplo são estatisticamente independentes umas das outras. Tal função de pontuação pode ser escrita como na Equação 2.1 seguinte [6]:

$$S(m) = G + \sum_i S(m_i) \tag{2.1}$$

Onde m_i é a i-ésima coluna do alinhamento múltiplo m, $S(m_i)$ é o *score* para a coluna i, e G é a função para pontuação de *gaps* que ocorrem no alinhamento. Utilizou-se G como uma função sem especificação porque os métodos para pontuação de *gaps* diferenciam fortemente entre si. O método mais simples é tratar um *gap* como um tipo de caracter a mais no alinhamento, resultando em uma equação simplificada, como mostra a Equação 2.2 seguinte [6]:

$$S(m) = \sum_i S(m_i) \tag{2.2}$$

Um dos principais métodos de pontuação de alinhamentos múltiplos de seqüências existente na literatura é o ***SP Score*** (*Score* Soma dos Pares, do inglês *Sum-of-Pairs Score*) [2].

O método da Soma dos Pares, ou simplesmente *SP Score* é amplamente utilizado devido a sua simplicidade e eficácia [22]. Tal método é definido como a soma das pontuações dois a dois dos caracteres presentes em uma coluna do alinhamento múltiplo, ou mais formalmente como é mostrado na Equação 2.3 seguinte [6].

[1] Uma árvore filogenética mostra a relação evolutiva entre os organismos com base nas suas características observáveis. Árvores filogenéticas podem ser construídas para populações, espécies, gêneros, etc, inclusive seqüências de DNA ou de proteínas [10, 22].

$$S(m_i) = \sum_{k<l} s(m_i^k, m_i^l) \tag{2.3}$$

Onde $s(m_i^k, m_i^l)$ corresponde ao *score* referente à coluna i do alinhamento múltiplo m entre uma seqüência k e uma seqüência l pertencentes a este alinhamento. Logo, o *score* total do alinhamento múltiplo corresponde à soma dos *scores* $S(m_i)$ de cada coluna do alinhamento.

Para o melhor entendimento deste método, considere o exemplo da Figura 2.1 constituído de um alinhamento múltiplo entre seqüências de proteína. Usando o *SP Score*, a pontuação da quarta coluna deste alinhamento é dada por:

$$S(m_4) = S(I, -, I, V) = s(I, -) + s(I, I) + s(I, V) + s(-, I) + s(-, V) + s(I, V)$$

Utilizando alguma matriz de pontuação[2] [5, 12] é possível definir os *scores* entre os pares de aminoácidos. Por exemplo, se a matriz Blosum 62 [12] for utilizada para calcular os *scores* da coluna acima e, além disso for definido o valor -10 para a penalização de *gaps*, teremos o seguinte *score*:

$$S(m_4) = S(I, -, I, V) = -10 + 4 + 3 - 10 - 10 + 3 = -20$$

Para os casos em que se tem um alinhamento de um *gap* com outro *gap*, ou seja, $s(-, -)$, defini-se este valor como sendo zero para que não influencie no resultado do *score* [22].

2.1.3 Técnicas existentes para a resolução do Alinhamento Múltiplo

Assim como existe um método de Programação Dinâmica para alinhar otimamente duas seqüências conforme um sistema de pontuação [17, 24], também existe a possibilidade de adaptar este método para alinhar múltiplas seqüências. Porém o mesmo se torna impraticável, uma vez que é necessário um tempo de CPU proporcional a n^k, onde n é o tamanho médio das seqüências e k o número de seqüências a serem alinhadas [11]. Existe ainda uma versão otimizada deste método proposta por Carrillo e Lipman [2] que reduz significativamente a complexidade de tempo e espaço do algoritmo. Mesmo assim, tal método consegue alinhar apenas entre cinco a sete seqüências de tamanho médio de 200 a 300 caracteres [6], não atendendo às necessidades atuais, ou seja, alinhamentos entre mais de 30 seqüências com tamanhos acima de 500 caracteres.

[2]Uma matriz de pontuação ou substituição corresponde a uma matriz 20x20 pré-definida contendo o *score* entre todos os pares de aminoácidos possíveis. Tal *score* reflete a possibilidade de um aminoácido ser substituído por outro.

Por este motivo, diversos métodos heurísticos de alinhamento múltiplo de seqüências têm sido desenvolvidos ultimamente com o propósito de realizar comparações mais rapidamente, sem perder qualidade nos resultados. A grande maioria destes métodos se baseia na estratégia do alinhamento progressivo e o restante na estratégia do alinhamento iterativo.

Estratégia de Alinhamento Iterativo. Primeiramente, um alinhamento múltiplo inicial é gerado, e baseado em algum mecanismo de avaliação, como por exemplo, o sistema de pontuação *SP Score*, o alinhamento múltiplo é melhorado iterativamente. O sistema de pontuação é, na verdade, a função-objetivo que deve ser otimizada, sendo de máxima importância neste processo. Se a função-objetivo for ruim, alinhamentos múltiplos ruins serão gerados. Um exemplo de método baseado nesta técnica é o SAGA [18].

Estratégia de Alinhamento Progressivo. A principal idéia do alinhamento progressivo é construir uma sucessão de alinhamentos entre pares de seqüências. Inicialmente, duas seqüências são escolhidas e alinhadas. Em seguida, uma terceira seqüência é escolhida e alinhada com o primeiro alinhamento, ou um novo par de seqüências é escolhido e alinhado. Este processo é repetido até que todas as seqüências tenham sido alinhadas [6].

Neste trabalho, porém, é utilizada somente a estratégia de alinhamento progressivo, sendo a mesma explicada com mais detalhes a seguir.

Como mencionado anteriormente, muitos métodos de resolução do alinhamento múltiplo se baseiam na estratégia do alinhamento progressivo, principalmente a estratégia proposta por Feng e Doolittle [9], na qual uma árvore-guia é construída com o objetivo de determinar a ordem de alinhamento das seqüências. As etapas básicas do alinhamento progressivo proposto por Feng e Doolittle são apresentadas nos tópicos seguintes e no diagrama da Figura 2.2.

- **Alinhamentos Par-a-Par/Matriz de Distâncias.** A primeira etapa da estratégia progressiva é a execução de alinhamentos par-a-par a partir das seqüências de entrada com o objetivo de gerar uma matriz de distâncias (etapa 2 da Figura 2.2). A partir de cada alinhamento par-a-par é calculado um valor de distância, o qual fornecerá o grau de divergência entre as seqüências, baseado em alguma métrica, como por exemplo a distância de Kimura [15], ou baseado apenas no valor de identidade[3] entre o par de seqüências alinhado.

[3] O valor de identidade corresponde à percentagem de nucleotídeos ou aminoácidos iguais na mesma posição no alinhamento entre duas seqüências.

- **Determinação da Árvore-Guia.** Uma árvore é construída a partir da matriz de distâncias para determinar a ordem na qual as seqüências serão progressivamente alinhadas (etapa 3 da Figura 2.2), por isto o nome de árvore-guia. Para a construção desta árvore-guia é utilizado um método específico, tal como *Neighbor-Joining* [21] ou UPGMA [25]. A árvore-guia corresponde a uma árvore binária cujas folhas representam seqüências e os nós internos representam alinhamentos múltiplos parciais. O nó raiz representa o alinhamento múltiplo completo. Os nós mais distantes da raiz representam os pares de seqüências mais similares entre si [6].

- **Alinhamento Progressivo.** Agora, o alinhamento múltiplo de seqüências pode ser construído seguindo a ordem estimada pela árvore-guia. As seqüências mais fortemente relacionadas são alinhadas primeiro, então as próximas duas seqüências mais similares são alinhadas ou uma seqüência é adicionada ao alinhamento existente entre as duas primeiras seqüências, dependendo do que for sugerido pela árvore-guia. As próximas duas seqüências ou grupos de seqüências pré-alinhadas mais similares são sempre unidos. Este processo iterativo continua até que todas as seqüências tenham sido alinhadas, formando assim, o alinhamento múltiplo final (ver etapa 4 da Figura 2.2).

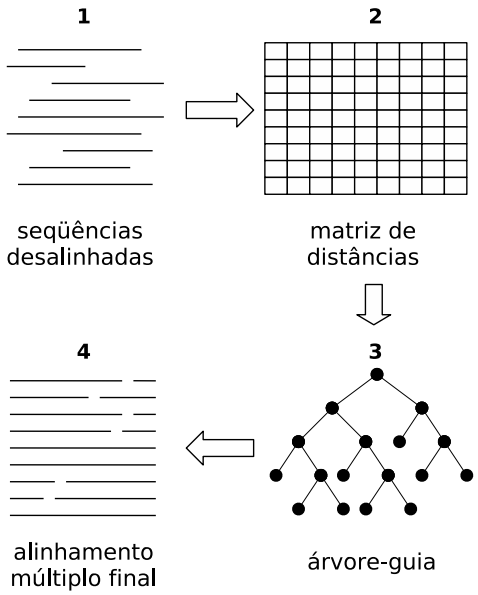

Figura 2.2: Diagrama para sumarizar as principais etapas do alinhamento múltiplo progressivo proposto por Feng e Doolittle. Estas etapas consistem na construção da matriz de distâncias, a determinação da árvore-guia e o alinhamento múlitplo progressivo.

A técnica do alinhamento progressivo é baseada em uma abordagem gulosa e o erro introduzido uma vez durante o alinhamento não pode ser corrigido subseqüentemente, ou seja, todos os *gaps* introduzidos durante os alinhamentos entre pares ou grupos de seqüências devem ser preservados até que se alcance o alinhamento múltiplo final. Esta propriedade é conhecida como "uma vez *gap*, sempre *gap*"(do inglês *"once a gap, always a gap"*) [9].

Cada passo no processo de alinhamento múltiplo pode gerar uma taxa de erro que tenderá a ser baixa no caso do alinhamento de seqüências muito similares entre si, porém tenderá a ser alta quando as seqüências forem divergentes [9].

Por este motivo, a determinação da ordem de alinhamento das seqüências constitui-se em um passo fundamental no processo de resolução do alinhamento múltiplo, onde o ideal é que as seqüências mais similares sejam alinhadas primeiro para que haja uma baixa taxa de erro no início do processo.

2.2 Técnicas de Agrupamento

As técnicas de agrupamento têm por principal objetivo criar grupos ou *clusters* contendo objetos que sejam de alguma forma similares em características. A métrica de similaridade adotada pode variar de método para método.

Freqüentemente, as técnicas de agrupamento são confundidas com técnicas de classificação, porém existem algumas diferenças entre estas duas. Na classificação os objetos são assinalados a classes pré-definidas, enquanto que no agrupamento, não existem classes pré-definidas, as mesmas são criadas no decorrer do processo.

Existem diversas técnicas de agrupamento disponíveis na literatura, porém, nos tópicos seguintes são apresentados apenas alguns exemplos destas técnicas.

K-means. O *k-means* é um dos mais populares métodos de agrupamento, onde o k representa o número de grupos desejados a serem criados. Tal método é considerado uma técnica de agrupamento *top-down* e funciona da seguinte maneira: (i) centróides representativos de cada grupo são inicializados; (ii) cada objeto é assinalado ao grupo cujo centróide seja mais similar; (iii) para cada grupo é recomputado um novo centróide baseado em todos os objetos adicionados ao grupo; (iv) os passos (ii) e (iii) são repetidos até que não seja mais possível nenhuma melhoria dentro dos grupos [4].

Agrupamento Aglomerativo. Também conhecido como agrupamento *bottom-up*, o agrupamento aglomerativo funciona da seguinte maneira: (i) cada objeto é considerado como um grupo unitário formando um conjunto inicial de grupos; (ii) dois grupos Γ e Δ são selecionados de acordo com alguma métrica de similaridade; (iii) os grupos Γ e Δ são removidos do conjunto de grupos; (iv) os grupos Γ e Δ são unidos formando um novo grupo Φ sendo o mesmo adicionado ao conjunto de grupos; os passos (ii) a (iv) são repetidos até que não seja possível unir mais grupos [4].

Agrupamento de Associação Cruzada. A técnica de agrupamento de associação cruzada (do inglês *cross-association clustering*) funciona, em linhas gerais, da seguinte maneira: dada uma matriz binária, automaticamente o algoritmo determina grupos homogêneos de objetos através do rearranjo da matriz em grupos disjuntos de linhas e colunas [3]. É importante ressaltar que se este algoritmo for adaptado para agrupar seqüências, o mesmo não levará em consideração a similaridade entre as mesmas ao realizar o agrupamento, uma vez que o algoritmo recebe como entrada uma matriz binária e não utiliza nenhuma outra métrica de similaridade/distância ao agrupar seqüências. Como a similaridade é uma importante característica para encontrar relacionamentos entre seqüências, é interessante que a mesma não seja ignorada.

Entropia Mínima. O objetivo do método de entropia mínima consiste em minimizar a entropia de cada grupo, onde o cálculo da entropia se baseia na proposta de Shannon [23]. O algoritmo de agrupamento baseado na entropia mínima possui os seguintes passos: (i) primeiramente, faz-se necessário a utilização de outro método de agrupamento para formar uma partição inicial dos objetos; (ii) para cada objeto x, se o grupo c_j contendo a maioria dos vizinhos (objetos mais similares a x) for diferente do grupo c_i ao qual x pertence então calcula-se a entropia do grupo c_j sem o objeto x e a entropia do grupo c_j com x; (iii) se a entropia diminuir, o objeto x é retirado do grupo c_i e assinalado ao grupo c_j; (iv) repete-se os passos (ii) e (iii) até que os grupos não mudem mais [16].

Capítulo 3

Alinhamento Múltiplo com Agrupamento Global

Como foi dito anteriormente, os métodos heurísticos mais populares são baseados na abordagem de alinhamento progressivo proposta por Feng e Dolittle [9]. Tal técnica é baseada em uma abordagem gulosa e o erro introduzido uma vez durante o alinhamento não pode ser corrigido subseqüentemente. Cada passo no processo de alinhamento múltiplo pode gerar uma taxa de erro que tenderá a ser baixa no caso do alinhamento de seqüências muito similares entre si, porém tenderá a ser alta quando as seqüências divergirem [9].

Sendo assim, a determinação da ordem de alinhamento das seqüências é passo fundamental na resolução do alinhamento múltiplo.

Baseado nesta hipótese, este trabalho tem por objetivo o estudo e aplicação de uma técnica de agrupamento no auxílio à determinação da ordem de alinhamento das seqüências, através de uma análise global das similaridades entre as mesmas (por isso o nome de agrupamento global), com o propósito de que alinhamentos melhores sejam obtidos ao final do processo.

A seguir são apresentados uma definição mais detalhada de agrupamento global, o motivo pelo qual é proposto neste trabalho a utilização de uma técnica de agrupamento global em métodos de alinhamento múltiplo, a estratégia proposta de agrupamento global e como a mesma pode ser inserida em métodos existentes de alinhamento múltiplo de seqüências.

3.1 Conceito de agrupamento local e global

Neste trabalho, as técnicas de agrupamento foram classificadas como locais ou globais, dependendo da decisão tomada no momento de agrupamento dos objetos.

Se um determinado método de agrupamento se baseia em uma estratégia gulosa para agrupar objetos, ou seja, a cada iteração apenas o par de objetos mais próximo é agrupado desconsiderando os demais, tal método é dito de **agrupamento local**, pois o mesmo avalia o ganho obtido a cada passo e não no processo de alinhamento como um todo.

O processo de construção da árvore-guia mostrado no capítulo anterior pode ser considerado como uma técnica de agrupamento, uma vez que o alinhamento múltiplo é gerado através do agrupamento de pares de seqüências ou grupos de acordo com a ordem estabelecida pela árvore-guia. Além disso, tal processo pode ser ainda considerado como de agrupamento local, pois o mesmo leva em consideração em cada iteração apenas o par de seqüências ou grupos mais similares entre si, desconsiderando as demais seqüências.

O conceito de **agrupamento global** proposto neste trabalho não é baseado em uma estratégia gulosa, como no caso do agrupamento local. Ao invés de considerar apenas o par de objetos mais próximo a cada iteração (tomada de decisões baseada na melhor alternativa a cada passo), no agrupamento global são levados em consideração um grupo de objetos mais similares entre si, com o objetivo de tomar decisões a cada iteração que tentem otimizar o alinhamento múltiplo como um todo.

3.2 O porquê da utilização do agrupamento global

A estratégia do alinhamento progressivo baseia-se na hipótese de que alinhamentos melhores são obtidos se seqüências mais similares são alinhadas primeiro. Para estimar a ordem em que as seqüências serão alinhadas é amplamente utilizada uma árvore-guia, sendo esta ordem de máxima importância para que bons alinhamentos múltiplos sejam gerados.

Conforme explicado anteriormente, a árvore-guia utilizada pela estratégia progressiva pode ser considerada como uma técnica de agrupamento local, pois o processo de construção da mesma leva em consideração a cada iteração apenas o par de seqüências ou grupos mais similares entre si, desconsiderando as demais seqüências.

Este comportamento pode causar perda de informação importante sobre o relacionamento entre o par de seqüências ou grupos alinhado e as seqüências restantes. Tal informação pode ser bastante útil no processo de alinhamento como um todo. Devido à propriedade de agrupamento local da árvore-guia, a ordem estabelecida pela mesma pode, em algum momento, gerar um alinhamento ruim entre duas seqüências devido à prioridade estabelecida pela árvore.

Como a ordem de alinhamento das seqüências possui grande impacto na geração de bons resultados e a árvore-guia por si só toma decisões locais de agrupamento das seqüências, é proposta a utilização de uma técnica de agrupamento global de seqüências como uma etapa preliminar para auxiliar na determinação desta ordem. O objetivo é que sejam gerados grupos contendo seqüências altamente similares entre si para posterior alinhamento.

Além disso, é importante ressaltar que o agrupamento global toma decisões a cada passo visando a otimização do resultado final, ao invés do agrupamento local que toma decisões com base na melhor solução a cada passo. O que é melhor localmente, nem sempre gera o melhor resultado no final.

A vantagem de se utilizar uma técnica de agrupamento global ao invés da simples utilização de uma árvore-guia para a geração de um alinhamento múltiplo é a possibilidade de se testar uma forma diferente para se estimar a ordem em que as seqüências serão alinhadas. A utilização de boas técnicas de agrupamento força a geração de grupos com seqüências muito similares entre si, fazendo com que, pelo menos dentro de cada grupo, o alinhamento gerado seja o melhor possível. A desvantagem, porém, é que esta estratégia acaba consumindo mais tempo que a simples construção da árvore-guia, devido à divisão prévia das seqüências em grupos.

3.3 Estratégia proposta de agrupamento global

O algoritmo de agrupamento global proposto neste trabalho utiliza como entrada uma matriz de distâncias e um limiar[1] que também consiste em um valor de distância. Este limiar irá estabelecer o grau de coesão dentro de cada grupo gerado. Quanto menor o limiar, maior será a coesão dentro de cada grupo. O algoritmo de agrupamento de seqüências é dividido em duas etapas: geração de uma lista de candidatos a grupos e, geração da lista final de grupos. Estas etapas são explicadas conforme os algoritmos das Figuras 3.1, 3.2 e 3.5.

[1] Detalhes sobre a determinação do limiar são dados posteriormente no Capítulo 4, seção 4.3.

O algoritmo apresentado na Figura 3.1 tem o propósito de criar uma lista de candidatos a grupos de seqüências a partir da matriz de distâncias. Cada linha da matriz representa o relacionamento entre a sua seqüência correspondente (chamada de seqüência representativa do grupo) e as demais seqüências. O algoritmo, então, gera um grupo que possui somente seqüências com distância menor ou igual ao limiar L quando alinhadas com a seqüência representativa do grupo. Logo após, é feita uma validação do grupo utilizando o método $ValidaGrupoCandidato(grupo, L)$, apresentado na Figura 3.2.

```
Algoritmo GeraListaGruposCandidatos
início
    Entrada: Um limiar L e uma matriz de distâncias especificada pela variável
             MatrizDistancias.
    Saída: Uma lista de candidatos a grupos especificada pela variável ListaGruposCandidatos.

    para cada linha i de MatrizDistancias faça
        Inicializa um novo grupo i;
        para cada coluna j de MatrizDistancias faça
            se MatrizDistancias[i, j] ≤ L então
                Adiciona a seqüência j ao grupo i;
            fim
        fim
        novoGrupo ← ValidaGrupoCandidato(grupo, L);
        Adiciona novoGrupo em ListaGruposCandidatos;
    fim
fim
```

Figura 3.1: Algoritmo para criar uma lista de candidatos a grupos de seqüências.

A validação consiste em retirar do grupo candidato seqüências que ao serem comparadas com as demais possuam valor de distância maior que a do limiar estabelecido por L. Portanto, primeiramente, localiza-se o par de seqüências com o maior valor de distância no grupo candidato. Para decidir qual das duas seqüências deve ser retirada, verifica-se qual das duas seqüências tem a menor contribuição para o grupo. A contribuição de uma seqüência é calculada como a soma das distâncias entre a mesma e as demais seqüências do grupo, conforme a equação abaixo:

$$DistanciaTotal(sequencia, grupo) = \sum_{s \in grupo} MatrizDistancias[sequencia, s] \tag{3.1}$$

A seqüência que obtiver a maior soma das distâncias é, então, a que menos contribui para o grupo candidato e pode ser retirada do mesmo.

```
Algoritmo ValidaGrupoCandidato
início
    Entrada: Um grupo candidato de seqüências especificado pela variável grupo e um limiar L.
    Saída: Um novo grupo validado de seqüências especificado pela variável grupo.

    grupoOK ← falso;
    enquanto grupoOK = falso faça
        Encontra par de seqüências s e t com a maior distância maiorDist no grupo;
        se maiorDist > L então
            se seqüência s contribuir menos para grupo então
                Remove a seqüência s de grupo;
            fim
            senão
                Remove a seqüência t de grupo;
            fim
        fim
        senão
            grupoOK ← verdadeiro;
        fim
    fim
fim
```

Figura 3.2: Algoritmo para validar um grupo candidato, fazendo com que o mesmo não tenha nenhum par de seqüências com uma distância maior que a do limiar L.

O processo de validação continua até que não haja mais nenhum par de seqüências no grupo candidato com uma distância maior que a do limiar estabelecido L.

Após o processo de validação, o novo grupo candidato gerado é adicionado em *ListaGruposCandidatos*, caso o mesmo ainda não exista na lista (ver Figura 3.1). Além disso, a lista de grupos *ListaGruposCandidatos* é ordenada de forma decrescente de acordo com a similaridade total de cada grupo. A função de similaridade total de um grupo é dada pela Equação 3.2 seguinte:

$$SimilaridadeTotalGrupo(grupo) = \sum_{s \in grupo} \sum_{t \in grupo} MAX - MatrizDistancias[s,t] \qquad (3.2)$$

Onde MAX representa o maior valor de distância que possa existir entre duas seqüências. Se os valores de distância variam entre 0 e 1, então $MAX = 1$. Além disso, se o grupo for unitário, a similaridade total do mesmo é estabelecida como 0, para que grupos que contenham mais de uma seqüência tenham prioridade sobre grupos unitários.

Para ilustrar o funcionamento desta primeira etapa de criação de uma lista de candidatos a grupos, é mostrado a seguir um exemplo baseado na matriz de distâncias da Figura 3.3 formada por seis seqüências A, B, C, D, E e F.

	A	B	C	D	E	F
A	0	0,10	0,20	0,30	0,40	0,50
B	0,10	0	0,20	0,40	0,50	0,60
C	0,20	0,20	0	0,15	0,50	0,40
D	0,30	0,40	0,15	0	0,40	0,50
E	0,40	0,50	0,50	0,40	0	0,10
F	0,50	0,60	0,40	0,50	0,10	0

Figura 3.3: Exemplo de matriz de distâncias formada por seis seqüências A, B, C, D, E e F. Os valores de distância variam entre 0 e 1.

Suponha que o valor do limiar L seja estabelecido em $0,3$. Portanto, após computar a linha da seqüência A na matriz de distâncias da Figura 3.3, tem-se o grupo candidato resultante **ABCD**, contendo apenas as seqüências com distância menor ou igual a L quando comparadas com a seqüência A.

A Figura 3.4 mostra a matriz de distâncias apenas para as seqüências do grupo candidato **ABCD**. A última coluna da matriz representa a soma das distâncias referente a cada linha, que nada mais é do que a aplicação da Equação 3.1 para cada seqüência.

	A	B	C	D	
A	0	0,10	0,20	0,30	0,60
B	0,10	0	0,20	0,40	0,70
C	0,20	0,20	0	0,15	0,55
D	0,30	0,40	0,15	0	0,85

Figura 3.4: Matriz de distâncias formada a partir da matriz da Figura 3.3, porém contendo apenas as seqüências do grupo candidato ABCD.

O maior valor de distância da matriz é $0,4$ entre as seqüências B e D, sendo este valor maior que o limiar $0,3$ estabelecido. Como a seqüência D contribui menos para o grupo candidato, ou seja, a distância total entre D e o grupo é maior que a distância total entre B e o grupo, a seqüência D é removida, resultando no grupo candidato **ABC**.

Como este novo grupo candidato não possui nenhum par de seqüências com distância maior que o limiar L, então o mesmo é adicionado na lista de candidatos a grupos. Como o maior valor de distância que pode existir na matriz é 1, então a similaridade total do grupo candidato **ABC** é $2,5$, conforme a aplicação da Equação 3.2.

As linhas referentes às seqüências B e C da matriz de distâncias da Figura 3.3 geram o grupo candidato ABC, porém o mesmo já existe na lista de candidatos a grupos, logo não deve ser adicionado novamente.

Para a linha da seqüência D da matriz de distâncias da Figura 3.3, tem-se o grupo candidato resultante **ACD**. Como este grupo não possui nenhum par de seqüências com distância maior que o limiar L, então o mesmo é adicionado na lista de candidatos a grupos com uma similaridade total de $2,35$.

E, por último, as linhas referentes às seqüências E e F da matriz de distâncias da Figura 3.3 geram o grupo candidato **EF**, sendo o mesmo adicionado na lista de candidatos a grupos com uma similaridade total de $0,9$.

Portanto, a lista resultante de candidatos a grupos é mostrada abaixo, lembrando que a mesma é ordenada decrescentemente por similaridade total de cada grupo:

ABC - $2,50$

ACD - $2,35$

EF - $0,90$

A complexidade de tempo do processo de validação de um grupo candidato é $O(g^2)$, onde g corresponde ao tamanho do grupo, pois para encontrar o maior valor de distância no grupo, é necessário percorrer a matriz de distâncias do mesmo. Como o tamanho do grupo é no máximo o número de seqüências k, então a sua complexidade é $O(k^2)$.

Portanto, a complexidade de tempo desta primeira etapa de geração de grupos candidatos é $O(k^3)$, pois para cada grupo candidato encontrado (no máximo k) é necessário executar o processo de validação.

A segunda etapa é descrita pelo algoritmo da Figura 3.5 e corresponde à geração da lista final de grupos de seqüências através da remoção de conflitos entre os grupos candidatos da lista gerada na etapa anterior.

Conforme o exemplo apresentado anteriormente, a lista de grupos candidatos possui um conflito entre os grupos candidatos **ABC** e **ACD**, ou seja, é melhor agrupar **AC** com **B** ou com **D**? Por este motivo, a lista de candidatos a grupos é ordenada de forma decrescente de acordo com a similaridade total de cada grupo.

Algoritmo GeraListaGrupos
início
  Entrada: Uma lista de candidatos a grupos especificada pela variável *ListaGruposCandidatos*.
  Saída: Uma lista final de grupos especificada pela variável *ListaGrupos*.

  enquanto *ListaGruposCandidatos não é vazia* **faça**
    grupo ← *ListaGruposCandidatos*[0];
    Adiciona *grupo* em *ListaGrupos*;
    Remove de cada grupo candidato restante em *ListaGruposCandidatos* as seqüências de *grupo*;
    Reordena os grupos candidatos resultantes em *ListaGruposCandidatos*;
  fim
fim

Figura 3.5: Algoritmo para a criação da lista final de grupos a partir da lista de candidatos a grupos de seqüências.

Como o grupo candidato **ABC** possui uma similaridade total maior que o grupo candidato **ACD**, o mesmo será tranformado em um grupo definitivo e as seqüências A e C serão removidas do grupo candidato **ACD**, resultando em um novo grupo candidato contendo apenas a seqüência **D**. Sendo assim, a lista de candidatos a grupos deve ser reordenada, gerando a nova lista a seguir:

$$\text{EF - } 0,9$$

$$\text{D - } 0,0$$

Este processo continua até que não haja mais grupos em *ListaGruposCandidatos*. Logo, a lista de grupos definitivos gerada a partir da lista de candidatos a grupos mostrada anteriormente é: **ABC**, **EF** e **D**.

A complexidade de tempo desta segunda e última etapa de geração de grupos definitivos é $O(l^3)$, onde l corresponde ao tamanho da lista de candidatos a grupos, sendo que a operação de remover e reordenar têm complexidade de $O(l^2)$. Como este tamanho pode ser no máximo o número de seqüências de entrada k, então a complexidade desta etapa é $O(k^3)$. Portanto, o processo completo de agrupamento das seqüências possui uma complexidade de tempo de $O(k^3)$.

Como será visto posteriormente, o Muscle tem complexidade de tempo de $O(k^4 + kn^2)$ e o ClustalW, $O(k^4 + n^2)$, onde n corresponde ao tamanho médio das seqüências. O MAFFT tem uma complexidade de tempo similar ao Muscle, ou seja, maior que $O(k^3)$ [7]. Logo, adicionar o algoritmo de agrupamento proposto aos métodos de alinhamento não aumenta a complexidade de tempo total dos mesmos.

3.4 Aplicando o agrupamento global ao alinhamento múltiplo

Para o desenvolvimento, avaliação e validação da idéia de que a aplicação da técnica proposta de agrupamento global como uma forma de auxiliar na determinação da ordem em que as seqüências serão alinhadas pode realmente gerar alinhamentos múltiplos melhores, três métodos conhecidos na literatura foram modificados. O objetivo é aplicar a técnica de agrupamento proposta como uma nova etapa do processo de resolução do alinhamento múltiplo.

Os três métodos modificados foram: **Muscle**[2] versão 3.52 [7, 8], **MAFFT**[3] versão 5.667 [13, 14] e **ClustalW**[4] versão 1.83 [26]. É importante notar que estes métodos são todos baseados na estratégia de alinhamento progressivo. Além disso, seus códigos-fonte estão livremente disponíveis na Internet, sendo que o Muscle foi desenvolvido na linguagem C++ e o MAFFT e o ClustalW foram ambos desenvolvidos na linguagem C.

Os métodos Muscle e MAFFT foram escolhidos por obterem os melhores alinhamentos dentre os métodos existentes na literatura. Por outro lado, o método ClustalW foi escolhido pois continua sendo o método mais popular mesmo sem gerar os melhores alinhamentos.

As modificações realizadas no Muscle, MAFFT e ClustalW são apresentadas a seguir e as principais etapas do processo de alinhamento múltiplo incluindo a nova etapa de agrupamento são mostradas na Figura 3.6.

O processo de agrupamento global de seqüências proposto é baseado nas entradas de uma matriz de distâncias e seu algoritmo funciona conforme explicado na seção 3.3. Portanto, a nova etapa de agrupamento das seqüências foi adicionada depois do processo de construção da matriz de distâncias na estratégia de alinhamento progressivo, como é mostrado na Figura 3.6 (etapas em fundo cinza).

Para encontrar grupos de seqüências similares entre si, um limiar deve ser definido. Este limiar irá estabelecer o grau de coesão dentro de cada grupo gerado. Quanto menor o limiar, maior será o grau de coesão dentro dos grupos. A escolha do valor do limiar irá definir a qualidade dos grupos gerados e, conseqüentemente, a qualidade do alinhamento múltiplo final.

Como será visto no capítulo seguinte de experimentos, é proposto neste trabalho a determinação automática deste limiar através da utilização de uma base de treinamento.

[2] http://www.drive5.com/muscle/
[3] http://www.biophys.kyoto-u.ac.jp/∼katoh/programs/align/mafft/
[4] http://www.ebi.ac.uk/clustalw/

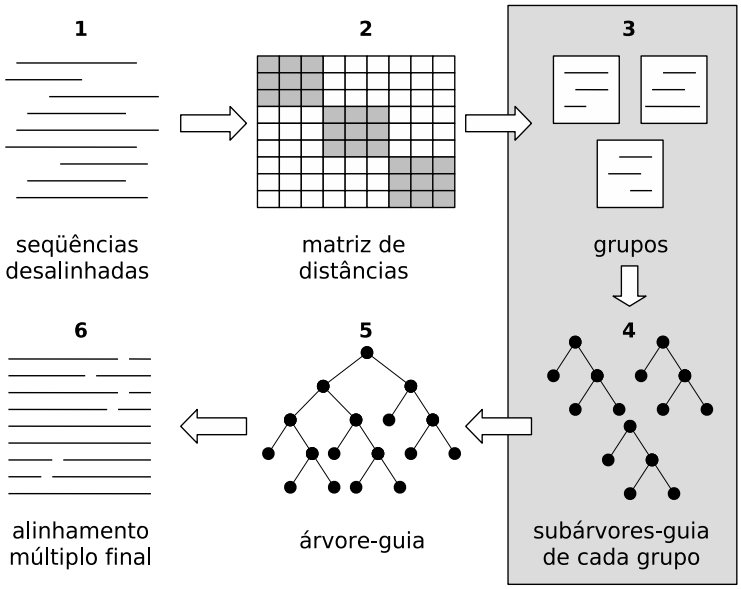

Figura 3.6: Diagrama para sumarizar as principais etapas do processo de alinhamento múltiplo usando a técnica de agrupamento global.

Depois de encontrar os grupos (ver passo 3 da Figura 3.6), deveria ser executado o alinhamento múltiplo dentro de cada grupo e então, o alinhamento múltiplo entre os grupos. Entretanto, para que não houvesse grandes modificações nos métodos originais, optou-se em gerar uma nova árvore-guia refletindo a ordem estabelecida pelo agrupamento, para que então o próprio método original criasse o alinhamento múltiplo final baseado nesta nova árvore.

A criação desta nova árvore-guia consiste em, primeiro, criar uma subárvore-guia para cada grupo (ver passo 4 da Figura 3.6) e então agrupar todas estas subárvores-guia gerando a nova árvore-guia final e completa (ver passo 5 da Figura 3.6), a qual será usada para criar o alinhamento múltiplo final.

Esta etapa foi implementada modificando-se o método utilizado para construir a árvore-guia, como por exemplo o *Neighbor-Joining* [21] ou o UPGMA [25].

A lista de grupos se torna um pré-requisito para a geração da árvore-guia, ou seja, ao invés de seguir diretamente a ordem de agrupamento estabelecida pela matriz de distâncias, utiliza-se a ordem estabelecida pelos grupos.

Primeiramente, cada grupo tem todas as suas seqüências agrupadas, como se existisse uma matriz de distâncias para cada grupo. A cada agrupamento de duas seqüências, a matriz de distâncias é atualizada conforme o método original de construção da árvore-guia.

Ao final do processamento de todos os grupos, a matriz de distâncias resultante terá exatamente como ordem o número de grupos. Em seguida, o algoritmo de construção da árvore-guia continua o seu processamento original, fazendo com que os grupos sejam agrupados gerando, então, a árvore-guia global.

Para ilustrar o processamento descrito anteriormente, suponha a matriz de distâncias da Figura 3.3 e os grupos gerados: **ABC**, **EF** e **D**.

Para o grupo ABC, o menor valor de distância é $0,1$ entre as seqüências A e B. Logo, estas duas seqüências devem ser agrupadas e a matriz de distâncias resultante, conforme o método UPGMA, é mostrada na Figura 3.7.

	AB	C	D	E	F
AB	0	0,20	0,35	0,45	0,55
C	0,20	0	0,15	0,50	0,40
D	0,35	0,15	0	0,40	0,50
E	0,45	0,50	0,40	0	0,10
F	0,55	0,40	0,50	0,10	0

Figura 3.7: Matriz de distâncias resultante após o agrupamento das seqüências A e B.

Agora, o menor valor de distância para este grupo é $0,20$ que corresponde ao agrupamento de AB e C. Note que, se o método original de construção da árvore-guia, neste caso o UPGMA, fosse executado, as próximas seqüências a serem agrupadas seriam E e F, por terem o menor valor de distância $0,10$.

Porém, nesta nova versão do método, cada grupo gerado tem prioridade no agrupamento. Portanto, primeiramente devem ser agrupadas todas as seqüências dentro de cada grupo, por isso o agrupamento entre AB e C.

A matriz de distâncias resultante é mostrada na Figura 3.8. Como não há outras seqüências a serem agrupadas dentro deste grupo, tem-se a sua respectiva subárvore-guia, que é mostrada na Figura 3.9.

	ABC	D	E	F
ABC	0	0,25	0,475	0,475
D	0,25	0	0,40	0,50
E	0,475	0,40	0	0,10
F	0,475	0,50	0,10	0

Figura 3.8: Matriz de distâncias resultante após o agrupamento entre AB e a seqüência C.

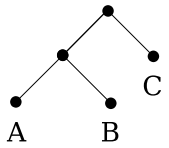

Figura 3.9: Árvore-guia resultante para o grupo ABC, utilizando como base a matriz de distâncias da Figura 3.3.

O próximo grupo a ser computado é EF. Como só existem duas seqüências a serem agrupadas, tem-se a subávore-guia mostrada na Figura 3.10. Após o agrupamento das seqüências E e F, tem-se a matriz resultante mostrada na Figura 3.11, que contém exatamente os valores de distância entre os grupos ABC, EF e D.

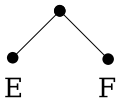

Figura 3.10: Árvore-guia resultante para o grupo EF, utilizando como base a matriz de distâncias da Figura 3.3.

Em seguida, o método de construção da árvore-guia continua sua execução original, que neste caso consiste em agrupar ABC e D, devido ao seu valor de distância $0,25$ ser o menor da matriz, gerando ABCD e, finalmente, agrupando ABCD com EF, gerando a árvore-guia final mostrada na Figura 3.12.

	ABC	D	EF
ABC	0	0,25	0,475
D	0,25	0	0,45
EF	0,475	0,45	0

Figura 3.11: Matriz de distâncias resultante após o agrupamento das seqüências E e F.

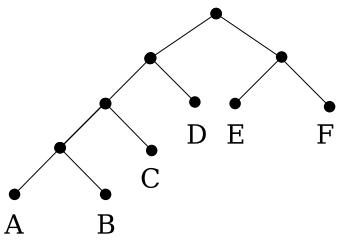

Figura 3.12: Árvore-guia resultante a partir da matriz de distâncias da Figura 3.3, utilizando como base os grupos ABC, EF e D.

Após a construção da árvore-guia baseada na ordem estabelecida pelos grupos, o método original (Muscle, MAFFT ou ClustalW) continua sua execução (no caso do Muscle e do MAFFT, a etapa de refinamento).

De acordo com [7], o algoritmo do Muscle tem complexidade de tempo de $O(k^4 + kn^2)$ e o algoritmo do ClustalW, $O(k^4 + n^2)$, onde n corresponde ao tamanho médio das seqüências. O MAFFT tem uma complexidade de tempo similar ao Muscle, ou seja, maior que $O(k^3)$.

É importante lembrar que em geral os alinhamentos são entre mais de 30 seqüências com tamanhos acima de 500 caracteres.

Como foi visto na seção 3.3, o algoritmo de agrupamento global proposto tem complexidade de $O(k^3)$. Portanto, adicionar o algoritmo de agrupamento proposto aos métodos de alinhamento não aumenta a complexidade de tempo total dos mesmos.

Além disso, para valores cada vez maiores de k, a diferença entre os tempos de execução dos métodos originais e suas versões modificadas tende a diminuir cada vez mais, devido ao fato do algoritmo de agrupamento proposto ter uma complexidade de tempo menor do que a dos algoritmos de alinhamento múltiplo aqui apresentados.

3.5 Utilização de outras estratégias de agrupamento

O algoritmo de agrupamento descrito na seção anterior foi desenvolvido com o propósito de criar grupos com um alto grau de coesão, usando para isto uma estratégia de agrupamento global. Existem outros algoritmos de agrupamento que podem ser considerados globais na literatura (ver Capítulo 2) e que poderiam ser aplicados ao invés do algoritmo proposto. Entretanto, decidiu-se não implementá-los, uma vez que o foco principal deste trabalho é apenas em avaliar e validar a idéia de aplicação de uma técnica de agrupamento global como forma de melhorar os alinhamentos resultantes. O estudo, aplicação e testes de outras técnicas de agrupamento global são deixados para trabalhos futuros.

Capítulo 4

Experimentos

Este capítulo apresenta experimentos realizados com o objetivo de avaliar e validar as novas versões dos métodos Muscle, MAFFT e ClustalW utilizando a etapa de agrupamento global de seqüências proposta no capítulo anterior. Além disso, são apresentadas as coleções de referência de alinhamentos múltiplos utilizadas, as formas de avaliar os resultados ao compará-los com os disponibilizados pelas coleções de referência, como os limiares para a etapa de agrupamento são determinados e, por último, são apresentados os resultados obtidos pelos métodos de alinhamento modificados e suas respectivas melhorias com relação aos métodos originais.

4.1 Coleções de Referências

Com o objetivo de testar a viabilidade e acurácia da nova estratégia aqui proposta, foram utilizadas três coleções de referências de alinhamentos múltiplos: BAliBASE [1, 27, 28], SABmark [29] e PREFAB [8]. Estas coleções de referências são apresentadas a seguir:

BAliBASE (*Benchmark Alignment dataBASE*) é uma coleção de referência constituída de alinhamentos de seqüências de proteínas refinados manualmente e categorizados em oito tipos de referência, onde cada uma caracteriza um problema real. A primeira referência é constituída de alinhamentos de seqüências de tamanho similar e filogeneticamente eqüidistantes entre si. Na segunda referência, cada alinhamento contém até três seqüências órfãs contidas em um grupo (família) de seqüências mais similares entre si. A terceira referência contém até quatro grupos de seqüências pouco relacionados entre si, enquanto que a quarta e a quinta referências envolvem longas inserções terminais e internas nos alinhamentos, respectivamente. A

sexta, sétima e oitava referências caracterizam problemas para os quais nenhum dos algoritmos testados foi designado, logo, tais referências não são utilizadas neste trabalho. As primeiras cinco referências são denotadas aqui como Ref1 – Ref5 e contêm 139 alinhamentos de referência, ressaltando que referências com menos de quatro seqüências foram descartadas uma vez que a etapa de agrupamento não influencia em nada nestes casos. Além disso, são disponibilizados os arquivos de anotação que marcam os *core blocks* onde as seqüências devem estar alinhadas. A versão 2 do BAliBASE[1] foi utilizada.

SABmark (*Sequence Alignment Benchmark*) foi designado para avaliar o desempenho de algoritmos de alinhamento de seqüências de proteínas usando seqüências com similaridade variando entre muito baixa e intermediária (0–50% identidade). Além disso, o SABmark permite avaliar casos onde nem todas as seqüências são similares umas com as outras. Atualmente, dois conjuntos de alinhamentos estão disponíveis, *Superfamily* e *Twilight*. O conjunto *Twilight* contém seqüências com similaridades muito baixas e o conjunto *Superfamily* contém seqüências com identidade entre pares ≤ 50 (similaridade intermediária). O conjunto *Twilight* não é usado neste trabalho devido à baixa similaridade entre as suas seqüências, logo a técnica de agrupamento não influenciaria nos resultados, uma vez que o principal objetivo é criar grupos com seqüências bastante similares entre si. O conjunto *Superfamily* contém 310 alinhamentos de referência com no máximo 25 seqüências cada, lembrando mais uma vez que referências com menos de quatro seqüências foram descartadas. A versão 1.65 do SABmark[2] foi utilizada.

PREFAB (*Protein REFerence Alignment Benchmark*) é uma coleção de referência onde duas seqüências de proteínas são alinhadas através de um método estrutural que não incorpora similaridades entre seqüências. Cada uma das duas seqüências é pesquisada em um banco de dados de seqüências. As seqüências mais similares com cada pesquisa são coletadas e combinadas (juntamente com as duas seqüências base) para formar conjuntos de ≤ 50 seqüências para serem, em seguida, alinhadas por um método de alinhamento múltiplo. O PREFAB contém ao todo 1666 alinhamentos de referência, ressaltando que referências com menos de quatro seqüências foram descartadas uma vez que a nova etapa de agrupamento não influencia em nada nestes casos. A versão 4 do PREFAB[3] foi utilizada.

[1] http://bips.u-strasbg.fr/en/Products/Databases/BAliBASE2/
[2] http://bioinformatics.vub.ac.be/databases/databases.html
[3] http://www.drive5.com/muscle/prefab.htm

4.2 Formas de Avaliação dos Resultados

Cada coleção de referência citada anteriormente utiliza um sistema de pontuação baseado no *SP Score* para computar a similaridade de um dado alinhamento comparado ao seu alinhamento de referência correspondente.

O BAliBASE utiliza dois tipos de pontuação, o *SP Score* (*Sum-of-Pairs Score*) e o *TC Score* (*Total Columm Score*).

O ***SP Score*** consiste na percentagem de pares de aminoácidos corretamente alinhados de acordo com o alinhamento de referência correspondente.

Suponha um alinhamento de teste com n seqüências consistindo de m colunas. A i-ésima coluna do alinhamento é especificado por $A_{i1}, A_{i2}, ..., A_{in}$. Para cada par de resíduos A_{ij} e A_{ik} define-se p_{ijk} tal que $p_{ijk} = 1$ se os resíduos A_{ij} e A_{ik} são alinhados um com o outro no alinhamento de referência, caso contrário $p_{ijk} = 0$. O *score* S_i para a i-ésima coluna é definido conforme a Equação 4.1 seguinte [28].

$$S_i = \sum_{j=1}^{n-1} \sum_{k=j+1}^{n} p_{ijk} \tag{4.1}$$

O *SP Score* é, então, definido conforme a Equação 4.2 abaixo, onde mr é o número de colunas do alinhamento de referência e S_{ri} é o *score* S_i para a i-ésima coluna do alinhamento de referência.

$$SP\ Score = \frac{\sum_{i=1}^{m} S_i}{\sum_{i=1}^{mr} S_{ri}} \tag{4.2}$$

Por sua vez, o ***TC Score*** consiste na percentagem de colunas alinhadas corretamente com relação ao alinhamento de referência correspondente. Para cada i-ésima coluna do alinhamento descrito acima, o *score* $C_i = 1$ se todos os resíduos da coluna estiverem alinhados no alinhamento de referência, caso contrário, $C_i = 0$. Portanto, o *TC Score* de um alinhamento é definido conforme a Equação 4.3 seguinte [28].

$$TC\ Score = \frac{\sum_{i=1}^{m} C_i}{m} \tag{4.3}$$

Para ilustrar o cálculo do *SP* e *TC Scores*, suponha o alinhamento de teste entre seqüências de proteína mostrado na Figura 4.1 (a) e o seu respectivo alinhamento de referência mostrado na Figura 4.1 (b).

MQPILLL	MQPILLL
MLR-LL-	ML-RL-L
MK-ILLL	MK-ILLL
MPPVLIL	MPPVLIL
alinhamento obtido	alinhamento de referência
(a)	**(b)**

Figura 4.1: (a) Alinhamento múltiplo obtido como teste; (b) Alinhamento múltiplo de referência.

Primeiramente, deve-se calcular o *SP Score* para o alinhamento de teste obtido. Para isto, deve-se calcular S_i conforme a Equação 4.1 para cada uma das sete colunas do alinhamento obtido. O cálculo detalhado de S_i é mostrado abaixo:

$$S_1 = (p_{112} + p_{113} + p_{114}) + (p_{123} + p_{124}) + (p_{134}) = (1 + 1 + 1) + (1 + 1) + (1) = 6$$
$$S_2 = (p_{212} + p_{213} + p_{214}) + (p_{223} + p_{224}) + (p_{234}) = (1 + 1 + 1) + (1 + 1) + (1) = 6$$
$$S_3 = (p_{312} + p_{313} + p_{314}) + (p_{323} + p_{324}) + (p_{334}) = (0 + 0 + 1) + (0 + 0) + (0) = 1$$
$$S_4 = (p_{412} + p_{413} + p_{414}) + (p_{423} + p_{424}) + (p_{434}) = (0 + 1 + 1) + (0 + 0) + (1) = 3$$
$$S_5 = (p_{512} + p_{513} + p_{514}) + (p_{523} + p_{524}) + (p_{534}) = (1 + 1 + 1) + (1 + 1) + (1) = 6$$
$$S_6 = (p_{612} + p_{613} + p_{614}) + (p_{623} + p_{624}) + (p_{634}) = (0 + 1 + 1) + (0 + 0) + (1) = 3$$
$$S_7 = (p_{712} + p_{713} + p_{714}) + (p_{723} + p_{724}) + (p_{734}) = (0 + 1 + 1) + (0 + 0) + (1) = 3$$

Agora, deve-se calcular S_i para cada uma das sete colunas do alinhamento de referência, ou seja, S_{ri}. O cálculo detalhado de S_{ri} é mostrado abaixo:

$$S_{r1} = (p_{112} + p_{113} + p_{114}) + (p_{123} + p_{124}) + (p_{134}) = (1 + 1 + 1) + (1 + 1) + (1) = 6$$
$$S_{r2} = (p_{212} + p_{213} + p_{214}) + (p_{223} + p_{224}) + (p_{234}) = (1 + 1 + 1) + (1 + 1) + (1) = 6$$
$$S_{r3} = (p_{312} + p_{313} + p_{314}) + (p_{323} + p_{324}) + (p_{334}) = (1 + 1 + 1) + (1 + 1) + (1) = 6$$
$$S_{r4} = (p_{412} + p_{413} + p_{414}) + (p_{423} + p_{424}) + (p_{434}) = (1 + 1 + 1) + (1 + 1) + (1) = 6$$
$$S_{r5} = (p_{512} + p_{513} + p_{514}) + (p_{523} + p_{524}) + (p_{534}) = (1 + 1 + 1) + (1 + 1) + (1) = 6$$
$$S_{r6} = (p_{612} + p_{613} + p_{614}) + (p_{623} + p_{624}) + (p_{634}) = (1 + 1 + 1) + (1 + 1) + (1) = 6$$
$$S_{r7} = (p_{712} + p_{713} + p_{714}) + (p_{723} + p_{724}) + (p_{734}) = (1 + 1 + 1) + (1 + 1) + (1) = 6$$

Após calculado o S_i e o S_{ri}, é possível calcular o *SP Score* conforme a Equação 4.2. O cálculo do *SP Score* para o alinhamento de teste da Figura 4.1 (a) utilizando como base o alinhamento de referência da Figura 4.1 (b) é mostrado a seguir:

$$SP\ Score = \frac{6+6+1+3+6+3+3}{6+6+6+6+6+6+6} = \frac{28}{42} = 0,667$$

O cálculo do *TC Score* é mais simples e consiste apenas em comparar as colunas do alinhamento de teste obtido e o alinhamento de referência, conforme a Equação 4.3. Neste caso, tem-se que o alinhamento de teste possui apenas três colunas exatamente iguais no alinhamento de referência. Portanto, o *TC Score* é conforme abaixo:

$$TC\ Score = \frac{1+1+0+0+1+0+0}{7} = \frac{3}{7} = 0,429$$

O SABmark e o PREFAB utilizam apenas o conceito do *SP Score* para pontuar alinhamentos. No SABmark tal *score* é chamado de f_d, porém este valor será referenciado aqui como *SP Score*.

Estes *scores*, então, serão utilizados para comparar, em cada coleção de referência, os resultados obtidos pelas versões originais do Muscle, MAFFT e ClustalW com os resultados obtidos por suas respectivas versões modificadas com a nova etapa de agrupamento.

4.3 Determinação dos Limiares

Conforme apresentado no capítulo anterior, o método de agrupamento aqui proposto tem como pré-requisito a determinação de um limiar de distância a ser usado como critério para o agrupamento das seqüências. A determinação automática de um bom limiar não é uma tarefa simples, pois a mesma depende de vários fatores, dentre os quais está a matriz de distâncias adotada[4].

Após realizar experimentos preliminares de alinhamentos múltiplos para analisar o comportamento dos limiares, observou-se que há uma relação entre a média das distâncias (média de todas as entradas da matriz de distâncias) e o melhor limiar para cada caso. Baseado nesta observação, decidiu-se adotar uma estratégia de treinamento para estimar os valores de limiar em função dos valores de média da matriz de distâncias.

[4]Os valores da matriz de distâncias do Muscle variam de 0 a 10; os valores da matriz de distâncias do MAFFT variam de 0 a 3; e os valores da matriz de distâncias do ClustalW variam de 0 a 1, onde os menores valores significam que maior é a similaridade entre o par de seqüências correspondente.

A proposta para a determinação automática de limiares consiste, então, em utilizar uma base de treinamento com o objetivo de descobrir a relação existente entre as médias de distâncias e bons limiares, ou seja, descobrir intervalos de médias de distâncias e um respectivo limiar que melhor atue em cada intervalo de médias estabelecido. Feito isto, aplicam-se estes intervalos e limiares correspondentes para as demais coleções de referência. A hipótese é de que bons resultados também sejam obtidos.

A coleção utilizada como base de treinamento foi o PREFAB, devido a quantidade e variedade de casos que o mesmo possui. Esta coleção foi dividida de acordo com intervalos de média de distâncias. No caso do Muscle o primeiro intervalo é 0 – 0,05 e o último 1,95 – 2, sempre variando a cada 0,05. Lembrando que os valores de distância para o Muscle variam de 0 a 10. Devido a esta variação ser grande, utilizou-se para o cálculo das médias apenas os valores de distância menores ou iguais a 2.

Em seguida, para cada intervalo de médias, o Muscle foi executado para diferentes limiares, começando com o limiar 0, que corresponde à execução do método original, até o limiar 2, sempre variando a cada 0,1. Para cada par de intervalo de médias e limiar, o respectivo *score* médio para o grupo em questão foi calculado, gerando, assim, uma matriz de *scores*. Parte da matriz gerada para o Muscle é mostrada na Figura 4.2.

	...	1,4	1,5	1,6	1,7	1,8	1,9	2,0
...	...	...	...	...	...	...	...	...
1,35–1,40	...	**0,730**	0,724	0,725	0,720	0,723	0,726	0,723
1,40–1,45	...	**0,743**	0,746	0,735	0,739	0,734	0,736	0,727
1,45–1,50	...	0,714	0,708	**0,711**	0,716	0,714	0,715	0,710
1,50–1,55	...	0,716	0,711	**0,715**	0,707	0,713	0,711	0,711
1,55–1,60	...	0,731	0,741	0,731	0,740	**0,739**	0,734	0,720
1,60–1,65	...	0,703	0,700	0,699	0,709	**0,698**	0,704	0,702
1,65–1,70	...	0,701	0,677	0,667	0,697	**0,719**	0,688	0,702
1,70–1,75	...	0,723	0,749	0,744	0,733	0,721	0,681	**0,724**
1,75–1,80	...	0,687	0,700	0,704	0,693	0,667	0,705	**0,734**
1,80–1,85	...	0,743	0,765	0,773	0,754	0,733	0,729	**0,798**
...	...	...	...	...	...	...	...	...

Figura 4.2: Parte da matriz de *scores* gerada para o Muscle. Os *scores* em negrito identificam uma parte do intervalo de limiares.

Para a geração dos intervalos de médias e limiares correspondentes, levou-se em consideração o seguinte fator: para cada limiar dentro de um intervalo, tentar associar o maior número possível de casos.

Este fator justifica o fato de que nem sempre o melhor *score* da linha da matriz é selecionado para constituir um intervalo. Leva-se sempre em consideração um grupo de *scores* em linhas adjacentes para que, então, o melhor limiar em geral seja associado ao intervalo de médias de distâncias.

Os intervalos de médias e limiares correspondentes gerado para o Muscle baseado na matriz de *scores* é apresentado na Figura 4.3. A partir da figura é possível observar que, quando a média das distâncias estiver entre 0 e 0,8 o limiar utilizado será 0,7. Quando a média das distâncias variar entre 0,8 e 0,9 o limiar utilizado será 0,8 e assim sucessivamente.

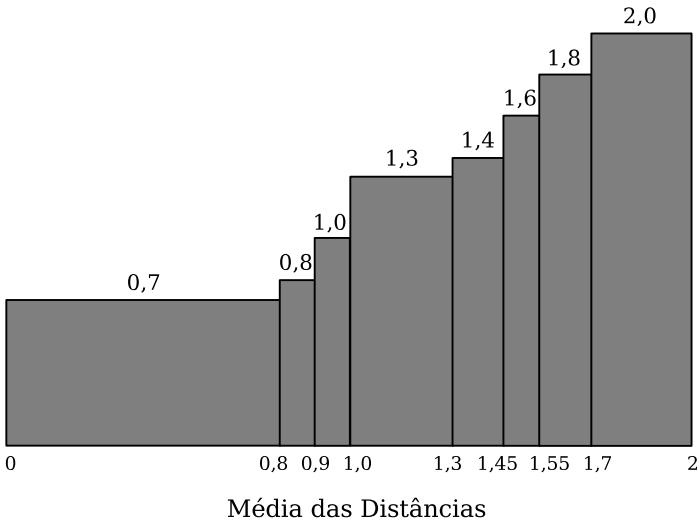

Figura 4.3: Gráfico representando os intervalos das médias de distâncias (eixo x) e seus limiares correspondentes (eixo y) para o método Muscle modificado.

Todo o processo descrito anteriormente foi repetido para o MAFFT, sendo que neste caso, a coleção PREFAB (base de treinamento) foi dividida de acordo com intervalos de média de distâncias começando com 0 – 0,05 e o último 2,95 – 3, sempre variando a cada 0,05. Lembrando que os valores de distância para o MAFFT variam de 0 a 3. Como é um intervalo pequeno, nenhum valor foi descartado para o cálculo da média das distâncias.

Em seguida, para cada intervalo de médias de distâncias, o MAFFT foi executado para diferentes limiares, começando com o limiar 0 até o limiar 3 (ambos correspondem à execução do método MAFFT original), sempre variando a cada 0,1. Para cada par de intervalo de médias de distâncias e limiar, o respectivo *score* médio para o grupo em questão foi calculado, gerando, assim, uma matriz de *scores*. Parte desta matriz gerada para o MAFFT é mostrada na Figura 4.4 seguinte.

	...	0,6	0,7	0,8	0,9	1,0	1,1	...
...	...	...	...	...	...	...	...	
1,10–1,15	...	**0,771**	0,766	0,758	0,769	0,765	0,766	...
1,15–1,20	...	**0,757**	0,755	0,757	0.759	0,756	0,756	...
1,20–1,25	...	0,652	**0,661**	0,659	0.648	0,654	0,650	...
1,25–1,30	...	0,675	**0,675**	0,665	0.668	0,669	0,668	...
1,30–1,35	...	0,614	0,622	0,618	**0,628**	0,629	0,630	...
1,35–1,40	...	0,577	0,586	0,595	**0,588**	0,585	0,578	...
1,40–1,45	...	0,544	0,551	0,540	0,543	0,543	**0,551**	...
1,45–1,50	...	0,534	0,534	0,539	0,530	0,529	**0,544**	...
...	...	...	...	...	...	...	....	...

Figura 4.4: Parte da matriz de *scores* gerada para o MAFFT. Os *scores* em negrito identificam uma parte do intervalo de limiares.

Os intervalos de médias de distâncias e limiares correspondentes gerado para o método MAFFT baseado na matriz de *scores* mostrada acima é apresentado na Figura 4.5 seguinte. Como pode ser observado, quando a média das distâncias estiver entre 0 e 1,1 o limiar utilizado será 0,4. Quando a média das distâncias variar entre 1,1 e 1,2 o limiar utilizado será 0,6 e assim sucessivamente.

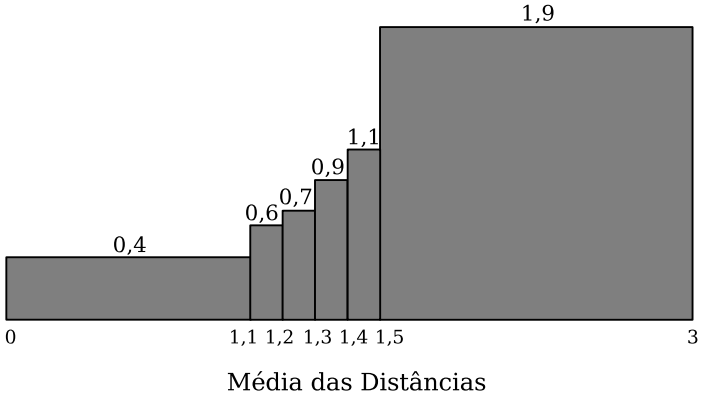

Figura 4.5: Gráfico representando os intervalos das médias de distâncias (eixo x) e seus limiares correspondentes (eixo y) para o método MAFFT modificado.

Novamente, todo este processo foi repetido, só que desta vez para o ClustalW. A coleção PREFAB (base de treinamento) foi dividida de acordo com intervalos de média de distâncias começando com 0 – 0,025 e terminando com 0,975 – 1, sempre variando a cada 0,025. Lembrando que os valores de distância para o ClustalW variam entre 0 e 1. Por ser um intervalo pequeno, assim como ocorre no MAFFT, nenhum valor da matriz de distâncias do ClustalW foi descartado para o cálculo da média das distâncias.

Em seguida, para cada intervalo de médias de distâncias, o ClustalW foi executado para diferentes limiares, começando com o limiar 0 até o limiar 1, sempre variando a cada 0,05. Parte da matriz de *scores* gerada para o ClustalW é mostrada na Figura 4.6 seguinte.

	...	0,60	0,65	0,70	0,75	0,80	...
...	...	...	...	...	...	...	...
0,700–0,725	...	0,567	**0,572**	0,566	0,566	0,593	...
0,725–0,750	...	0,497	**0,494**	0,495	0,501	0,505	...
0,750–0,775	...	0,418	0,419	**0,422**	0,415	0,417	...
0,775–0,800	...	0,396	0,388	**0,404**	0,398	0,377	...
0,800–0,825	...	0,377	0,368	0,374	0,365	**0,381**	...
...	...	...	...	...	...	...	....

Figura 4.6: Parte da matriz de *scores* gerada para o ClustalW. Os *scores* em negrito identificam uma parte do intervalo de limiares.

Os intervalos de médias e limiares correspondentes gerado para o ClustalW baseado na matriz de *scores* é apresentado na Figura 4.7. Isto quer dizer que, quando a média das distâncias estiver entre 0 e 0,475 o limiar utilizado será 0,15. Quando a média das distâncias variar entre 0,475 e 0,525 o limiar utilizado será 0,4 e assim sucessivamente.

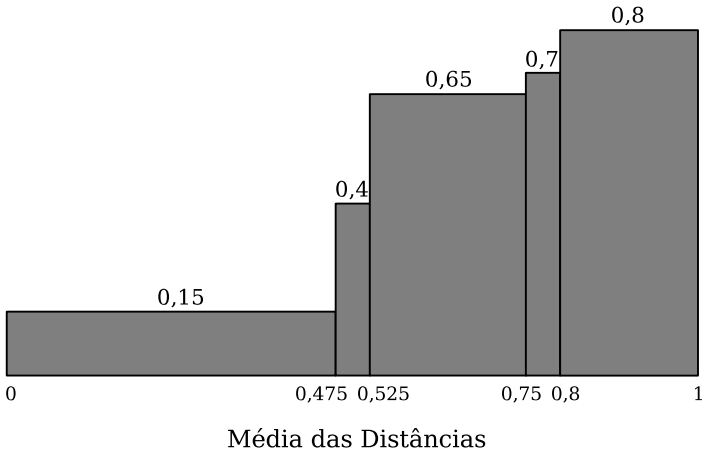

Figura 4.7: Gráfico representando os intervalos das médias de distâncias (eixo x) e seus limiares correspondentes (eixo y) para o método ClustalW modificado.

Cada intervalo encontrado para o Muscle, MAFFT e ClustalW foi aplicado para as demais coleções de referência com o objetivo de constatar a validade dos mesmos. Portanto, todos os experimentos relacionados às novas versões do Muscle, MAFFT e ClustalW e seus respectivos resultados mostrados na seção seguinte são baseados nos intervalos de médias e limiares correspondentes apresentados nas Figuras 4.3, 4.5 e 4.7.

Além da possibilidade do usuário utilizar o limiar automático, o mesmo pode ainda determinar um outro limiar desejado através da linha de comando de cada método modificado, utilizando a opção ***-limiar***.

As linhas de comando para executar os métodos Muscle, MAFFT e ClustalW são mostrados na Tabela 4.1. É importante notar que a opção *-limiar* consiste em uma nova opção adicionada às novas versões de cada método modificado. As linhas de comando para executar o Muscle e o ClustalW representam seus parâmetros padrão, enquanto que a linha de comando para executar o MAFFT representa a sua opção FFT-NS-i [13, 14]. O MAFFT FFT-NS-i foi usado pois o mesmo executa a etapa de refinamento, a qual obtém melhores resultados do que o MAFFT utilizado com seus parâmetros padrão.

Para que o limiar seja determinado automaticamente, basta utilizar a palavra ***auto*** após a opção *-limiar*, caso contrário, basta informar o limiar desejado após a opção.

Tabela 4.1: Linhas de comando para executar o Muscle, MAFFT e ClustalW.

Método	**Comando**
Muscle 3.52	muscle **-limiar** $< L\|auto >$ -in <*infile*> -out <*outfile*>
MAFFT 5.667	mafft –maxiterate 1000 **-limiar** $< L\|auto >$ <*infile*> > <*outfile*>
ClustalW 1.83	clustalw **-limiar** $< L\|auto >$ -infile=<*infile*> -outfile=<*outfile*>

4.4 Resultados Obtidos

Três coleções de referências diferentes foram utilizadas com o propósito de avaliar a viabilidade da estratégia aqui proposta. Os resultados para cada coleção são mostrados nas Tabelas 4.2 – 4.6, nas quais são feitas uma comparação entre o Muscle original e sua versão modificada chamada C-Muscle; uma comparação entre o MAFFT original e sua versão modificada chamada C-MAFFT; e uma comparação entre o ClustalW original e sua versão modificada chamada C-ClustalW. Os melhores resultados em cada caso estão sublinhados.

Os valores apresentados na Tabela 4.2 correspondem ao *SP* e *TC Scores* gerais, ou seja, todo o alinhamento é levado em consideração, associados às cinco referências do BAliBASE denotadas aqui como Ref1 – Ref5. Enquanto que os valores apresentados na Tabela 4.3 correspondem ao *SP* e *TC Scores* medidos apenas nos *core blocks* (partes mais relevantes do alinhamento) conforme os arquivos de anotação disponibilizados pelo BAliBASE.

Como pode ser observado a partir da Tabela 4.2, a nova versão C-Muscle obteve resultados melhores que sua versão original em todas as referências. Por sua vez, o C-MAFFT obteve resultados melhores que sua versão original em todas as referências com exceção das referências 1 e 4 (apenas em *SP Score*) onde houve uma pequena queda dos resultados. A nova versão C-ClustalW também obteve resultados melhores que sua versão original em todas as referências com exceção da referência 2 onde também houve uma pequena perda em *SP Score*.

Tabela 4.2: *SP* e *TC Scores* gerais em cinco referências do BAliBASE para cada versão do Muscle, MAFFT e ClustalW.

	Ref1		**Ref2**		**Ref3**		**Ref4**		**Ref5**	
Method	**SP**	**TC**	**SP**	**TC**	**SP**	**TC**	**SP**	**TC**	**SP**	**TC**
C-Muscle	0,787	0,685	0,883	0,487	0,717	0,436	0,694	0,403	0,830	0,653
Muscle	0,785	0,682	0,878	0,468	0,704	0,415	0,682	0,378	0,828	0,652
C-MAFFT	0,787	0,684	0,876	0,459	0,733	0,466	0,695	0,436	0,822	0,634
MAFFT	0,789	0,686	0,875	0,458	0,726	0,444	0,702	0,418	0,808	0,613
C-ClustalW	0,778	0,670	0,855	0,459	0,649	0,350	0,641	0,348	0,771	0,538
ClustalW	0,777	0,669	0,860	0,456	0,647	0,344	0,631	0,335	0,758	0,508

A partir dos resultados da Tabela 4.3 relacionados aos *core blocks*, pode-se observar que o C-Muscle obteve resultados melhores que sua respectiva versão original em todas as cinco referências do BAliBASE, sem exceção. Por sua vez, o C-MAFFT também obteve resultados melhores que sua versão original em todas as referências com exceção da referência 1 onde houve uma pequena perda nos resultados. A nova versão C-ClustalW obteve resultados melhores que sua versão original nas referências 1, 2 (exceto em *SP Score*), 3 e 5. Na referência 4 tem-se uma queda nos resultados, principalmente em *TC Score*.

Tabela 4.3: *SP* e *TC Scores* medidos apenas em *core blocks* em cinco referências do BAliBASE para cada versão do Muscle, MAFFT e ClustalW.

	Ref1		**Ref2**		**Ref3**		**Ref4**		**Ref5**	
Method	**SP**	**TC**	**SP**	**TC**	**SP**	**TC**	**SP**	**TC**	**SP**	**TC**
C-Muscle	0,881	0,815	0,941	0,635	0,807	0,610	0,901	0,675	0,980	0,921
Muscle	0,879	0,813	0,937	0,602	0,796	0,583	0,889	0,633	0,978	0,914
C-MAFFT	0,882	0,814	0,937	0,589	0,840	0,653	0,922	0,773	0,971	0,890
MAFFT	0,886	0,817	0,936	0,587	0,822	0,618	0,918	0,752	0,938	0,855
C-ClustalW	0,861	0,786	0,927	0,603	0,733	0,503	0,830	0,601	0,875	0,677
ClustalW	0,860	0,785	0,933	0,593	0,723	0,481	0,834	0,623	0,858	0,634

Outra observação interessante é que as melhorias obtidas pelas novas versões dos métodos na referência 1 do BAliBASE são as menos significativas. Isto acontece porque tal referência é constituída em sua grande maioria de alinhamentos entre quatro a seis seqüências, fazendo com que a etapa de agrupamento não faça muita diferença nos resultados finais, pois o agrupamento gerado acaba sendo muito parecido com o original.

A Tabela 4.4 por sua vez, mostra a média dos *SP* e *TC Scores* entre as cinco referências do BAliBASE medidos apenas em *core blocks*, já que estes são mais relevantes, e os respectivos tempos de execução em segundos para cada versão do Muscle, MAFFT e ClustalW. Novamente, é possível observar que as novas versões dos métodos obtiveram os melhores resultados, isto com um aumento irrelevante no tempo de execução.

Tabela 4.4: Média dos *SP* e *TC Scores* medidos apenas em *core blocks* entre as cinco referências do BAliBASE e os respectivos tempos de CPU em segundos para cada versão do Muscle, MAFFT e ClustalW.

Método	**SP**	**TC**	**CPU**
C-Muscle	0,902	0,731	94
Muscle	0,896	0,709	92
C-MAFFT	0,910	0,743	100
MAFFT	0,900	0,726	97
C-ClustalW	0,845	0,634	109
ClustalW	0,842	0,623	107

A Tabela 4.5 mostra os resultados apenas de *SP Scores* obtidos por cada versão do Muscle, MAFFT e ClustalW usando a coleção de referência SABmark *Superfamily*, além dos seus respectivos tempos de execução em segundos. Como pode ser observado, as novas versões C-Muscle, C-MAFFT e C-ClustalW obtiveram resultados melhores do que suas versões originais.

Tabela 4.5: *SP Scores* utilizando a coleção SABmark *Superfamily* e os respectivos tempos de CPU em segundos para cada versão do Muscle, MAFFT e ClustalW.

Método	**Superfamily**	**CPU**
C-Muscle	0,505	618
Muscle	0,502	600
C-MAFFT	0,510	290
MAFFT	0,507	300
C-ClustalW	0,470	120
ClustalW	0,469	120

Porém, as melhorias obtidas no SABmark *Superfamily* não são tão significativas quanto às obtidas no BAliBASE. A mesma explicação dada para o caso da referência 1 do BAliBASE se aplica ao SABmark *Superfamily*. Mesmo que esta referência contenha 310 casos de teste, a maioria destes consiste em alinhamentos entre poucas seqüências, fazendo com que a etapa de agrupamento não cause tanto impacto no resultado final. Com relação aos tempos de execução, pode-se observar que não houve alterações relevantes nos mesmos.

É importante observar também que, existem outros casos onde as seqüências são muito similares entre si. Nestes casos, a técnica de agrupamento também não interfere nos resultados gerados, fazendo com que as melhorias obtidas acabem sendo dissolvidas ao se tirar a média dos *scores* resultantes.

A Tabela 4.6 mostra os resultados apenas de *SP Scores* obtidos por cada versão do Muscle, MAFFT e ClustalW usando a coleção de referência PREFAB, além dos seus respectivos tempos de execução em segundos. Por ser uma coleção grande, o PREFAB foi dividido conforme a média de identidade de cada caso de teste. Como pode ser observado, as novas versões C-Muscle, C-MAFFT e C-ClustalW obtiveram resultados melhores do que suas versões originais em todos os casos, com um aumento muito pequeno no tempo de execução.

Tabela 4.6: *SP Scores* utilizando a coleção PREFAB e os respectivos tempos de CPU em segundos para cada versão do Muscle, MAFFT e ClustalW.

Método	**Geral**	**0–20%**	**20–40%**	**40–70%**	**70–100%**	**CPU**
C-Muscle	0,714	0,733	0,713	0,639	0,772	9402
Muscle	0,708	0,717	0,709	0,636	0,768	9144
C-MAFFT	0,742	0,792	0,733	0,658	0,787	4933
MAFFT	0,736	0,775	0,726	0,657	0,786	4846
C-ClustalW	0,655	0,633	0,637	0,586	0,762	15240
ClustalW	0,647	0,622	0,633	0,576	0,759	15085

Para avaliar a significância das melhorias obtidas a partir dos resultados mostrados anteriormente utilizando as coleções de referência BAliBASE, SABmark *Superfamily* e PREFAB, foi feita uma comparação, conforme mostra a Tabela 4.7, das melhorias médias obtidas por cada método modificado com relação as suas versões originais, com as melhorias médias obtidas pelo Muscle em cada coleção de referência quando comparado com um dos seus principais concorrentes, o método T-Coffee [19], de acordo com os resultados mostrados em [7].

Esta comparação é feita para as três coleções de referência, sendo que para o BAliBASE utilizou-se apenas os resultados obtidos para os *core blocks*, uma vez que estes resultados são mais relevantes do que os gerais, e também porque são os únicos disponibilizados em [7].

Tabela 4.7: Melhorias obtidas nas coleções BAliBASE, SABmark *Superfamily* e PREFAB para cada versão dos métodos Muscle, MAFFT e ClustalW.

	BAliBASE		**SABmark**	**PREFAB**
Muscle x T-Coffee	0,006	–0,004	0,004	0,008
C-Muscle x Muscle	0,006	0,022	0,003	0,007
C-MAFFT x MAFFT	0,010	0,018	0,003	0,006
C-ClustalW x ClustalW	0,004	0,011	0,001	0,007

Como pode ser observado a partir da Tabela 4.7, o Muscle original obtém uma melhoria de 0,006 sobre o T-Coffee em *SP Score* na coleção de referência BAliBASE e as novas versões dos métodos obtêm melhorias similares, e no caso do C-MAFFT uma melhoria maior, ou seja, de 0,010 sobre o MAFFT original.

Com relação ao *TC Score* o Muscle original na média dos resultados das cinco referências do BAliBASE obtém na verdade uma perda no resultados de -0,004, ao contrário das novas versões dos métodos que obtêm melhorias nos resultados, como no caso do C-Muscle que obtém uma melhoria de 0,022 sobre sua versão original. Além do mais, é importante lembrar que o *TC Score* é mais relevante do que o *SP Score*, uma vez que o mesmo corresponde à percentagem de colunas alinhadas corretamente.

Nos casos do SABmark e PREFAB apenas *SP Scores* são calculados e em ambas as referências tem-se que as melhorias obtidas pelas novas versões dos métodos são próximas às melhorias obtidas pelo Muscle original quando comparado com o T-Coffee.

Estas comparações mostram que as melhorias obtidas pelas novas versões dos métodos são significativas, uma vez que estas equiparam-se às melhorias obtidas pelo Muscle original sobre o T-Coffee, lembrando que o Muscle está entre os métodos mais eficazes encontrados na literatura, sempre obtendo bons resultados.

Como uma nova etapa foi adicionada aos métodos Muscle, MAFFT e ClustalW, espera-se que suas novas versões tenham um tempo de execução maior do que suas versões originais. Este acréscimo no tempo de execução é mostrado para cada método em cada coleção de referência na Tabela 4.8 seguinte.

Tabela 4.8: Acréscimo nos tempos de execução para cada método em cada coleção de referência.

Método	**BAliBASE**	**SABmark**	**PREFAB**
C-Muscle	2%	3%	3%
C-MAFFT	3%	0%	2%
C-ClustalW	2%	0%	1%

A partir desta tabela, pode-se observar que o aumento no tempo de execução varia entre 0 a 3%. Tomando como exemplo o C-Muscle, tem-se um acréscimo de 258 segundos com relação ao Muscle original na execução de alinhamentos para a coleção inteira do PREFAB (ver Tabela 4.6), ou seja, 1666 casos de alinhamento. Logo, para casos unitários tem-se um acréscimo de apenas 0,155 segundos, ou seja, um aumento ínfimo no tempo de execução, fazendo com que a utilização da técnica de agrupamento global aqui proposta em métodos de alinhamento múltiplo já existentes seja bastante relevante na obtenção de alinhamentos melhores.

Capítulo 5

Conclusão e Trabalhos Futuros

Este trabalho realizou estudos sobre a aplicação de uma técnica de agrupamento em métodos de alinhamento múltiplo já existentes com o objetivo de auxiliar na determinação da ordem em que as seqüências serão alinhadas e, com isto, obter alinhamentos melhores ao final do processo sem comprometer o desempenho dos métodos originais.

A principal técnica de alinhamento múltiplo utilizada em trabalhos encontrados na literatura é a do alinhamento progressivo, a qual consiste em alinhar primeiramente as seqüências mais similares entre si e depois as seqüências mais distantes são gradativamente adicionadas ao alinhamento.

Como já foi explicado no decorrer deste trabalho, o alinhamento múltiplo progressivo é baseado em uma abordagem gulosa e o erro introduzido uma vez durante um alinhamento parcial não pode ser corrigido subseqüentemente. No caso do alinhamento entre seqüências muito similares entre si, a taxa de erro tende a ser baixa, porém, a medida que as seqüências divergem esta taxa de erro tende a aumentar. Sendo assim, a determinação da ordem de alinhamento das seqüências constitui-se em um passo fundamental no processo de alinhamento múltiplo progressivo.

Em geral, a estratégia de alinhamento progressivo utiliza uma árvore binária, denominada árvore-guia, para estimar a ordem em que as seqüências serão alinhadas. As folhas desta árvore-guia representam seqüências e os nós internos representam alinhamentos múltiplos parciais. O nó raiz representa o alinhamento múltiplo completo e os nós mais distantes da raiz representam os pares de seqüências mais similares entre si. Esta ordem é de máxima importância para que bons alinhamentos múltiplos sejam gerados.

A árvore-guia utilizada pela estratégia de alinhamento progressivo pode ser considerada como uma técnica de agrupamento, uma vez que o alinhamento múltiplo é gerado através do agrupamento de pares de seqüências ou grupos de acordo com a ordem estabelecida pela árvore-guia. O processo de construção desta árvore pode ser ainda considerado como de agrupamento local, pois o mesmo leva em consideração a cada iteração apenas o par de seqüências ou grupos mais similares entre si, desconsiderando as seqüências restantes.

Este comportamento pode causar perda de informação importante sobre o relacionamento entre o par de seqüências ou grupos alinhado e as seqüências restantes. Tal informação pode ser bastante útil no processo de alinhamento como um todo.

Devido à propriedade de agrupamento local da árvore-guia, a ordem estabelecida pela mesma pode, em algum momento, gerar um alinhamento ruim entre duas seqüências devido à prioridade estabelecida pela árvore.

Como a ordem de alinhamento das seqüências possui grande impacto na geração de bons resultados e a árvore-guia por si só toma decisões locais de agrupamento das seqüências, é proposta a utilização de uma técnica de agrupamento global de seqüências como uma etapa preliminar para auxiliar na determinação desta ordem. A técnica de agrupamento global toma decisões a cada passo visando a otimização do resultado final, ao invés do agrupamento local que toma decisões com base na melhor solução a cada passo. É importante ressaltar que, a melhor solução localmente, nem sempre gera o melhor resultado no final.

Para testar a viabilidade e acurácia da nova estratégia aqui proposta, três métodos conhecidos de alinhamento múltiplo foram modificados com o objetivo de aplicar a nova etapa de agrupamento global de seqüências. Tais métodos foram: Muscle, MAFFT e ClustalW. Além disso, foram utilizadas três coleções de referências de alinhamentos múltiplos: BAliBASE, SABmark e PREFAB. As novas versões dos métodos foram, então, comparadas com suas respectivas versões originais.

Os resultados dos experimentos mostram que as novas versões dos métodos com a etapa de agrupamento global realmente obtiveram resultados melhores do que suas versões originais nas três coleções de referência, com um aumento de apenas 3% em média no tempo de execução dos métodos modificados. Além disso, as melhorias obtidas no BAliBASE são maiores do que as obtidas pelo Muscle original quando comparado com o T-Coffee (um dos seus principais concorrentes), e no caso do SABmark e PREFAB, as melhorias obtidas correspondem à media.

Em resumo, pode-se concluir deste trabalho que a utilização da técnica proposta de agrupamento global de seqüências em métodos de alinhamentos múltiplos realmente acrescenta melhorias nos alinhamentos finais gerados, e o mais importante, sem perder a eficiência dos métodos originais aqui testados. A qualidade dos resultados e a eficiência na geração dos mesmos são características indispensáveis ao se propor uma nova técnica de alinhamento múltiplo. Por este motivo, pode-se concluir que este trabalho atingiu seus objetivos.

Trabalhos futuros

O algoritmo de agrupamento global proposto e descrito neste trabalho foi desenvolvido com o propósito de auxiliar na determinação da ordem em que as seqüências serão alinhadas em métodos de alinhamento múltiplo progressivo. Existem outros algoritmos de agrupamento que podem ser considerados globais na literatura e que poderiam ser aplicados ao invés do algoritmo proposto. Entretanto, decidiu-se não implementá-los, uma vez que o foco principal deste trabalho é apenas em avaliar e validar a idéia da aplicação de uma técnica de agrupamento global como forma de melhorar os alinhamentos resultantes. Portanto, o estudo, aplicação e testes de outras técnicas de agrupamento global são deixados para trabalhos futuros.

Outra sugestão importante de trabalho futuro consiste no estudo e verificação do impacto da utilização da nova estratégia proposta de alinhamento múltiplo utilizando o agrupamento global em aplicações práticas existentes que utilizem o resultado de alinhamentos múltiplos, como por exemplo, a predição de estruturas.

Referências Bibliográficas

[1] A. Bahr, J. D. Thompson, J. C. Thierry, and O. Poch. Balibase (benchmark alignment database): enhancements for repeats, transmembrane sequences and circular permutations. *Nucleic Acids Res.*, 29(1):323–326, 2001.

[2] H. Carrillo and D. J. Lipman. The multiple sequence alignment problem in biology. *SIAM J. Appl. Math.*, 48:1073–1082, 1988.

[3] D. Chakrabarti, S. Papadimitriou, D Modha, and C Faloutsos. Fully automatic cross-associations. *KDD*, pages 79–88, 2004.

[4] S. Chakrabarti. *Mining the Web: Discovering Knowledge from Hypertext Data*. Morgan-Kauffman, 2002.

[5] M. O. Dayhoff, R. M. Schwartz, and B. C. Orcutt. *A model of evolutionary change in proteins in Atlas of protein sequence and structure*. National Biomedical Research Foundation, Maryland, 1978.

[6] R. Durbin, S. Eddy, A. Krogh, and G. Mitchison. *Biological sequence analysis*. Cambridge University Press, 1998.

[7] R. C. Edgar. Muscle: a multiple sequence alignment method with reduced time and space complexity. *BMC Bioinformatics*, 5:113, 2004.

[8] R. C. Edgar. Muscle: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Res.*, 32:1792–1797, 2004.

[9] D. F. Feng and R. F. Doolittle. Progressive sequence alignment as a prerequisite to correct phylogenetic trees. *J. Mol. Evol.*, 25:351–360, 1987.

[10] C. Gibas and P. Jambeck. *Desenvolvendo Bioinformática: ferramentas de software para aplicações em biologia*. Campus, Rio de Janeiro, 2001.

[11] D. Gusfield. *Algorithms on strings, tress, and sequences: computer science and computational biology*. Cambridge University Press, 1997.

[12] S. Henikoff and J. G. Henikoff. Amino acid substitution matrices from protein blocks. *Proc. Natl Acad. Sci. USA*, 89:10915–10919, 1992.

[13] K. Katoh, K. Kuma, H. Toh, and T. Miyata. Mafft version 5: improvement in accuracy of multiple sequence alignment. *Nucleic Acids Res.*, 33:511–518, 2005.

[14] K. Katoh, K. Misawa, K. Kuma, and T. Miyata. Mafft: a novel method for rapid multiple sequence alignment based on fast fourier transform. *Nucleic Acids Res.*, 30:3059–3066, 2002.

[15] M. Kimura. *The Neutral Theory of Molecular Evolution*. Cambridge University Press, 1983.

[16] H. Li, K. Zhang, and T. Jiang. Minimum entropy clustering and applications to gene expression analysis. *Proc. of the 3rd International IEEE Computer Society Computational Systems Bioinformatics Conference, Stanford, USA*, pages 142–151, 2004.

[17] S. B. Needleman and C. D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.*, 48:443–453, 1970.

[18] C. Notredame and D. G. Higgins. Saga: sequence alignment by genetic algorithm. *Nucleic Acids Res.*, 24:1515–1524, 1996.

[19] C. Notredame, D. G. Higgins, and J. Heringa. T-coffee: a novel method for fast and accurate multiple sequence alignment. *J. Mol. Biol.*, 302:205–217, 2000.

[20] P. S. Peres and E. S. Moura. Application of clustering technique in multiple sequence alignment. *Proc. of the 12th International Conference, SPIRE 2005, Buenos Aires, Argentina*, pages 202–205, 2005.

[21] N. Saitou and M. Nei. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Mol. Biol. Evol.*, 4:406–425, 1987.

[22] J. C. Setúbal and J. Meidanis. *Introduction to computational molecular biology.* PSW Publishing Company, 1997.

[23] C. E. Shannon. A mathematical theory of communication. *Bell System Techical Journal*, 27:379–423 and 623–656, 1948.

[24] T. F. Smith and M. S. Waterman. Identification of common molecular subsequences. *J. Mol. Biol.*, 147:195–197, 1981.

[25] P. H. A. Sneath and R. R. Sokal. *Numerical Taxonomy.* Freeman, San Francisco, 1973.

[26] 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 Res.*, 22:4673–4680, 1994.

[27] J. D. Thompson, F. Plewniak, and O. Poch. Balibase: a benchmark alignment database for the evaluation of multiple sequence alignment programs. *Bioinformatics*, 15:87–88, 1999.

[28] J. D. Thompson, F. Plewniak, and O. Poch. A comprehensive comparison of multiple sequence alignment programs. *Nucleic Acids Res.*, 27:2682–2690, 1999.

[29] I. Van Walle, I. Lasters, and L. Wyns. Align-m – a new algorithm for multiple alignment of highly divergent sequences. *Bioinformatics*, 20:1428–1435, 2004.

[30] S. Vinga and J. Almeida. Alignment-free sequence comparison-a review. *Bioinformatics*, 19:513–523, 2003.